

WebSphere Application Server V8: Administration and Configuration Guide

Learn about Websphere Application Server V8

Configure and administer a WebSphere system

Deploy applications in a WebSphere environment



Martin Bentancour
Libor Cada
Jing Wen Cui
Marcio d'Amico
Ural Emekci
Sebastian Kapciak
Jennifer Ricciuti
Margaret Ticknor

Redbooks



International Technical Support Organization

WebSphere Application Server V8: Administration and Configuration Guide

November 2011

Note: Before using this information and the product it supports, read the information in “Notices” on page xvii.

First Edition (November 2011)

This edition applies to WebSphere Application Server V8.

© Copyright International Business Machines Corporation 2011. All rights reserved.

Note to U.S. Government Users Restricted Rights -- Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

Notices	xvii
Trademarks	xviii
Preface	xix
The team who wrote this book	xix
Now you can become a published author, too!	xxi
Comments welcome	xxi
Stay connected to IBM Redbooks	xxii
Part 1. Installation and profile management	1
Chapter 1. System management: Technical overview	3
1.1 System management overview	4
1.1.1 Terminology	4
1.1.2 Directory conventions	4
1.1.3 Profiles	5
1.1.4 System management tools	6
1.2 System management in a stand-alone server environment	8
1.3 System management of multiple stand-alone servers	8
1.4 System management in a distributed server environment	11
1.4.1 Centralized changes to configuration and application data	12
1.4.2 Rules for process startup	12
1.4.3 Distributed process discovery	14
1.4.4 Configuration and application data repository	15
1.4.5 File synchronization in distributed server environments	21
1.5 Management of distributed and stand-alone servers	25
1.6 Centralized installation manager	27
1.7 Java Management Extensions (JMX)	29
1.7.1 JMX MBeans	30
1.7.2 JMX usage scenarios	30
1.8 IBM Support Assistant	31
Chapter 2. Installing WebSphere Application Server on distributed systems	33
2.1 IBM Installation Manager overview	34
2.1.1 Terminology overview	34
2.2 Installation Manager installation	35
2.2.1 Installation options	35
2.2.2 Installation	36
2.3 Using the Installation Manager	45
2.3.1 Various modes in Installation Manager	45
2.4 Working with the Installation Manager	50
2.4.1 Installation Manager preferences	50
2.4.2 Repository overview	51
2.4.3 Repository configuration	52
2.4.4 Updating the Installation Manager	54
2.4.5 Key features of Installation Manager	55
2.4.6 Listing packages	56
2.4.7 Examining the log files	56
2.4.8 Uninstalling the Installation Manager	57

2.5	Installing WebSphere Application Server	58
2.5.1	Checking prerequisites	58
2.5.2	Installing WebSphere Application Server using the GUI	58
2.5.3	Creating a response file	66
2.5.4	Installing silently	69
2.6	WebSphere Customization Toolbox	70
2.6.1	Embedded WebSphere Customization Toolbox	70
2.6.2	Stand-alone WebSphere Customization Toolbox	71
2.6.3	Overview of the tools in the WebSphere Customization Toolbox offerings	71
2.6.4	Installing the stand-alone WebSphere Customization Toolbox	72
2.6.5	Starting the WebSphere Customization Toolbox	74
2.6.6	WebSphere Customization Toolbox command line tool	76
Chapter 3. Working with profiles on distributed systems		77
3.1	Types of profiles	78
3.1.1	Application server profile	78
3.1.2	Deployment manager profile	79
3.1.3	Custom profile	79
3.1.4	Cell profile	79
3.1.5	Administrative agent profile	80
3.1.6	Job manager profile	80
3.1.7	Profile generation	81
3.2	Planning for profiles	82
3.3	Building systems with profiles	82
3.3.1	Starting the WebSphere Customization Toolbox Profile Management Tool	82
3.3.2	Common windows and steps for all profiles	83
3.3.3	Creating an application server profile	95
3.3.4	Creating a deployment manager profile	103
3.3.5	Creating a cell profile	107
3.3.6	Creating a custom profile	108
3.3.7	Federating nodes to a cell	112
3.3.8	Creating an administrative agent profile	119
3.3.9	Creating a job manager profile	124
3.3.10	Registering nodes to an administrative agent	125
3.3.11	Deregistering a node from the administrative agent	129
3.3.12	Registering an administrative agent node with a job manager	130
3.4	Managing profiles	135
3.4.1	Using the manageprofiles command	135
3.4.2	Getting help	136
3.4.3	Getting a list of profiles	136
3.4.4	Creating a profile with the manageprofiles command	137
3.4.5	Deleting profiles	139
Chapter 4. Installing WebSphere Application Server on z/OS systems		141
4.1	IBM Installation Manager overview	142
4.2	IBM Installation Manager installation	143
4.2.1	Checking prerequisites	144
4.2.2	Obtaining an Installation Manager installation kit	144
4.2.3	Installing Installation Manager on the system	144
4.3	Working with Installation Manager	146
4.3.1	Installation Manager preferences	146
4.3.2	Repository overview	146
4.3.3	Updating Installation Manager	147

4.3.4	Installing the WebSphere Application Server initial repository	147
4.4	Using Installation Manager	148
4.4.1	Key features of Installation Manager	148
4.4.2	Uninstalling Installation Manager	151
4.5	Installing WebSphere Application Server	151
4.5.1	Installing using the command line	151
4.5.2	Installing additional packages	153
4.5.3	Creating response files	153
4.5.4	Installing silently	154
4.5.5	The post-installer	155
4.5.6	Service information	155
4.5.7	Uninstalling packages	156
4.5.8	Preparing the base z/OS operating system	156
4.6	WebSphere Customization Toolbox	157
Chapter 5. Working with profiles on z/OS systems		159
5.1	Creating WebSphere environments	160
5.1.1	WebSphere Application Server for z/OS	161
5.1.2	WebSphere DMZ secure proxy server for z/OS	161
5.2	Getting started with the Profile Management tool	162
5.3	Creating a sample z/OS Network Deployment cell	166
5.3.1	Creating a deployment manager definition	166
5.3.2	Creating the base application server definition	186
5.3.3	Federating an application server	199
5.3.4	Uploading jobs and associated instructions	204
5.4	Creating a job manager profile	204
5.4.1	Creating the customization definition	204
5.4.2	Uploading the jobs and the associated instructions	212
5.5	Creating an administrative agent profile	212
5.5.1	Creating the customization definition	213
5.5.2	Uploading jobs and the associated instructions	220
Part 2. Administration and configuration techniques		221
Chapter 6. Administration consoles and commands		223
6.1	Introducing the WebSphere administrative consoles	224
6.1.1	Starting and accessing the consoles	224
6.1.2	Logging in to an administrative console	226
6.1.3	Changing the administrative console session timeout	229
6.1.4	The graphical interface	229
6.1.5	Finding an item in the administrative console	236
6.1.6	Updating existing items	240
6.1.7	Adding new items	242
6.1.8	Removing items	243
6.1.9	Starting and stopping items	243
6.1.10	Using variables	244
6.1.11	Saving work	245
6.1.12	Getting help	246
6.2	Securing the administrative console	247
6.2.1	Enabling security after profile creation	248
6.2.2	Administrative security roles	249
6.3	Job manager console	251
6.3.1	Submitting a job with the job manager	253
6.3.2	Distributing files using the job manager	260

6.4 Using command-line tools	261
6.4.1 Command location	261
6.4.2 Key usage parameters	262
6.4.3 Entering commands	262
Chapter 7. Administration of WebSphere processes	265
7.1 Working with the deployment manager	266
7.1.1 Deployment manager configuration settings	266
7.1.2 Starting and stopping the deployment manager	268
7.2 Starting and stopping an administrative agent	271
7.3 Starting and stopping the job manager	271
7.4 Working with application servers	272
7.4.1 Creating an application server	272
7.4.2 Viewing the status of an application server	282
7.4.3 Starting an application server	284
7.4.4 Stopping an application server	289
7.4.5 Viewing runtime attributes of an application server	292
7.4.6 Customizing application servers	293
7.5 Working with nodes in a distributed environment	301
7.5.1 Starting and stopping nodes	301
7.5.2 Node agent synchronization	303
7.5.3 Removing a node from a cell	305
7.5.4 Renaming a node	307
7.5.5 Node groups	308
7.6 Working with clusters	310
7.6.1 Creating application server clusters	310
7.6.2 Viewing the cluster topology	320
7.6.3 Managing clusters	320
7.7 Working with virtual hosts	321
7.7.1 Creating and updating virtual hosts	322
7.8 Managing applications	323
7.8.1 Managing enterprise applications: Administrative console	324
7.8.2 Deploying an enterprise application	325
7.8.3 Uninstalling an enterprise application	330
7.8.4 Starting an enterprise application	331
7.8.5 Stopping an enterprise application	331
7.8.6 Preventing an enterprise application from starting on a server	331
7.8.7 Viewing application details	332
7.8.8 Finding a URL for a servlet or JSP	334
7.9 Enabling process restart on failure	337
7.9.1 Windows	337
7.9.2 UNIX and Linux	339
7.9.3 z/OS	339
Chapter 8. Administration with scripting	343
8.1 Overview of WebSphere scripting	344
8.2 Launching wsadmin	345
8.2.1 Scripting environment properties file	346
8.2.2 Script profile file	347
8.2.3 Connected versus local mode	347
8.3 Command and script invocation	348
8.3.1 Invoking a single command (-c)	348
8.3.2 Running script files (-f)	348

8.3.3 Invoking commands interactively	349
8.4 The wsadmin tool management objects	349
8.4.1 Help	350
8.4.2 AdminControl	351
8.4.3 AdminConfig	351
8.4.4 AdminApp	351
8.4.5 AdminTask	351
8.4.6 Properties file based configuration	352
8.5 Managing WebSphere using script libraries	353
8.5.1 Invoking script libraries	354
8.5.2 Displaying help for script libraries	355
8.5.3 Application script library	356
8.5.4 Resource script library	358
8.5.5 Security script library	362
8.5.6 Server script library	362
8.5.7 System management script library	366
8.6 Assistance with scripting	367
8.6.1 Enabling command assistance	367
8.6.2 Building script files using command assist	369
8.7 Example: Using scripts with the job manager	371
8.7.1 Introduction	371
8.7.2 Creating the customized script	373
8.7.3 Submitting the job	376
8.7.4 Verifying the results	377
8.8 Online resources	378
Chapter 9. Accessing databases from WebSphere	379
9.1 JDBC resources	380
9.1.1 JDBC providers and data sources	380
9.1.2 WebSphere support for data sources	381
9.2 Steps in defining access to a database	383
9.2.1 Creating an authentication alias	384
9.3 Example: Connecting to an IBM DB2 database	384
9.3.1 Creating the JDBC provider	385
9.3.2 Creating the data source	387
9.4 Example: Connecting to an Oracle database	391
9.4.1 Creating the JDBC provider	391
9.4.2 Creating the data source	392
9.5 Example: Connecting to an SQL Server database	394
9.5.1 Creating the JDBC provider	395
9.5.2 Creating the data source	397
9.6 Example: Connecting to an Informix Dynamic Server database	399
9.6.1 Creating the JDBC provider	400
9.6.2 Creating the data source	401
9.7 Configuring connection pooling properties	403
9.7.1 WebSphere Application Server data source properties	407
9.7.2 Extended DB2 data source	409
9.7.3 JDBCProviderManagement group commands	409
9.7.4 DB2 Lock sharing between transaction branches	410
Chapter 10. Accessing EIS applications from WebSphere	413
10.1 JCA resource adapters	414
10.1.1 WebSphere Application Server JCA support	415

10.2 Resource adapters	415
10.2.1 Connection factory	416
10.2.2 Installing and configuring resource adapters	416
10.3 Configuring J2C connection factories	420
10.4 Resource authentication	422
10.4.1 Container-managed authentication	423
10.4.2 Component-managed authentication	423
Chapter 11. Configuring messaging providers	425
11.1 Messaging providers introduction	426
11.2 Configuring the default messaging provider	426
11.2.1 Configuring a connection factory	426
11.2.2 Configuring JMS destinations	429
11.2.3 Configuring JMS activation specifications	431
11.3 Configuring the WebSphere MQ provider	432
11.3.1 Configuring a connection factory	432
11.3.2 Configuring MQ destinations	435
11.3.3 Configuring MQ activation specifications	439
11.4 Configuring a generic JMS provider	440
11.4.1 Configuring a connection factory	440
11.4.2 Configuring JMS destinations	441
Chapter 12. Configuring and managing web servers	443
12.1 Web server support overview	444
12.1.1 Request routing using the plug-in	445
12.1.2 Web server and plug-in management	445
12.2 Installation	449
12.3 Web server configuration using WebSphere Customization Toolbox (WCT)	450
12.3.1 Configuration files	451
12.3.2 Stand-alone server environment	451
12.3.3 Distributed server environment	454
12.3.4 Configuring a remote web server in a distributed environment	456
12.4 Working with web servers	464
12.4.1 Manually defining nodes and web servers	465
12.4.2 Viewing the status of a web server	469
12.4.3 Starting and stopping a web server	469
12.4.4 IBM HTTP Server remote administration	470
12.4.5 Mapping modules to servers	474
12.5 Working with the plug-in configuration file	476
12.5.1 Regenerating the plug-in configuration file	477
12.5.2 Propagating the plug-in configuration file	483
12.5.3 Modifying the plug-in request routing options	484
12.6 IBM HTTP Server and Web Server Plug-ins for IBM WebSphere Application Server for z/OS	486
12.6.1 IBM HTTP Server	486
12.6.2 Web Server Plug-ins for IBM WebSphere Application Server for z/OS	487
Part 3. Managing distributed systems	493
Chapter 13. Performance tuning	495
13.1 Performance tuning facts	496
13.2 Using the queue analogy to tune WebSphere resource pools	496
13.2.1 Upstream queuing	498
13.2.2 Data source tuning	499

13.2.3 EJB container	501
13.2.4 Web container tuning	502
13.2.5 Web server tuning.	504
13.2.6 Determining the optimum queue sizes	504
13.2.7 Estimating web container and ORB thread pool initial sizes.	506
13.3 JVM tuning	506
13.3.1 Garbage collection policies	507
13.3.2 Setting maximum and minimum heap sizes	510
13.3.3 Sizing the nursery and tenured space when using the gencon policy	510
13.3.4 Using compressed references	511
13.4 Other tuning considerations	512
13.4.1 Dynamic caching.	512
13.4.2 The pass by reference parameter.	512
13.4.3 Large page support.	513
13.4.4 High Performance Extensible Logging	513
13.4.5 Application tuning	514
13.5 Tools	514
13.5.1 Tivoli Performance Viewer	514
13.5.2 Collecting Java dumps and core files using the administrative console	515
13.5.3 IBM Pattern Modelling and Analysis Tool for Java Garbage Collector	515
13.5.4 IBM Monitoring and Diagnostic tools for Java.	515
13.5.5 IBM HTTP server status monitoring page.	516
13.5.6 WebSphere performance advisors	516
Chapter 14. Clustering, workload management, and high availability	519
14.1 Clustering	520
14.1.1 Clustering for scalability and failover.	521
14.1.2 Creating a cluster	523
14.2 Workload management.	525
14.2.1 Components that can be workload managed	526
14.2.2 WLM benefits	529
14.3 High availability and failover	529
14.3.1 Overview	529
14.3.2 WebSphere Application Server high availability and failover	530
14.3.3 How high availability features work.	536
Chapter 15. Monitoring distributed systems	541
15.1 Overview	542
15.1.1 Monitoring scenarios	543
15.2 Enabling monitoring infrastructures.	545
15.2.1 PMI defaults and monitoring settings	545
15.2.2 Enabling request metrics	552
15.3 Viewing the monitoring data	556
15.3.1 Starting TPV monitoring and configuring settings.	556
15.3.2 Exploring Tivoli Performance Viewer data views	560
15.4 Monitoring scenarios	566
15.4.1 Database interactions	566
15.4.2 Threading resources	567
15.4.3 JVM memory usage	570
15.4.4 Request level details.	572
15.5 IBM Tivoli Composite Application Manager for WebSphere Application Server	577
15.5.1 Installing the data collector	578
15.5.2 Configuring IBM Tivoli Composite Application Manager for WebSphere metrics	578

578	
15.5.3 Viewing IBM Tivoli Composite Application Manager for WebSphere data	582
15.6 Additional resources for monitoring.	586
15.6.1 Verbose garbage collection	586
15.6.2 Java dump and core files	589
15.6.3 Basic logging.	590
15.6.4 Advanced logging	592
15.6.5 Operating system monitoring	597
15.6.6 Summary of monitoring tips	597
Part 4. Managing z/OS systems.	599
Chapter 16. Performance tuning	601
16.1 Introduction to WebSphere Application Server for z/OS V8 performance.	602
16.2 External factors and z/OS specifics	603
16.2.1 Getting the most benefit from collocation	603
16.2.2 Addressing hardware configuration.	603
16.2.3 z/OS tuning tips.	603
16.3 Performance tuning templates	606
16.4 64-bit considerations.	607
16.4.1 Enablement of 64-bit mode	607
16.4.2 Effects of switching to 64-bit mode	608
16.5 JVM tuning	613
16.5.1 Default garbage collection	613
16.5.2 General JVM suggestions.	614
16.6 Connection pool tuning	618
16.7 Runtime provisioning.	619
16.8 Pass by reference	620
16.9 Logging and tracing.	621
16.9.1 HPEL overview	621
16.9.2 Enabling HPEL mode	621
16.9.3 z/OS logging and tracing tips	621
16.10 Tuning workload management on z/OS systems	625
16.10.1 The concept of workload management on z/OS systems.	625
16.10.2 Classification rules	626
16.10.3 Classification XML	627
16.10.4 Commands and tools	628
16.11 Fast response cache accelerator and caching	629
16.11.1 FRCA overview	630
16.11.2 Enabling FRCA in WebSphere Application Server	630
16.11.3 Cache specification XML	637
16.11.4 FRCA and RACF integration.	638
16.11.5 Caching enhancements in WebSphere Application Server V8.	638
16.11.6 Using IBM Extended Dynamic Cache Monitor to supervise caching	638
16.12 Using WebSphere for z/OS Optimized Local Adapters.	639
16.12.1 Introduction to Optimized Local Adapters.	639
16.12.2 Enabling WebSphere for z/OS Optimized Local Adapters	641
16.13 IBM HTTP Server Status monitoring page	644
16.14 Tools	644
Chapter 17. Clustering and high availability.	647
17.1 Clustering on z/OS systems	648
17.1.1 Clustering for scalability and failover.	648
17.1.2 Creating a cluster on a z/OS system.	648

17.2 High availability	652
17.2.1 High availability manager	652
17.2.2 Core groups	655
17.2.3 High availability policies and groups	673
17.3 Failover and failback	676
17.3.1 High availability and failover of singletons	676
17.3.2 Data replication domains	687
17.3.3 Session management replication	689
17.3.4 EJB stateful session bean replication	691
17.3.5 Cache replication	694
17.3.6 Resource workload routing	695
17.3.7 High availability application update rollout	699
17.3.8 Additional resources	702
Chapter 18. Monitoring z/OS systems	705
18.1 Overview	706
18.2 Monitoring from the administrative console	707
18.3 IBM Tivoli Composite Application Manager for WebSphere Application Server	707
18.3.1 Installing the data collector	708
18.3.2 Configuring IBM Tivoli Composite Application Manager for WebSphere metrics	708
18.3.3 Viewing IBM Tivoli Composite Application Manager for WebSphere data	718
18.4 Additional resources for monitoring	718
18.4.1 IBM Support Assistant	718
18.4.2 Verbose garbage collection	718
18.4.3 Java dump and core files	721
18.4.4 Basic logging	722
18.4.5 Advanced logging	723
18.4.6 z/OS monitoring	729
18.4.7 Summary of monitoring tips	733
Part 5. Working with applications	735
Chapter 19. New features for application development and deployment	737
19.1 Java Enterprise Edition 6 support	738
19.2 Integrated standards-base programming models and extensions	739
19.2.1 Session Initiation Protocol applications	739
19.2.2 Java batch programming model	739
19.2.3 OSGi applications programming model	741
19.2.4 Communications enabled applications	742
19.2.5 Service Component Architecture programming model	742
19.2.6 Extensible Markup Language programming model	744
19.2.7 Integrated Web Services support	744
19.2.8 Simplified development of server-side REST applications using Java API for RESTful Web Services	744
19.3 Monitored directory support	745
19.4 Development and deployment tools	745
19.4.1 IBM Assembly and Deploy Tools for WebSphere Administration	745
19.4.2 IBM Rational Application Developer Standard Edition for WebSphere Software Version 8	745
19.4.3 IBM Rational Application Developer for WebSphere Software Version 8	746
Chapter 20. Understanding class loaders	747
20.1 JVM class loaders	748

20.2 WebSphere Application Server and Java EE application class loaders	751
20.2.1 WebSphere extensions class loader.	753
20.2.2 Application and web module class loaders.	754
20.2.3 Handling Java Native Interface code	755
20.3 Configuring class loaders for Java EE applications	756
20.3.1 Application server class loader policies	757
20.3.2 Class loading and delegation mode	759
20.3.3 Shared libraries.	761
20.3.4 Class loader viewer.	762
20.4 Learning class loaders for Java EE by example	762
20.4.1 Example 1: Simple web module packaging	763
20.4.2 Example 2: Adding an EJB module and utility jar	765
20.4.3 Example 3: Changing the WAR class loader delegation mode.	766
20.4.4 Example 4: Sharing utility JAR files using shared libraries	768
20.5 OSGi class loaders	773
Chapter 21. Packaging and deploying Java EE applications	775
21.1 Java EE applications introduction	776
21.1.1 Java EE 6 EAR files	776
21.1.2 Development tools	777
21.1.3 Enterprise applications	777
21.1.4 EJB 3.1 modules.	778
21.1.5 JPA persistence units	781
21.1.6 JPA access intents	782
21.1.7 Resource adapters	783
21.1.8 Web modules	783
21.1.9 WebSphere extensions to web modules.	784
21.1.10 EJB 3.1 content in WAR modules.	787
21.2 Preparing to use the sample application.	788
21.2.1 Downloading the application	788
21.2.2 Importing the application to the development tool.	789
21.2.3 Customizing the sample application	789
21.2.4 Creating the ITSO Bank DB2 database	790
21.3 Configuring web module extensions	791
21.4 Packaging recommendations	793
21.5 Creating WebSphere Enhanced EAR files	793
21.5.1 Configuring a WebSphere Enhanced EAR	794
21.5.2 Configuring application options.	795
21.5.3 Configuring the JDBC provider and data source for DB2	796
21.5.4 Configuring a virtual host	801
21.5.5 Setting the default virtual host for web modules	802
21.5.6 Examining the WebSphere Enhanced EAR file	803
21.6 Exporting an application project to an EAR file	804
21.7 Preparing the runtime environment for the application	805
21.7.1 Creating an environment variable for the application file directory	805
21.7.2 Creating the ITSO Bank application server.	806
21.7.3 Defining the ITSO Bank virtual host	809
21.7.4 Creating the virtual host for IBM HTTP Server and Apache	810
21.7.5 Creating a DB2 JDBC provider and data source	811
21.8 Deploying the application	818
21.9 Deploying business-level applications	822
21.9.1 Creating a business-level application	825
21.10 Deploying application clients	828

21.10.1	Installing application clients	829
21.10.2	Preparing the sample application	829
21.10.3	Launching the J2EE client	830
Chapter 22.	Updating Java EE applications	833
22.1	Working with applications	834
22.2	Replacing an entire application EAR file	834
22.3	Replacing or adding an application module	836
22.3.1	Replacing or adding single files in an application or module	837
22.3.2	Removing application content	837
22.3.3	Performing multiple updates to an application or module	838
22.3.4	Rolling out application updates to a cluster	840
22.3.5	Hot deployment and dynamic reloading	843
22.4	Using a monitored directory	844
22.4.1	Setting up a monitored directory	846
22.4.2	Working with a monitored directory	846
Chapter 23.	Working with SCA applications	851
23.1	SCA application introduction	852
23.1.1	SCA component	853
23.1.2	SCA composite	853
23.1.3	SCA contribution	855
23.2	Preparing to use the sample application	856
23.2.1	Downloading the application	856
23.2.2	Importing the application to the development tool	857
23.2.3	Completing the service definition	857
23.3	Packaging an SCA application for deployment	858
23.3.1	Creating the contribution	859
23.3.2	Exporting the SCA application for deployment	862
23.4	Deploying an SCA application	863
23.4.1	Importing the SCA archive file as an asset	863
23.4.2	Creating a new business-level application	866
23.4.3	Adding the new asset to the business-level application	867
23.4.4	Starting and verifying the business-level application	870
Chapter 24.	Working with OSGi applications	871
24.1	OSGi overview	872
24.1.1	OSGi application model	872
24.1.2	OSGi application life cycle	874
24.1.3	Enterprise OSGi	875
24.2	Using the sample application	876
24.3	Packaging OSGi applications	879
24.3.1	Enterprise bundle archives	879
24.3.2	Composite bundle archives	880
24.3.3	Common OSGi patterns	880
24.3.4	Sample application packaging	880
24.4	Exporting OSGi applications	882
24.5	Deploying OSGi applications	883
24.5.1	Deploying the application	884
24.6	Updating OSGi applications	886
Chapter 25.	Session management	889
25.1	HTTP session management	890
25.2	Session management configuration	890

25.2.1	Session management properties	890
25.2.2	Accessing session management properties	891
25.3	Session identifiers	891
25.3.1	Choosing a session tracking mechanism	892
25.3.2	Cookies	894
25.3.3	URL rewriting	895
25.4	Local sessions	897
25.5	General properties for session management	898
25.6	Session affinity	900
25.7	Session affinity and failover	901
25.8	Persistent session management	903
25.8.1	Enabling database persistence	905
25.8.2	Memory-to-memory replication	908
25.8.3	Session management tuning	915
25.8.4	Larger DB2 page sizes and database persistence	921
25.8.5	Single and multi-row schemas (database persistence)	921
25.8.6	Contents written to the persistent store using a database	923
25.9	Invalidating sessions	925
25.10	Session listeners	926
25.11	Session security	927
25.12	Session performance considerations	928
25.12.1	Session size	928
25.12.2	Reducing persistent store I/O	931
25.12.3	Multirow persistent sessions: Database persistence	931
25.12.4	Managing your session database connection pool	932
25.12.5	Session database tuning	933
25.13	Stateful session bean failover	933
25.13.1	Enabling stateful session bean failover	933
25.13.2	Stateful session bean failover consideration	936
Part 6.	Maintenance	939
Chapter 26. Managing your environment with the centralized installation manager		941
26.1	The centralized installation manager prerequisites	942
26.1.1	Linux and UNIX target requirements	942
26.1.2	Windows target requirements	943
26.1.3	IBM i targets	943
26.1.4	Additional requirements	943
26.2	Planning considerations	944
26.2.1	WebSphere Application Server V8	944
26.2.2	WebSphere Application Server V6.1 and V7	945
26.3	Working with the centralized installation manager and WebSphere Application Server Version 8	945
26.3.1	Installation Manager	945
26.3.2	Accessing the centralized installation manager	947
26.4	Working with the centralized installation manager and WebSphere Application Server V6.1 and V7	948
26.4.1	IBM Update Installer	948
26.4.2	The centralized installation manager repository structure	948
26.4.3	Package types	949
26.4.4	Accessing the central installation manager	950
26.5	Managing WebSphere Application Server V8 environment with the centralized installation manager	952

26.5.1 Adding new targets	952
26.5.2 Installing Installation Manager on remote targets	955
26.5.3 Installing a Secure Shell (SSH) public key	961
26.5.4 Installing WebSphere Application Server binaries on remote host	962
26.5.5 Creating a WebSphere Application Server profile on a remote target	965
26.5.6 Registering and unregistering the profile in the Job Manager console	967
26.5.7 Working with remote targets	969
26.5.8 Installing maintenance to remote targets	973
26.5.9 Using the centralized installation manager with a command line	975
26.6 Managing WebSphere Application Server V6.1 and V7 with the centralized installation manager	976
26.6.1 Installing the IBM Installation Factory	976
26.6.2 Creating the centralized installation manager repository	976
26.6.3 Adding packages to the centralized installation manger repository when the deployment manager is connected to the Internet	977
26.6.4 When the deployment manager is not connected to the Internet	982
26.6.5 Adding and removing additional installation targets	984
26.6.6 Installing a Secure Shell public key	985
26.6.7 Installing packages to the target systems	987
26.6.8 Product installation	987
26.6.9 Installing maintenance to a target system	989
26.6.10 Uninstalling packages	992
26.6.11 The centralized installation manager AdminTask commands	993
Chapter 27. System recovery	995
27.1 Backups	996
27.1.1 Backing up a profile	996
27.1.2 Restoring a profile	997
27.1.3 Exporting and importing profiles	998
27.2 Recovery of transactions	1000
27.3 Environment recovery	1001
27.3.1 Rapid node recovery	1002
27.3.2 Moving nodes to new environments	1004
27.3.3 Recreating cells from a template	1008
Related publications	1011
IBM Redbooks	1011
Other publications	1011
Online resources	1011
Help from IBM	1013

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785 U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. These and other IBM trademarked terms are marked on their first occurrence in this information with the appropriate symbol (® or ™), indicating US registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at <http://www.ibm.com/legal/copytrade.shtml>

The following terms are trademarks of the International Business Machines Corporation in the United States, other countries, or both:

AIX 5L™	Language Environment®	RMF™
AIX®	Lotus®	System z10®
CICS®	MVS™	System z®
ClearCase®	OmniFind®	SystemPac®
DataPower®	OS/400®	Tivoli®
DB2 Universal Database™	Parallel Sysplex®	VTAM®
DB2®	Passport Advantage®	WebSphere®
developerWorks®	POWER®	z/Architecture®
Domino®	RACF®	z/OS®
eServer™	Rational Team Concert™	z10™
i5/OS®	Rational®	z9®
IBM®	Redbooks®	zSeries®
IMS™	Redbooks (logo)  ®	
Informix®	Resource Measurement Facility™	

The following terms are trademarks of other companies:

Intel, Intel logo, Intel Inside logo, and Intel Centrino logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Microsoft, Windows, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java, and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.

Preface

This IBM® Redbooks® publication provides system administrators and developers with the knowledge to configure an IBM WebSphere® Application Server Version 8 runtime environment, to package and deploy applications, and to perform ongoing management of the WebSphere environment.

As one in a series of IBM Redbooks publications and IBM Redpapers publications for V8, the entire series is designed to give you in-depth information about key WebSphere Application Server features. In this book, we provide a detailed exploration of the WebSphere Application Server V8 runtime administration process.

This book includes configuration and administration information for WebSphere Application Server V8 and WebSphere Application Server Network Deployment V8 on distributed platforms and WebSphere Application Server for z/OS® V8.

The following publications are prerequisites for this book:

- ▶ *WebSphere Application Server V8.0 Technical Overview*, REDP-4756
- ▶ *IBM WebSphere Application Server V8 Concepts, Planning, and Design Guide*, SG24-7957

The team who wrote this book

This book was produced by a team of specialists from around the world working at the International Technical Support Organization, Raleigh Center.



Figure 1 Left to right: Margaret, Jennifer, Jing Wen, Martin, Ural, Libor, Marcio, and Sebastian

Martin Bentancour is a Senior WebSphere Application Server and WebSphere Portal consultant for ISA LTDA, an IBM premier Business Partner located in Montevideo, Uruguay. He specializes in high availability and disaster recovery architectures for mission-critical environments. His main responsibility is to keep every component of each solution up and running 24x7, including WebSphere Application Server, IBM WebSphere Portal, OmniFind® Enterprise Edition, IBM HTTP Server, and IBM DB2® HADR databases. Previously, Martin worked as a Java EE developer, a task he still performs on occasion for the Portal development team.

Libor Cada is a IT Specialist working for Integrated Delivery Center SSO, in Brno, The Czech Republic. He has 8 years of experience in the IT and banking industries on mainframe IBM eServer™ zSeries® and Linux on System z® environments. He previously held a position of IBM z/OS DBDC systems programmer for CICS®, DB2, WMQ, and IBM IMS™ products. He currently supports customers from multiple geographies in his role as a WebSphere Application Server and z/OS system programmer.

Jing Wen Cui is a software engineer working for China Development Lab, in Beijing, China. She has four years of experience in testing IBM WebSphere Business Process Management products. She worked as functional verification tester and a functional integration verification tester for WebSphere Process Server, and is now working as a system verification tester for WebSphere Business Process Management. Her areas of expertise include service-oriented architecture (SOA), implementation, and problem determination for IBM WebSphere Application Server and WebSphere Process Server. She is certified in WebSphere Application Server V6.1.

Marcio d'Amico is a Senior IT Specialist for IBM Global Technology Services in Brazil. He has 18 years of experience in IT and has been supporting WebSphere Application Server for the past 4 years. Previously, Marcio worked in a services team providing solutions and support for mainframe networks, and security, host integration, and financial products. He holds a Bachelor's degree in Systems Analysis from Pontifícia Universidade Católica de Campinas. He is certified in WebSphere Application Server and has co-authored another IBM Redbooks publication.

Ural Emekci is a Technical Sales Specialist for Software Group, IBM Turkey. He has been working on the WebSphere product portfolio for 7 years. His areas of expertise include infrastructure architecture, SOA, and WebSphere products. He has extensive industry knowledge and hands-on project experience in multiple sectors, such as finance, telecommunications, public, and healthcare. Currently, he is responsible for delivering frequent customer presentations, solution designs, technical workshops, large scale proof of concepts (POCs), proofs of technology (POTs), and technical training to customers and Business Partners. He holds a Bachelor's degree in Computer Science from Sabancı University. He is certified in WebSphere Application Server, IBM WebSphere Lombardi Edition, and SOA.

Sebastian Kapciak is an Advisory IT Specialist working for IBM Global Technology Services in Warsaw, Poland. Sebastian joined IBM in 2007 and has over 6 years of experience in software architecture and development. His areas of expertise include system integration and JEE technologies. He also specializes in the WebSphere Application Server, IBM DataPower® appliances, and IBM Tivoli® Access Manager. Sebastian holds a Master's degree in Information Technology from the University of Technology of Warsaw.

Jennifer Ricciuti is a Course Developer and Instructor in WebSphere Education. She has 15 years of experience in developing and delivering education courses on various WebSphere products, including WebSphere Application Server, WebSphere Process Server, and IBM WebSphere eXtreme Scale. Jennifer has contributed to the development of several IBM Certification Tests for both WebSphere Application Server and WebSphere Process Server. Her areas of expertise include course design and development. She holds a Bachelor's degree in Computer Science from Point Park University. She works and resides in Pittsburgh, Pennsylvania.

Margaret Ticknor is an IT Specialist at the IBM ITSO Center in Raleigh. She writes about WebSphere products and solutions. Prior to joining the ITSO in 1997, Margaret worked in Endicott, supporting internal VM customers. Margaret attended the Computer Science program at State University of New York at Binghamton.

Thanks to the following people for their contributions to this project:

Carla Sadtler, Rich Conway, Bob Haimowitz, Tamikia Barrow, Debbie Willmschen, Stephen Smith, Karen Lawrence, Linda Robinson

International Technical Support Organization, Raleigh Center

Felix Wong
IBM Canada

Tom Alcott, Dave Follis, Tony Garcia
IBM US

Thanks to the authors of the previous editions of this book.

- ▶ Authors of the first editions of this book, *WebSphere Application Server V6.1: System Management and Configuration*, SG24-7304, published in November 2006:
Carla Sadtler, Fabio Albertoni, Bernardo Fagalde, Thiago Kleinubing, Henrik Sjostrand, Ken Worland, Lars Bek Laursen, Martin Phillips, Martin Smithson, and Kwan-Ming Wan.
- ▶ Authors of the second edition, *WebSphere application Server V7: Administration and Configuration Guide*, SG24-7615, published in March 2010:
Fabio Albertoni, Leonard Blunt, Michael Connolly, Stefan Kwiatkowski, Carla Sadtler, Thayaparan Shanmugaratnam, Henrik Sjostrand, Saori Tanikawa, Margaret Ticknor, and Joerg-Ulrich Veser

Now you can become a published author, too!

Here's an opportunity to spotlight your skills, grow your career, and become a published author—all at the same time! Join an ITSO residency project and help write a book in your area of expertise, while honing your experience using leading-edge technologies. Your efforts will help to increase product acceptance and customer satisfaction, as you expand your network of technical contacts and relationships. Residencies run from two to six weeks in length, and you can participate either in person or as a remote resident working from your home base.

Find out more about the residency program, browse the residency index, and apply online at:
ibm.com/redbooks/residencies.html

Comments welcome

Your comments are important to us!

We want our books to be as helpful as possible. Send us your comments about this book or other IBM Redbooks publications in one of the following ways:

- ▶ Use the online **Contact us** review Redbooks form found at:
ibm.com/redbooks
- ▶ Send your comments in an email to:
redbooks@us.ibm.com

- ▶ Mail your comments to:
IBM Corporation, International Technical Support Organization
Dept. HYTD Mail Station P099
2455 South Road
Poughkeepsie, NY 12601-5400

Stay connected to IBM Redbooks

- ▶ Find us on Facebook:
<http://www.facebook.com/IBMRedbooks>
- ▶ Follow us on Twitter:
<http://twitter.com/ibmredbooks>
- ▶ Look for us on LinkedIn:
<http://www.linkedin.com/groups?home=&gid=2130806>
- ▶ Explore new Redbooks publications, residencies, and workshops with the IBM Redbooks weekly newsletter:
<https://www.redbooks.ibm.com/Redbooks.nsf/subscribe?OpenForm>
- ▶ Stay current on recent Redbooks publications with RSS Feeds:
<http://www.redbooks.ibm.com/rss.html>



Part 1

Installation and profile management



System management: Technical overview

In this chapter, we provide a technical overview of the system management functionality of WebSphere Application Server. This information can help you understand how system administration occurs. It is particularly useful in a multi-server environment to understand the distributed administration and synchronization topics.

This chapter assumes that you are familiar with the concepts in *WebSphere Application Server V8.0 Technical Overview*, REDP-4756.

We cover the following topics:

- ▶ System management overview
- ▶ System management in a stand-alone server environment
- ▶ System management of multiple stand-alone servers
- ▶ System management in a distributed server environment
- ▶ Management of distributed and stand-alone servers
- ▶ Centralized installation manager
- ▶ Java Management Extensions (JMX)
- ▶ IBM Support Assistant

1.1 System management overview

At first glance, system management concepts in WebSphere Application Server might seem complex. However, the fact that the system management architecture is based on Java Management Extensions (JMX), and the fact that WebSphere Application Server provides easy-to-use administration tools make it fairly simple to use and understand.

1.1.1 Terminology

There are differences in how WebSphere Application Server handles administration, depending on the environment that you have set up. As you go through this book, you will see the following terms used:

- ▶ *Stand-alone server environment* refers to a single server that is not managed as part of a cell. (The server has not been “federated” to the cell). With the Base and Express offerings, this option is your only option. You can also create a stand-alone server environment with the Network Deployment offering.
- ▶ *Distributed server environment* refers to the situation where you have multiple servers managed from a single deployment manager in the cell. We also refer to these servers as *managed servers*. This is only valid with the Network Deployment offering.
- ▶ *Application server* refers to the process that provides the functions that are required to support and host user applications. An application server can be a *stand-alone application server*, a *distributed application server* managed by a deployment manager, or a *stand-alone application server* that is managed from an administrative agent.
- ▶ *Managed processes* refer to the deployment manager, nodes (node agents), and application servers.
- ▶ *Flexible management* refers to asynchronous job management through the job manager (available in Network Deployment only) and managing multiple unfederated application servers from an administrative agent (available on all offerings).
- ▶ *System* refers to one physical computer.

1.1.2 Directory conventions

Throughout this book, we use the following notations when indicating the location of files and commands:

- ▶ *install_root* is used to denote the installation directory for a product. The default installation directory locations can be found at the following website:
http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.installation.express.doc/info/exp/ae/rins_dircon.html
- ▶ *profile_root* is used to denote the home profile for a directory. This is equivalent to:
install_root/profiles/profile_name
Special instances of *profile_root* are used to denote the profile home for the following processes:
 - Deployment manager:
dmgr_profile_root
 - Administrative agent:
adminAgnt_profile_root

- Job manager:

jmgr_profile_root

1.1.3 Profiles

To create an application server, node agent, deployment manager, administrative agent, or a job manager, you must first install the WebSphere Application Server core product files and then create their respective profiles, as shown in Figure 1-1.

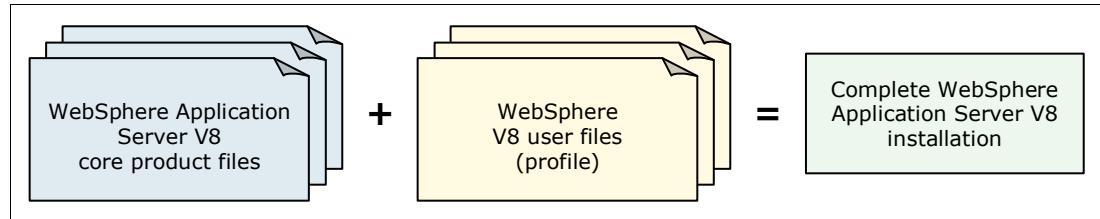


Figure 1-1 *Product files and profiles*

When a profile is created, the configuration details for the server are stored in a folder that is unique to the profile. You can think of the product files as the runtime component, and the profiles as the input for the run time.

There are seven profile types that can be created, as shown in Figure 1-2.

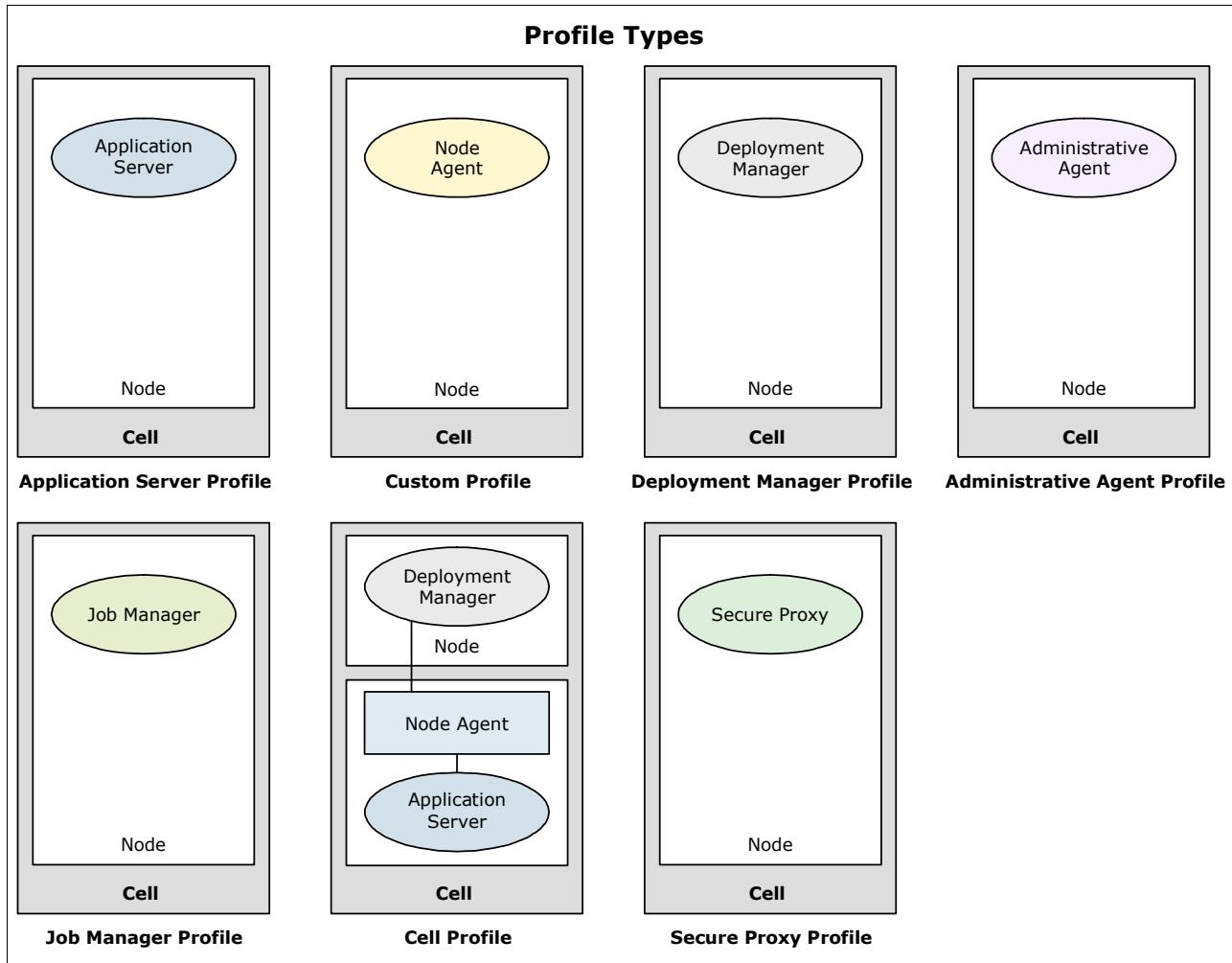


Figure 1-2 Profile types

Notice that a server always belongs to a node, and the node always belongs to a cell. The purpose of the node and cell is to define the administrative domain. The node and cell are logical concepts used to group servers, and to define their scope. Every time a profile is created, a node is created. Therefore, there is a one-to-one relationship between a profile and a node.

1.1.4 System management tools

WebSphere Application Server V8 provides a variety of administrative tools for configuring and managing your runtime environment. The choice of which combination of administrative tools you want to employ depends on the size and complexity of your runtime environment:

- ▶ The IBM WebSphere Customization Toolbox (WCT):

WebSphere Customization Toolbox includes tools for customizing various parts of your WebSphere application server environment. Tools include:

- The Web Server Plug-in Configuration Tool
- Profile Management Tool
- The z/OS Migration Tool

- ▶ Integrated Solutions Console, referred to as the *administrative console*:
 The administrative console is a browser-based client that uses a web application running in the web container to administer WebSphere Application Server.
 The administrative console uses SOAP through HTTP or HTTPS to connect to the administrative application in the application server or deployment manager. The administrative application connects to the administrative service, which is responsible for modifying the configuration repository using JMX managed beans (MBeans).
- ▶ WebSphere scripting client (**wsadmin**):
 The **wsadmin** client is a non-graphical scripting interface that can be used to administer WebSphere Application Server from a command-line prompt. It uses the Bean Scripting Framework (BSF), which supports a variety of scripting languages. It provides multiple options to connect to the Mbeans: SOAP, RMI, JSR160RMI, IPC, and NONE. Note that the RMI connection option is deprecated. The default connection type is SOAP.
 The NONE connection type connects directly to the configuration files, bypassing the application server. The NONE option allows the administrator to administer the application server even if the application server is not started, or if the embedded HTTP Server or administrative service is not responding. When using the NONE option, the administrator must be on the same system as the application server. There are some commands that require a running system (that is, \$AdminControl); these commands are not functional with connection type NONE.
- ▶ Task automation with Another Neat Tool (ANT):
 With ANT, you can create scripts that compile, package, install, and test your application on WebSphere Application Server.
- ▶ Administrative applications:
 You can develop custom Java applications that use the Java Management Extensions (JMX) based on the WebSphere Application Programming Interface (API).
- ▶ Command-line utilities:
 WebSphere Application Server provides administrative utilities to help manage your environment. These utilities are called from a command line. They can be used to perform common administrative tasks, such as starting and stopping WebSphere Application Server, backing up the configuration, and so on. Command-line utilities work on local processes (application servers, nodes, and deployment manager) only.
- ▶ High Performance Extensible Logging (HPEL):
 HPEL provides a convenient mechanism for storing and accessing log, trace, System.err, and System.out information produced by the application server or your applications. HPEL provides flexibility for administrators to manage logging resources while making it easier to work with log and trace content in addition to the basic logging and trace facility.

There are multiple levels of administration in WebSphere Application Server:

- ▶ In the WebSphere Application Server Express and Base offerings, you can administer stand-alone server instances individually.
- ▶ In the WebSphere Application Server Express and Base offerings, you can administer multiple stand-alone server instances on a single system using an administrative agent.
- ▶ In the WebSphere Application Server Network Deployment offering, you can administer an entire cell of application servers using a deployment manager.
- ▶ You can administer multiple stand-alone application servers, administrative agents, and deployment managers using a job manager.

1.2 System management in a stand-alone server environment

A stand-alone application server provides the necessary capabilities to run J2EE compliant applications. A stand-alone application server is a good starting point for development and test teams. It can also be used for proof of concept or light-weight applications that do not require intensive system resources.

To create a stand-alone application server, you must create an Application Server profile. The profile defines the application server, node, and cell.

The application server can be administered using the web-based administrative console, command-line utilities, and **wsadmin**. The administrative console and **wsadmin** provide remote administration access. All of the configuration data for an application server, including the installed applications, is stored in a configuration repository created when the profile is created.

Figure 1-3 shows the components of a stand-alone application server environment.

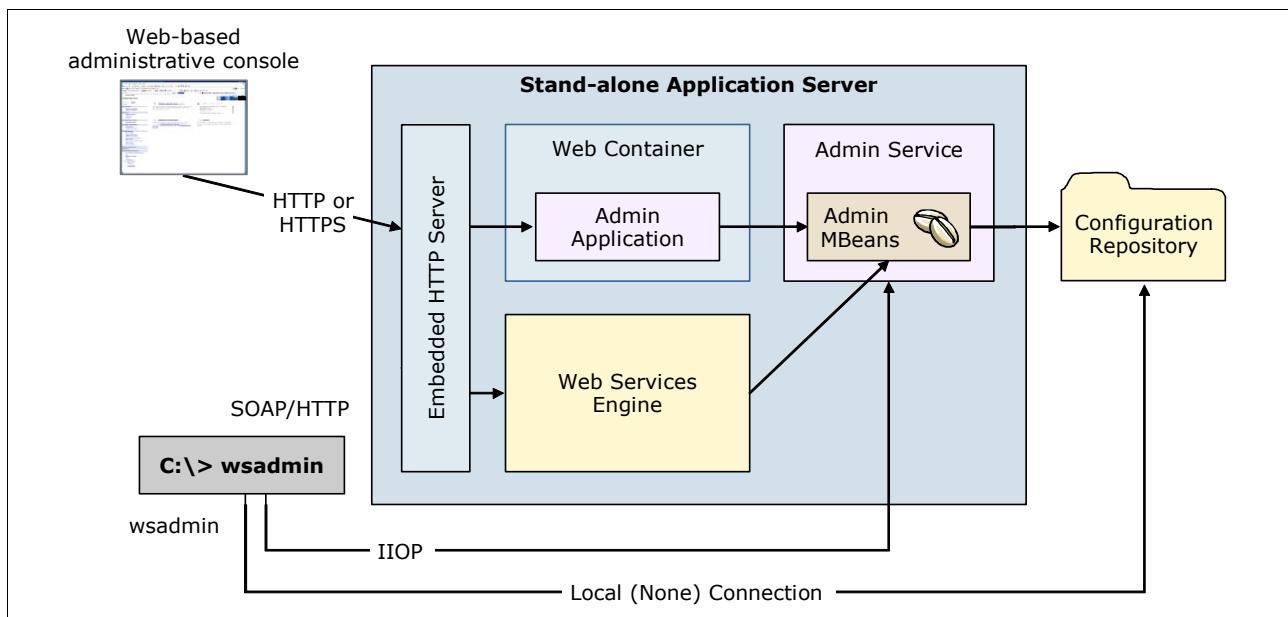


Figure 1-3 Stand-alone application server system management environment

1.3 System management of multiple stand-alone servers

Based on their business requirements, an organization can have multiple stand-alone application servers installed on the same system or on multiple systems. These servers might be used for development environments, unit testing, regression testing, staging environments, and so on.

When there are multiple stand-alone application servers installed, administration of each application server can become difficult. Each application server runs its own administrative service as well as the administrative console application. An administrator could have to juggle multiple consoles. And as the number of stand-alone servers increase on a given system, the system resources used for administrative functions increase.

An environment with independently managed application servers is shown in Figure 1-4.

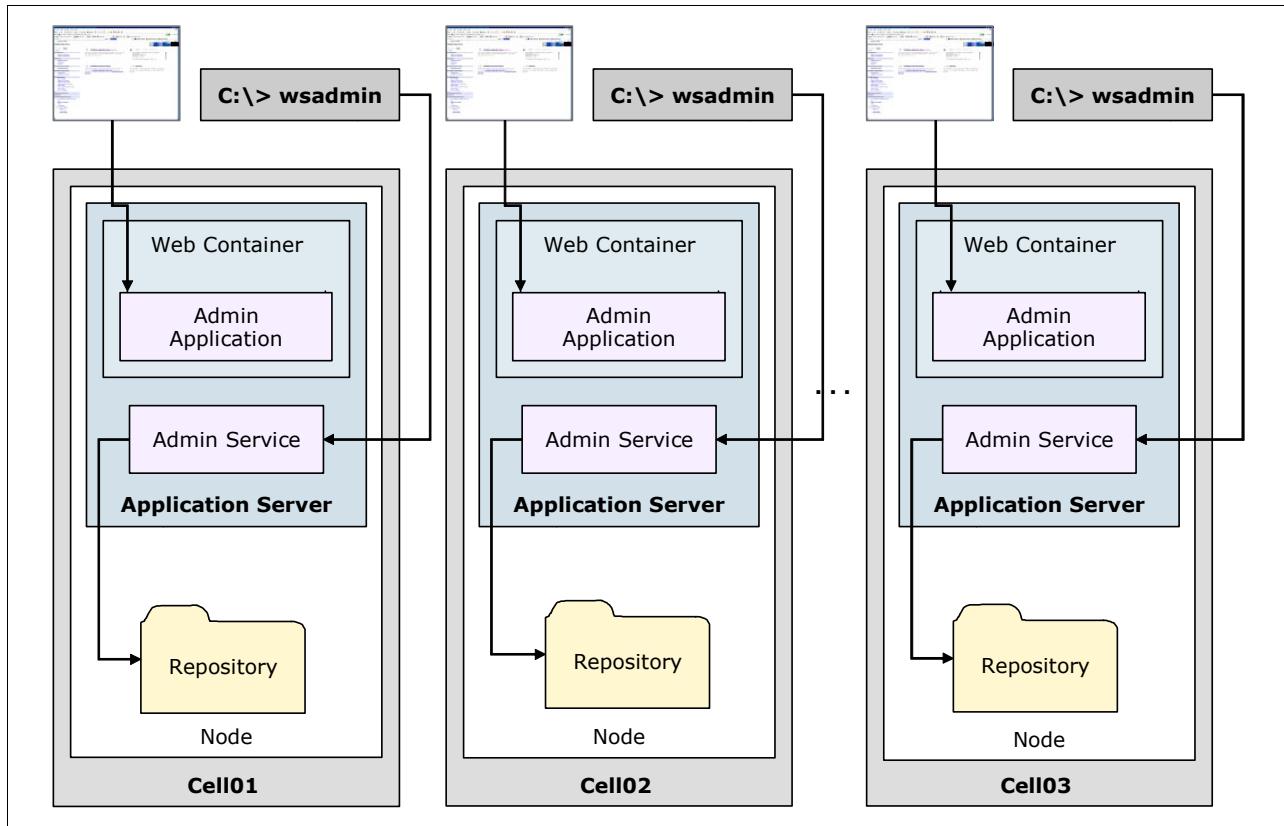


Figure 1-4 Multiple stand-alone servers with independent administration

With this infrastructure, the administrator must know which administrative port is used for each stand-alone server. Another drawback to managing stand-alone servers is that when the server is not running, the administrative application and administrative service are not available. Therefore, to start the application server, or make any changes, the application servers must be locally started.

The administrative agent reduces the problems related to managing multiple stand-alone servers by providing a centralized point of control for all of the stand-alone application servers within a system. The administrative agent is responsible for running the administrative application and administrative service to manage all of the servers on the registered node within the given system. This environment is shown in Figure 1-5.

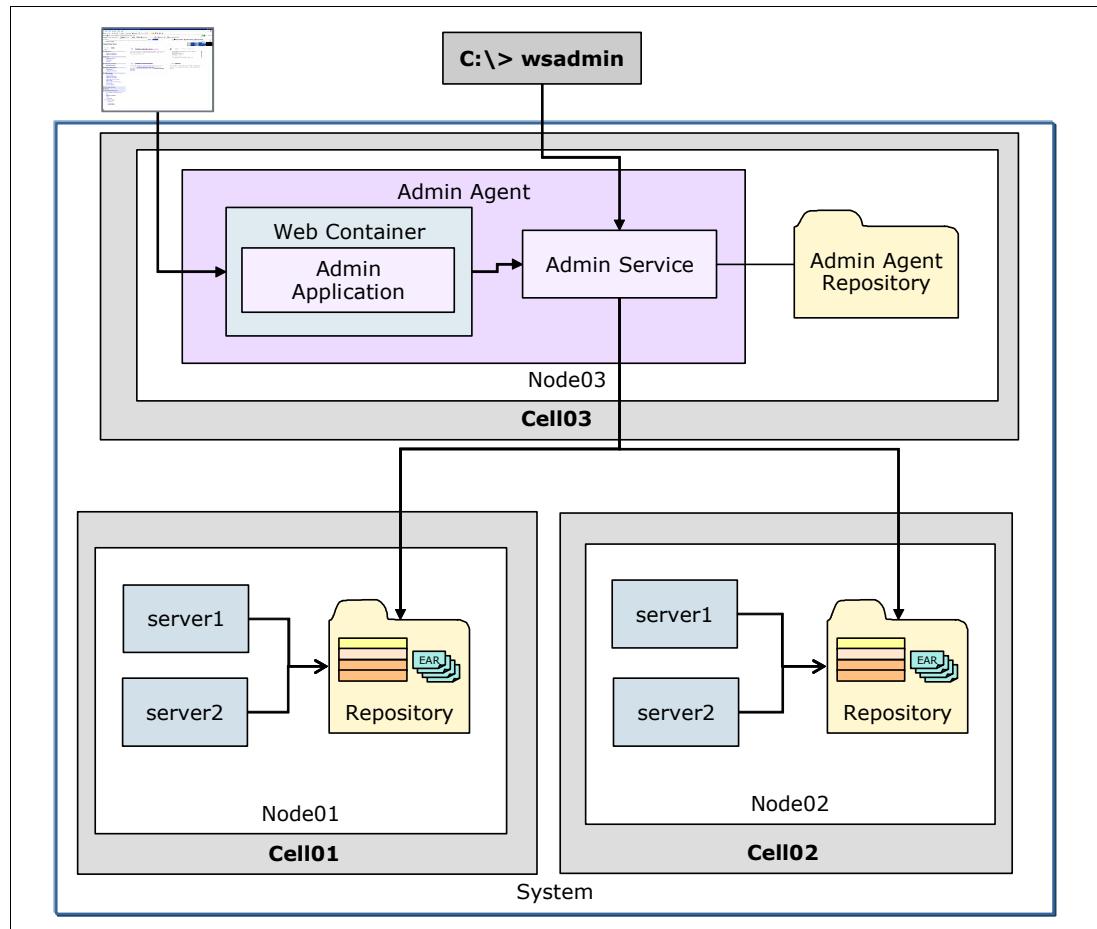


Figure 1-5 Managing multiple stand-alone servers with flexible management

After a node containing a stand-alone server is registered with the administrative agent, the administrative console application and administrative service are stopped on the application server. System resources are efficiently used because the administrative functionality is not enabled on multiple stand-alone servers. The stand-alone application server resources are dedicated to running applications.

Another key feature is the ability to administer an application server when it is not running. The administrative agent modifies the stand-alone servers configuration repository directly using the administrative service. The administrative agent can also start, stop, and create new servers within the managed node.

The administrative agent, along with multiple stand-alone servers, is a great starting point for simplifying administration. However, note that features such as failover, workload management, session data replication, and many other features cannot be configured without a distributed server environment.

1.4 System management in a distributed server environment

Although a stand-alone application server is sufficient to run any J2EE compliant application and other programming model applications, there are many runtime capabilities that are not addressed, such as what happens when an application server crashes, or needs to be restarted, or if the demand on the application exceeds the resources available on the hardware. A distributed server environment allows the developer to concentrate on solving the business problem at hand, while leaving infrastructure problems to WebSphere Application Server Network Deployment.

A distributed server environment provides the capabilities to create clusters of application servers. Clustering servers provides work load balancing, session data replication, and failover. The application servers are centrally managed using the deployment manager.

To build a distributed server environment, you start by creating a deployment manager profile. The deployment manager is responsible for administering the entire cell. The concept of cells and nodes become important in a distributed server environment. A deployment manager administers one and only one cell.

After the deployment manager is created, the next step is to create a custom profile, which creates a second cell (defaultCell), a node, and a node agent. At this point, you do not have a functioning application server environment. Figure 1-6 shows this temporary stage of the environment.

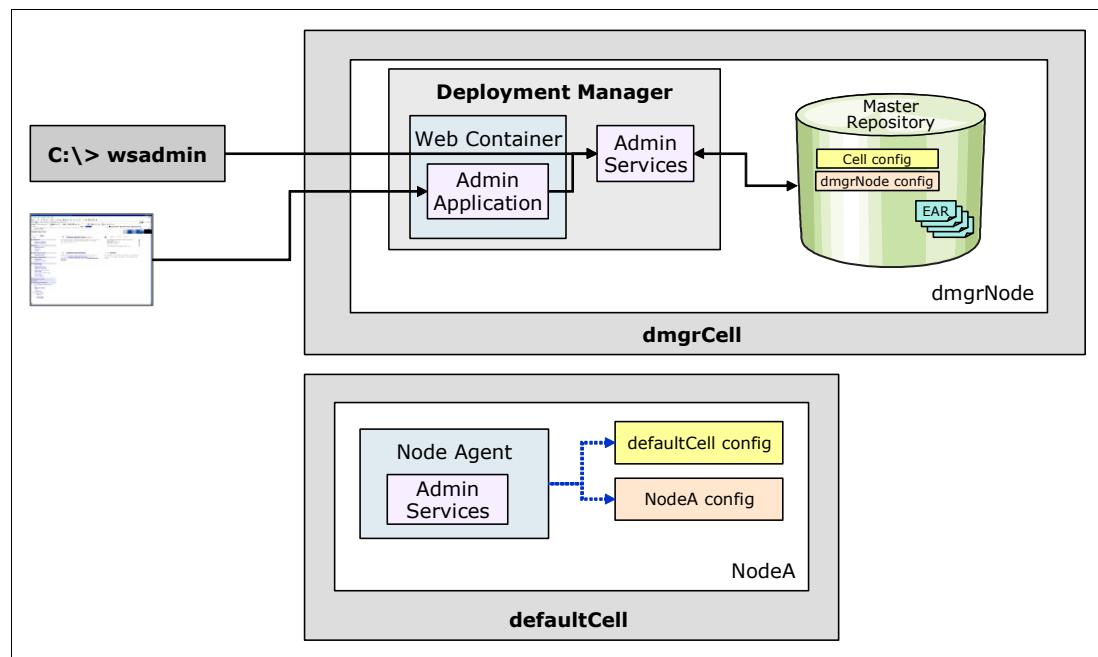


Figure 1-6 A deployment manager and unfederated custom profile

The next step is to federate the node (NodeA in Figure 1-6) to the deployment manager's cell by using the **addNode** command. After being federated, NodeA is no longer part of the defaultCell, but rather is part of the deployment manager's cell (dmgrCell).

Now that NodeA has been federated, all administration of NodeA is delegated to the deployment manager and new application servers can be created on the node using the administrative tools for the deployment manager.

This environment is reflected in Figure 1-7. Additional nodes can be added and servers created, to create a distributed server environment.

For design considerations, such as scalability, caching, high availability, load balancing, failover, disaster recovery security, and serviceability, refer to *IBM WebSphere Application Server V8 Concepts, Planning, and Design Guide*, SG24-7957.

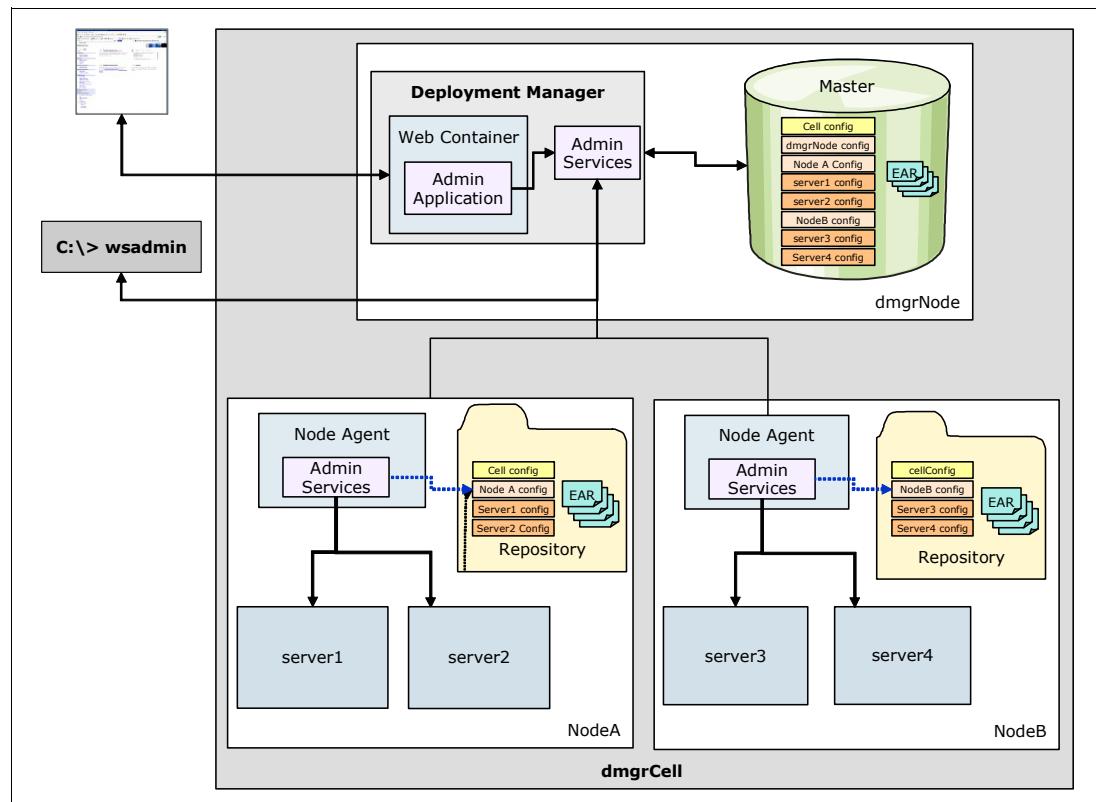


Figure 1-7 Distributed server environment

1.4.1 Centralized changes to configuration and application data

The deployment manager maintains a master repository of all the configuration files for nodes and servers in the cell. When configuration changes are made at the deployment manager, the changes are first stored in the master repository. Automatic or manual synchronization pushes the changes down to the affected nodes. Information about synchronization is given in 1.4.5, “File synchronization in distributed server environments” on page 21.

1.4.2 Rules for process startup

When a managed server begins its startup, it sends a discovery request message that allows other processes to discover its existence and establish communication channels with the process. This action makes it possible to start the processes in a distributed server environment without following a strict order for startup. For example:

- ▶ A node agent can be running while the deployment manager is not, and vice versa. When the stopped process is started, discovery occurs automatically.

- ▶ The deployment manager can be running while a managed server is not, and vice versa. The execution of a managed server is not dependent on the presence of a running deployment manager. The deployment manager is only required for permanent configuration changes written to the master repository.

The only rule to remember is that the node agent should be started before any application servers on that node. The node agent contains the Location Service Daemon (LSD) in which each application server registers on startup. However, the node agent is purely an administrative agent and is not involved in application serving functions. Each managed server has the data necessary to start itself.

Example discovery scenarios

In this section, we describe two typical situations that might occur:

- ▶ The node agent is not running and the deployment manager starts.
 - a. The deployment manager tries to determine if the node agent is running. The process fails.
 - b. When the node agent is started, it contacts the deployment manager, creates a communication channel, and synchronizes data.
- ▶ The node agent starts but no managed servers are started.
 - a. The node agent knows all about its managed servers and checks whether they are started. If so, it creates communication channels to these processes.
 - b. When a managed server starts, it checks whether the node agent is started and then creates a communication channel to it.

1.4.3 Distributed process discovery

Figure 1-8 shows an example of the distributed discovery process for a topology containing two nodes that are located on different machines. Note that both node agents in the figure use ports 7272 and 5000, which assumes that they reside on separate physical machines. If nodes are located on the same machine, they must be configured to use non-conflicting IP ports. The profile wizard selects non-conflicting ports for you automatically.

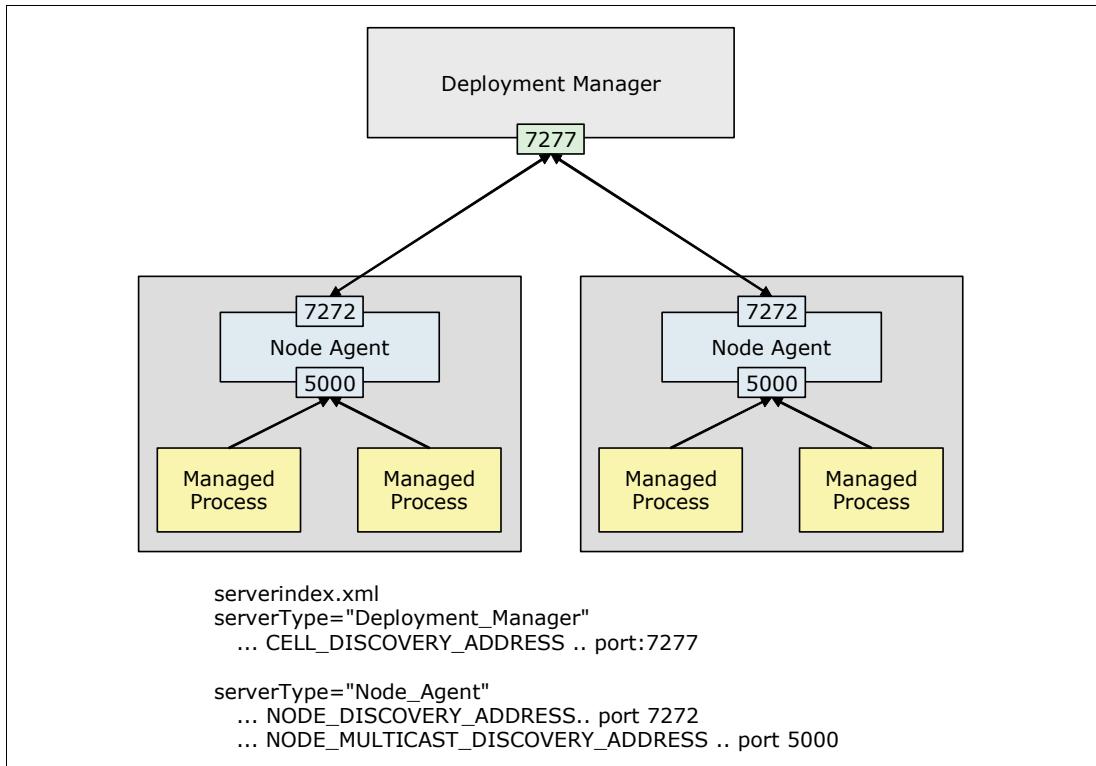


Figure 1-8 Distributed discovery process

Each node agent and deployment manager maintains status and configuration information by using discovery addresses, or *ports*. On startup, processes discover other running components, and create communication channels between them, through the discovery addresses.

During discovery, the following actions occur:

- ▶ The master repository located on the deployment manager installation contains the serverindex.xml file for each node. The deployment manager reads this file on startup to determine the host name and IP port of each node agent's NODE_DISCOVERY_ADDRESS.

The default port for the NODE_DISCOVERY_ADDRESS is 7272. You can verify this port by looking at the NODE_AGENT stanza in the serverindex.xml file of each node, located at:

`dmgr_profile_root/config/cells/cell_name/nodes/node_name/serverindex.xml`

You can also display this port from the administrative console by selecting **System Administration** → **Node agents**. Select each node agent and expand **Ports** under the Additional Properties section.

- ▶ The copy of the configuration repository located on each node contains the `serverindex.xml` file for the deployment manager. The node agent reads this file on startup to determine the host name and IP port of the deployment manager's `CELL_DISCOVERY_ADDRESS`.

The default port for the `CELL_DISCOVERY_ADDRESS` is port 7277. You can verify this port by looking at the `DEPLOYMENT_MANAGER` stanza in the `serverindex.xml` file for the deployment manager node, located at:

`profile_root/config/cells/cell_name/nodes/dmgr_node_name/serverindex.xml`

You can also display this port from the administrative console by selecting **System Administration** → **Deployment manager**. Expand **Ports** under the Additional Properties section.

- ▶ The copy of the configuration repository located on each node also contains the `serverindex.xml` file for the node. Each managed server reads this file on startup to determine the host name and IP port of the node agent's `NODE_MULTICAST_DISCOVERY_ADDRESS`.

A multicast address is used to prevent the usage of a large number of IP ports for managed server to node agent discovery requests. Using multicast, a node agent can listen on a single IP port for any number of local servers.

The default port for the `NODE_MULTICAST_DISCOVERY_ADDRESS` is 5000. You can verify this port by looking at the `NODE_AGENT` stanza in the `serverindex.xml` file of the node, located at:

`profile_root/config/cells/cell_name/nodes/node_name/serverindex.xml`

You can also display this port from the administrative console by selecting **System Administration** → **Node agents**. Select the node agent and expand **Ports** in the Additional Properties section.

Important: Keep the following considerations in mind:

- ▶ The discovery service uses the `InetAddress.getLocalHost()` call to retrieve the IP address for the local machine's host name. The network configuration of each machine must be configured so that `getLocalHost()` does not return the loopback address (127.0.0.1). It must return the real IP address of the correctly chosen Network Interface Card (NIC).
- ▶ A multicast address is a logical address. Therefore, it is not bound to a real, physical network interface, and is not the same as the host name (or IP address) of the host on which the node agent is executed.

Multicast host addresses must be within a special range (224.0.0.0 to 239.255.255.255) defined by the IP standards and must never be a host name value. The default for WebSphere Application Server node agents is 232.133.104.73.

Each server has its own copy of the configuration and application data necessary for startup of the run time and the installed applications.

1.4.4 Configuration and application data repository

The configuration and application data repository is a collection of files containing all the information necessary to configure and execute servers and their applications. Configuration files are stored in XML format, while application data is stored as Enterprise ARchive (EAR) files and deployment descriptors.

Repository directory structure

It is important to know that configuration files defining a runtime environment are stored in profile directories. Each node containing a deployment manager, application server, administrative agent, or job manager has its own profile directory under the `install_root/profiles` directory.

Terminology: In the remainder of this book, when we talk about a specific profile directory, located at `install_root/profiles/profile_name`, we refer to it as the `profile_root` directory.

When we are speaking of a profile directory for a specific profile, we use the following terms:

- ▶ Deployment manager profile: `dmgr_profile_root`
- ▶ Administrative agent profile: `adminAgnt_profile_root`
- ▶ Job manager profile: `jmgr_profile_root`

The repository files are arranged in a set of cascading directories under each profile directory structure, with each directory containing a number of files relating to different components of the cell, as shown in Figure 1-9. The repository structure follows the same format, regardless of whether you have a stand-alone server environment or distributed server environment.

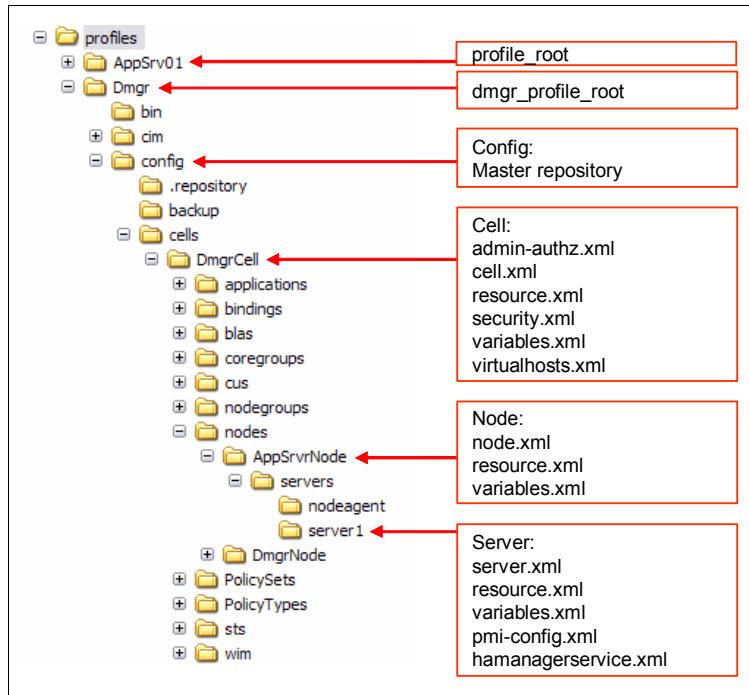


Figure 1-9 Repository directory structure

The `profile_root/config` directory is the root of the repository for each profile. It contains the following directory structure:

- ▶ `cells/cell_name/`

This is the root level of the configuration for the cell. The directory contains a number of cell-level configuration files. Depending on the types of resources that have been configured, you might see the following subdirectories:

- `cells/cell_name/applications/` contains one subdirectory for every application that has been deployed within the cell.
- `cells/cell_name/buses/` contains one directory for each service integration bus (SIBus) defined.
- `cells/cell_name/coregroups/` contains one directory for each core group defined.
- `cells/cell_name/nodegroups/` contains one directory for each node group defined.
- `cells/cell_name/nodes/` contains the configuration settings for all nodes and servers managed as part of this cell. The directory contains one directory per node. Each `cells/cell_name/nodes/<node>` directory contains node-specific configuration files and a server directory that in turn contains one directory per server and node agent on that node.
- `cells/cell_name/clusters/` contains one directory for each of the clusters managed as part of this cell. Each cluster directory contains a single file, `cluster.xml`, which defines the application servers of one or more nodes that are members of the cluster.

The overall structure of the master repository is the same for both a stand-alone server environment and a distributed server environment. The differences are summarized in the following sections.

In a stand-alone server environment, the structure has the following characteristics:

- ▶ The master repository is held on a single machine. There are no copies of this specific repository on any other node.
- ▶ The repository contains a single cell and node.
- ▶ There is no node agent because each application server is stand-alone, so there is no directory for the node agent (nodeagent).
- ▶ Clusters are not supported. Therefore, the repository tree does not contain the clusters directory or subdirectories.

In a distributed server environment, the structure has the following characteristics:

- ▶ The master repository is held on the node containing the deployment manager. It contains the master copies of the configuration and application data files for all nodes and servers in the cell.
- ▶ Each node also has a local copy of the configuration and application data files from the master repository that are relevant to the node.
- ▶ When changes are made to the configuration in the master repository, those changes need to be synchronized to the configuration files on the nodes.
- ▶ Changes can be made to the configuration files on a node, but the changes are temporary and are overwritten by the next file synchronization from the deployment manager. Permanent changes to the configuration require changes to the file or files in the master repository. Configuration changes made to node repositories are not propagated up to the cell.

- ▶ The applications directory of the master repository contains the application data (binaries and deployment descriptors) for all applications deployed in the cell. The local copy of the applications directory on each node only contains the directories and files for the applications deployed on application servers within that node.

Information about the individual files in each of these directories can be found in the topic, *Configuration Document Descriptions*, at the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.nd.multiplatform.doc/info/ae/ae/rcfg_rconfdoc_descriptions.html

Variable scoped files

Identically named files that exist at differing levels of the configuration hierarchy are termed *variable scoped* files. There are two uses for variable scoped files:

- ▶ Configuration data contained in a document at one level is logically combined with data from documents at other levels of the configuration hierarchy. In the case of conflicting definitions, the “most specific” value takes precedence. For example, if an identical entry exists in the files at the cell and node level (as with a variable defined in both the cell and node’s variables.xml file), the entry at the node level takes precedence.
- ▶ Documents representing data is not merged but is rather scoped to a specific level of the topology. For example, the namestore.xml document at the cell level contains the cell persistent portion of the name space, while the namestore.xml at the node level contains the node persistent root of the name space.

Application data files

The master repository is also used to store the application binaries and deployment descriptors. The *profile_root/config* directory of the master repository contains the following directory structure used to hold application binaries and deployment settings:

- ▶ *cells/cell_name/applications/*

This directory contains a subdirectory for each application deployed in the cell. Names of the directories match the names of the deployed applications.

Note: The name of the deployed application does not have to match the name of the original EAR file used to install it. Any name can be chosen when deploying a new application, as long as the name is unique across all applications in the cell.

- ▶ *cells/cell_name/applications/app_name.ear*

Each application’s directory in the master repository contains:

- A copy of the original EAR, called *app_name.ear*, which does not contain any of the bindings specified during the installation of the application.
- A deployments directory, which contains a single *app_name* directory used to contain the deployed application configuration.

- ▶ *cells/cell_name/applications/app_name.ear/deployments/app_name*

The deployment descriptors in this directory contain the bindings specified during application deployment. The deployment directory of each application contains these files:

- *deployment.xml*

This file contains configuration data for the application deployment, including the allocation of application modules to application servers, and the module startup order.

- META-INF/

This directory contains these files:

- application.xml: J2EE standard application deployment descriptor
- ibm-application-bnd.xmi: IBM WebSphere-specific application bindings
- ibm-application-ext.xmi: IBM WebSphere-specific application extensions
- was.policy: Application-specific Java 2 security configuration

This file is optional. If not present, then the policy files defined at the node level apply for the application.

The subdirectories for all application modules (WARs and EJB JARs) are contained in was.policy, along with each module's deployment descriptors. The subdirectories for each module do not contain application binaries (JARs, classes, and JSPs), only deployment descriptors and other configuration files.

The installation of an application onto a WebSphere Application Server application server results in:

- ▶ The storage of the application binaries and deployment descriptors within the master repository.
- ▶ The publishing of the application binaries and deployment descriptors to each node that will be hosting the application. These files are stored in the local copy of the repository on each node.

Each node then installs applications ready for execution by exploding the EARs under *profile_root*/installedApps/*cell_name*/ as follows:

- ▶ *profile_root*/installedApps/*cell_name*/

This directory contains a subdirectory for each application deployed to the local node. The name of each application's directory reflects the name under which the application is installed, not the name of the original EAR. For example, if an application is called myapp, then the *installedApps*/*cell_name* directory will contain a *myapp.ear* subdirectory.

- ▶ *profile_root*/installedApps/*cell_name*/*app_name.ear*/

Each application-specific directory contains the contents of the original EAR used to install the application.

- The deployment descriptors from the original EAR. These descriptors do not contain any of the bindings specified during application deployment.
- All application binaries (JARs, classes, and JSPs)

Figure 1-10 summarizes how the node's local copy of the repository contains the application's installed deployment descriptors, while the directory under `installedApps` contains the application binaries.

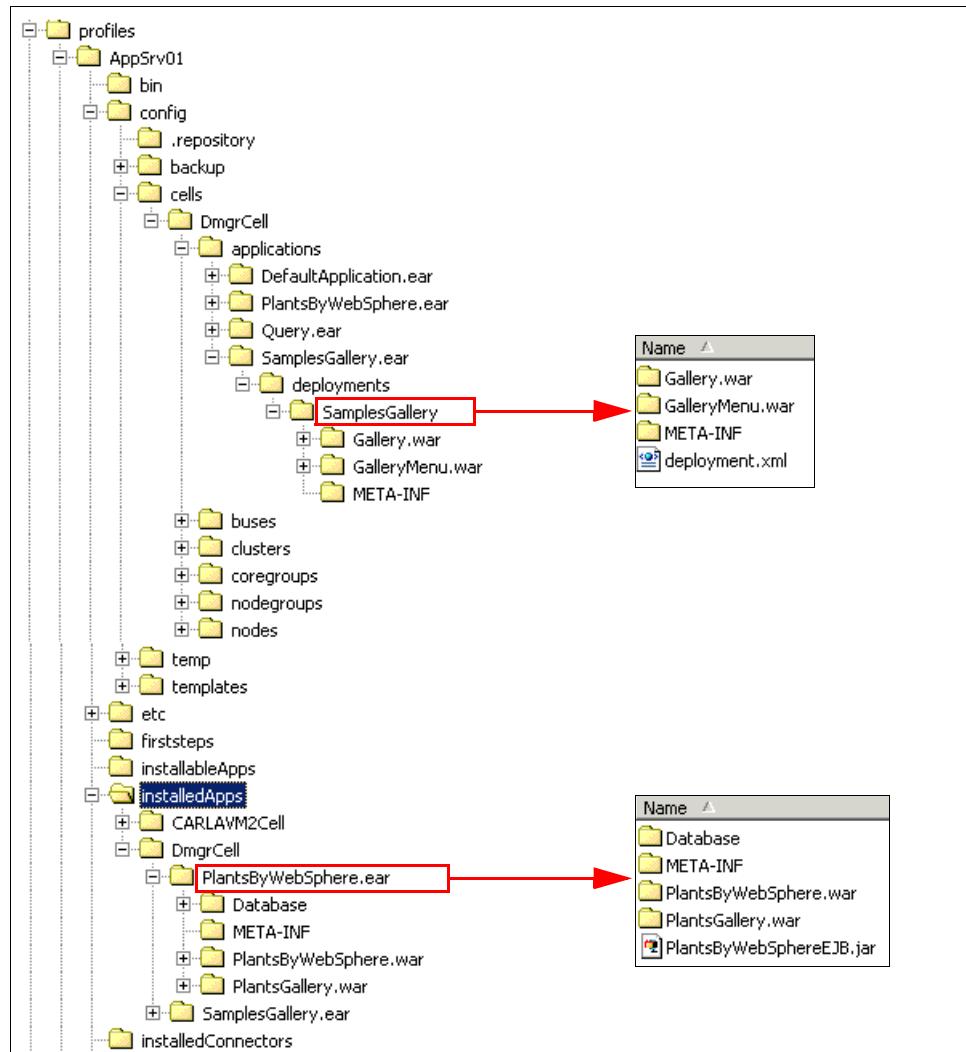


Figure 1-10 Location of application data files

By default, a WebSphere Application Server application server executes an application by completing the following tasks:

1. Loading the application binaries stored under:

`profile_root/installedApps/cell_name/app_name.ear/`

You can change this location by altering the `Application binaries` setting for the enterprise application or by altering the `$(APP_INSTALL_ROOT)` variable setting.

2. Configuring the application using the deployment descriptors stored under:

`profile_root/config/cells/cell_name/applications/app_name.ear/deployments/app_name`

You can change this setting by modifying the `Use configuration information in binary` setting for the enterprise application. This is the `Use Binary Configuration` field on the application installation and update wizards.

By default, the setting is not enabled. Enabling it specifies that you want the application server to use the binding, extensions, and deployment descriptors located in the application EAR file rather than those stored in the deployments directory.

1.4.5 File synchronization in distributed server environments

The file synchronization service is the administrative service responsible for keeping the configuration and application data files that are distributed across the cell up to date. The service runs in the deployment manager and node agents, and ensures that changes made to the master repository are propagated out to the nodes, as necessary. The file transfer system application is used for the synchronization process. File synchronization can be forced from an administration client, or can be scheduled to happen automatically.

During the synchronization operation, the node agent checks with the deployment manager to see if any files that apply to the node have been updated in the master repository. New or updated files are sent to the node, while any deleted files are also deleted from the node.

Synchronization is one-way. The changes are sent from the deployment manager to the node agent. No changes are sent from the node agent back to the deployment manager.

Synchronization scheduling

The scheduling of file synchronization is configured using an administrative client. The available options are:

- ▶ Automatic synchronization:

Synchronization can be made to operate automatically by configuring the file synchronization service of the node agent. These settings allow you to:

- Enable periodic synchronization to occur at a specified time interval.
By default, this option is enabled with a time interval of one minute.
- Enable synchronization at server startup.

The synchronization occurs before the node agent starts a server. Note that if you start a server using the `startServer` command, this setting has no effect.

- ▶ Explicit/forced synchronization:

Synchronization can be explicitly forced at any time using an administrative client.

Tip: In a production environment, the automatic synchronization interval should be increased from the one minute default so that processing and network impact is reduced.

How files are identified for synchronization

Deep dive: This section provides in-depth knowledge that can be useful when debugging or testing, but it is not necessary when trying to understand the overall architecture.

When synchronization occurs, WebSphere Application Server must be able to identify the files that have changed and therefore need to be synchronized. To do this, WebSphere Application Server uses the following scheme:

- ▶ A calculated digest is kept by both the node agent and the deployment manager for each file in the configuration they manage. These digest values are stored in memory. If the digest for a file is recalculated and it does not match the digest stored in memory, this indicates the file has changed.

- ▶ An *epoch* for each folder in the repository and one for the overall repository is also stored in memory. These epochs are used to determine whether any files in the directory have changed. When a configuration file is altered through one of the WebSphere Application Server administration interfaces, then the overall repository epoch and the epoch for the folder in which that file resides is modified.

Manually updating a configuration file does not cause the digest to change. Only files updated with administration clients are marked as changed. Manually updating the files is not recommended, but if you do, a forced synchronization will include manually updated files.

- ▶ During configuration synchronization operations, if the repository epoch has changed since the previous synchronize operation, then individual folder epochs are compared. If the epochs for corresponding node and cell directories do not match, then the digests for all files in the directory are recalculated, including that changed file.

Ensuring that manual changes are synchronized

Manually changing configuration files is not recommended. This action should only be done as a diagnostic measure or on the rare occasion that you need to modify a configuration setting that is not exposed by the administration clients.

The following topic lists several configuration files that have settings not exposed in the administration clients. In the event that you find it necessary to edit a file manually, this information can help make sure that you do not lose your changes. Refer to *Configuration Document Descriptions* in the Information Center at the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.nd.multiplatform.doc/info/ae/ae/rcfg_rconfdoc_descriptions.html

Manual editing has several drawbacks, including:

- ▶ When using **wsadmin** and the administrative console, you have the benefit of a validation process before the changes are applied. With manual editing, you have no such fail-safe.
- ▶ Updates made manually are not marked for synchronization and will be lost at the next synchronization process unless you make them in the master repository and manually force synchronization.

If a change to a configuration file is made by editing the file, then the digest for the file is not recalculated, because epochs for the directories continue to match and the synchronization process does not recognize that the files have changed.

However, manual edits of configuration files in the master cell repository can be picked up if the repository is reset so that it re-reads all the files and recalculates all of the digests. You can reset either the master cell repository epoch or the node repository epoch:

- ▶ Resetting the master cell repository causes any manual changes made in the master configuration repository to be replicated to the nodes where the file is applicable.
- ▶ Resetting the node repository causes any manual changes to the local node files to be overwritten by whatever is in the master cell repository, regardless of whether the cell repository was changed or not. Any manual changes in the master repository will be picked up and brought down to the node.

The main difference between cell reset and node reset is that cell reset is likely to impact the entire cell, not just one node. This situation holds true for changes to installed applications as well. They are treated the same as other configuration files in the repository. For each installed application, there is an EAR file in the repository and configuration files associated with the deployment of the application.

If you manually change the EAR file and reset the master cell repository, the changed EAR file is replicated out to the nodes where it is configured to be served and is expanded in the appropriate location on that node for the application server to find it. The application on that node is stopped and restarted automatically so that whatever is changed is picked up and made available in the application server.

Important: Manually changing the EAR file is best performed by advanced users. Otherwise, unpredictable results can occur.

If you manually edit one of the deployment configuration files for the application and reset the repository, that change is replicated to the applicable nodes and is picked up the next time the application on that node is restarted.

Resetting the master cell repository

Note: The use of `wsadmin` is covered in Chapter 8, “Administration with scripting” on page 343. The only thing that you might need to know about `wsadmin` to complete these tasks is to start `wsadmin` on the SOAP connector port of the process on which you want to run the commands. The default is to start to port 8879. If the process you are connecting to has a different port number specified, start `wsadmin` with the `-port` argument.

To perform a reset of the master cell repository, complete the following steps:

1. Open a command prompt, change to the `dmgr_profile_root/bin` directory, and start a `wsadmin` session. Note that the deployment manager must be running. Use the following command:

```
cd dmgr_profile_root\bin  
wsadmin
```

2. Enter the following statements:

```
wsadmin>set config [$AdminControl queryNames  
*:*,type=ConfigRepository,process=dmgr]
```

```
wsadmin>$AdminControl invoke $config refreshRepositoryEpoch
```

You can see a number returned by the `refreshRepositoryEpoch` operation, for example, 1047961605195, as shown in Example 1-1.

Example 1-1 Resetting the master cell repository

```
dmgr_profile_root\bin>wsadmin  
WASX7209I: Connected to process "dmgr" on node DmgrNode using SOAP connector; The  
type of process is: DeploymentManager  
WASX7029I: For help, enter: "$Help help"
```

```
wsadmin>set config [$AdminControl queryNames  
*:*,type=ConfigRepository,process=dmgr]
```

```
WebSphere:name=repository,process=dmgr,platform=common,node=DmgrNode,version=8.0,t  
ype=ConfigRepository,mbeanIdentifier=repository,cell=DmgrCell,spec=1.0
```

```
wsadmin>$AdminControl invoke $config refreshRepositoryEpoch  
1237317922687  
wsadmin>
```

This procedure resets the entire cell repository digest set. On the next synchronize operation, all files in the master cell repository have their digests recalculated. Any manual changes are replicated to the applicable nodes.

Resetting the node repository

There are multiple ways to reset a node repository for synchronization:

- ▶ In a **wsadmin** session connected to the deployment manager or node agent, enter the following:

```
wsadmin>set config [$AdminControl queryNames  
*:*,type=ConfigRepository,process=nodeagent]
```

```
wsadmin>$AdminControl invoke $config refreshRepositoryEpoch
```

This action resets the node digest set. Any file that does not match what is in the repository is overwritten.

Example 1-2 gives an overview of resetting the node repository.

Example 1-2 Resetting the node repository

```
profile_root\bin>wsadmin -port 8883
```

```
WASX7209I: Connected to process "nodeagent" on node AppSrvrNode using SOAP  
connector; The type of process is: NodeAgent  
WASX7029I: For help, enter: "$Help help"
```

```
wsadmin>set config [$AdminControl queryNames  
*:*,type=ConfigRepository,process=nodeagent]  
WebSphere:name=repository,process=nodeagent,platform=common,node=AppSrvrNode,versi  
on=5.0,type=ConfigRepository,mbeanIdentifier=repository,cell=DmgrCell
```

```
wsadmin>$AdminControl invoke $config refreshRepositoryEpoch  
1237319314359
```

- ▶ From the deployment manager administrative console, select **System Administration → Nodes** to see a list of the nodes in the cell. Notice the Synchronize and Full Resynchronize buttons on the page. The Synchronize button causes a normal synchronize operation with no re-reading of the files. The Full Resynchronize button is the reset and recalculate function. Select the node or nodes to be updated with manual changes, and then click the **Full Resynchronize** button.
- ▶ Use the **syncNode** command. This command is a stand-alone program that runs separately from the node agent. It has no cache of epoch values that could be used for an optimized synchronization, therefore performing a complete synchronization. For this same reason, if you restart a node agent, the first synchronization that it performs will always be a complete synchronization. Note that this action requires the node agent to be stopped.

The **syncNode** command resides in the bin directory of the base install. Use the following command:

```
cd profile_root\bin  
syncNode cell_host
```

Example 1-3 shows the use of the **syncNode** command.

Example 1-3 Using the syncNode command

```
profile_root\bin>stopnode  
ADMU0116I: Tool information is being logged in file
```

```
profile_root\logs\nodeagent\stopServer.log
ADMU0128I: Starting tool with the AppSrv01 profile
ADMU3100I: Reading configuration for server: nodeagent
ADMU3201I: Server stop request issued. Waiting for stop status.
ADMU4000I: Server nodeagent stop completed.

profile_root\bin>syncnode sysvm2
ADMU0116I: Tool information is being logged in file
profile_root\logs\syncNode.log
ADMU0128I: Starting tool with the AppSrv01 profile
ADMU0401I: Begin syncNode operation for node AppSrvrNode with Deployment Manager
sysvm2: 8879
ADMU0016I: Synchronizing configuration between node and cell.
ADMU0402I: The configuration for node AppSrvrNode has been synchronized with
Deployment Manager sysvm2: 8879
```

The repository is flexible in that there is no predefined list of document types that it permits. You can add any file you want. Perhaps you have some unique configuration data that needs to be used on all nodes. You could put it in the config/cells/*cell_name* folder and it would be synchronized to all nodes. If it applies to just one node, you could put it in the folder corresponding to that node and it would be synchronized only to that node. The same applies for any additional documents in a server level folder.

For example, under normal circumstances, all application files are packaged in the EAR file for the application. However, consider a configuration file specific to an application. Any changes to that file would require that you update the EAR file and synchronize the entire application.

One possibility is to put a properties file in the application deployment directory in the master configuration repository, so that it is replicated automatically to all nodes where the application is installed but the entire EAR is not replicated. Then you could have an ExtensionMBean update the properties file in the master repository and normal synchronization would replicate just those changes out to the nodes without the need to synchronize the whole EAR and restart the application.

1.5 Management of distributed and stand-alone servers

It is possible to encounter a scenario where there might be multiple distributed environments, each managed by their own deployment manager. With multiple deployment managers, they must be administered individually and there is no way of coordinating management actions between the different distributed environments without the job manager. Distributed environment administration performance would be affected by low latency networks because file synchronization between the deployment manager and node agent are dependent on network communication.

The job manager can be used to administer multiple distributed environments as well as stand-alone servers. The job manager administers the environment asynchronously using the concept of jobs. Because jobs are submitted asynchronously, a low latency network is sufficient, which can be useful when the environment is distributed over distant geographical areas.

The job manager is available only with WebSphere Application Server Network Deployment and WebSphere Application Server for z/OS.

To administer a distributed environment, the deployment manager is registered with the job manager. To administer stand-alone servers, the nodes managed by the administrative agent are registered with the job manager. This relation between the job manager and the environments it can interact with is shown in Figure 1-11.

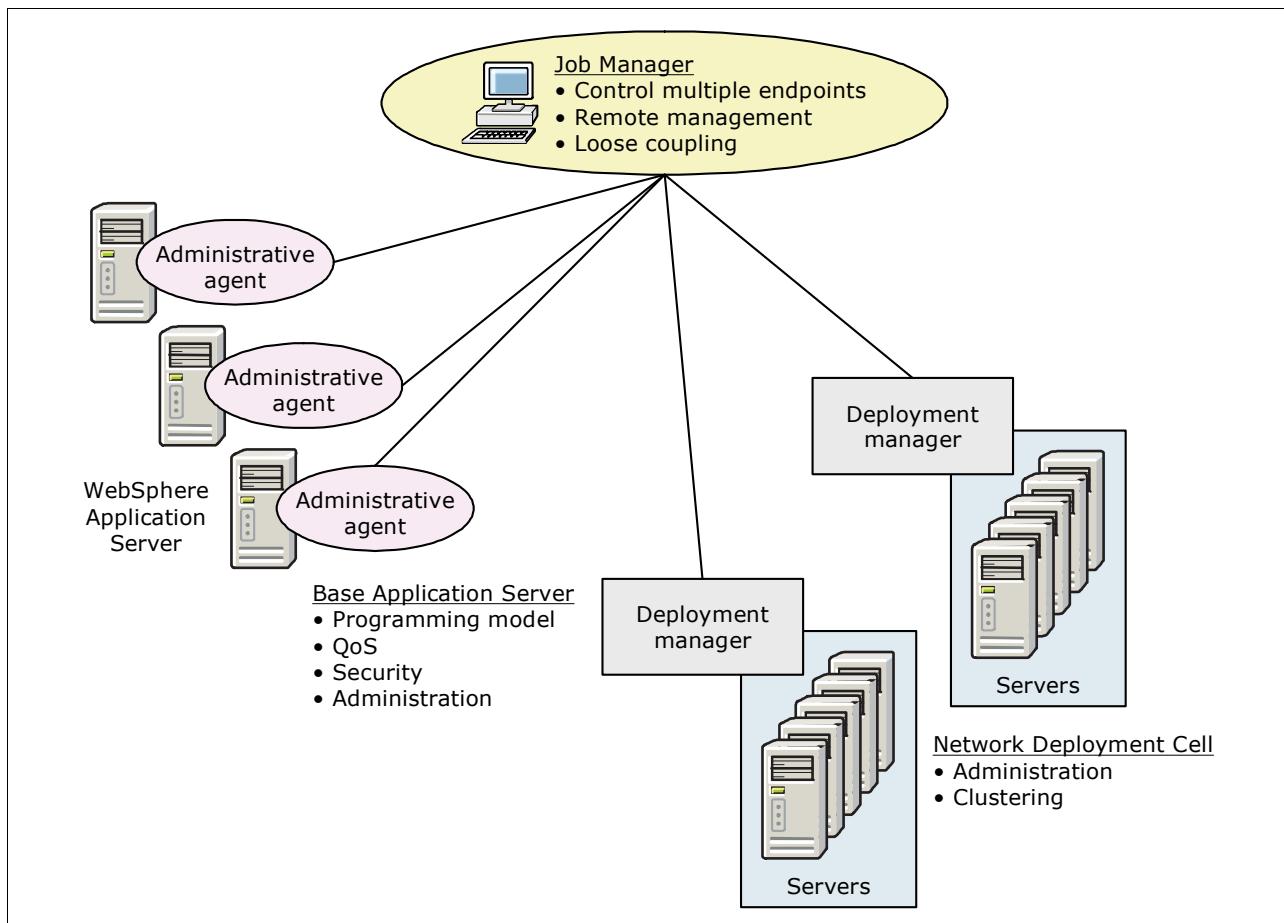


Figure 1-11 Flexible management

The job manager administers the registered environments by submitting jobs that perform tasks, for example:

- ▶ Start and stop servers.
- ▶ Create and delete servers
- ▶ Install and uninstall applications.
- ▶ Start and stop applications.
- ▶ Run `wsadmin` scripts.
- ▶ Distribute files.

The job manager has a repository for its own configuration files, which are related to security, administration of job manager, configurations, and so on. However, it does not maintain a master repository the way a deployment manager does. Rather, the job manager allows the administrative agents and deployment managers to continue managing their environments as they would have had they not been registered with the job manager. The job manager simply provides another point of administration, as shown in Figure 1-12.

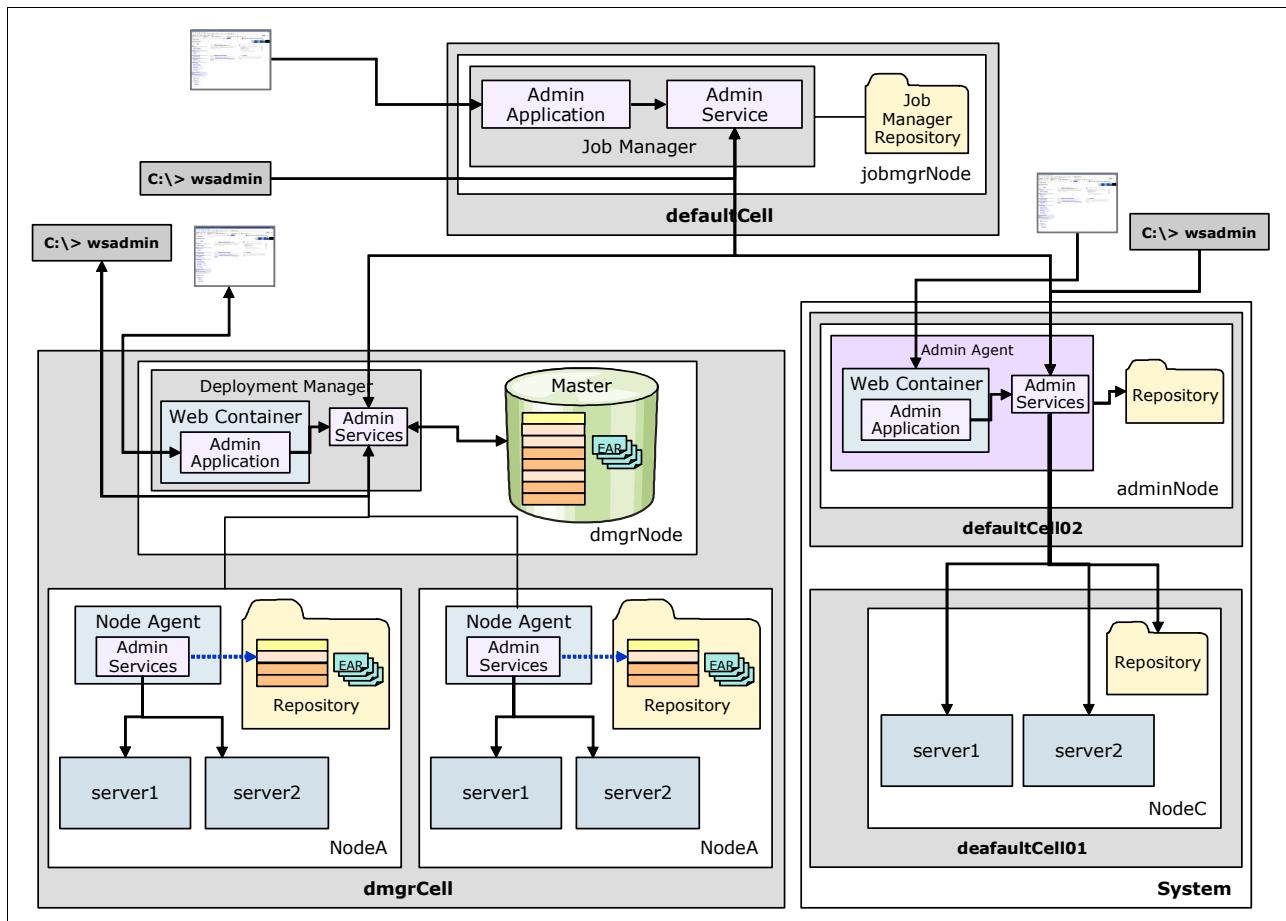


Figure 1-12 Job manager administration environment

The job manager can administer multiple administrative agents and deployment managers. Conversely, each administrative agent and deployment manager can be registered with multiple job managers.

1.6 Centralized installation manager

In WebSphere Application Server V8, the centralized installation manager has new capabilities and is now integrated with IBM Installation Manager and the job manager. In this section, we provide information about changes to the centralized installation manager use case model between Version 7 and Version 8.

As shown in Figure 1-13, in WebSphere Application Server V7, the centralized installation manager component resides inside the deployment manager and has access to targets that are limited only to the cell.

There is a single, file-based repository that holds the binaries that are needed to install WebSphere Application Server, maintenance packages, and the Update Installer. You can create this repository when installing the deployment manager or later using the Installation Factory. The centralized installation manager, with all the required data to manage external WebSphere Application Server instances, uses its administrative commands to push binary data to the target hosts. The centralized installation manager can install a new product on a target and can also create a new WebSphere Application Server custom profile (node agent).

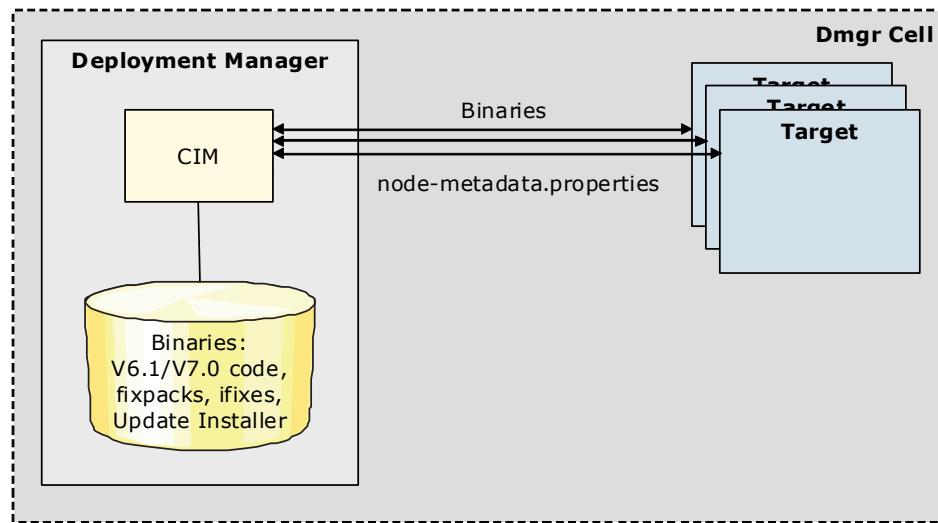


Figure 1-13 Centralized installation manager model introduced in WebSphere Application Server V7

The centralized installation manager relies on remote node status by synchronizing its information in the `node-metadata.properties` file that is kept locally on the deployment manager.

Figure 1-14 illustrates the enhanced centralized installation manager use case model available in WebSphere Application Server V8.

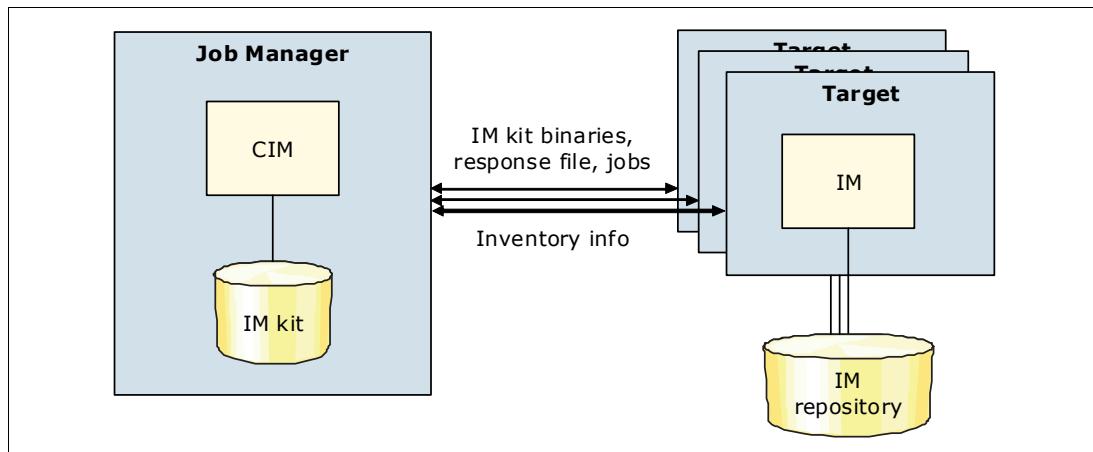


Figure 1-14 The centralized installation manager concept introduced with WebSphere Application V8

With WebSphere Application Server V8, you can use the centralized installation manager from the deployment manager or from the job manager, which removes the cell limitation. This design allows you to create independent topologies of WebSphere Application Server installations using a single point of administration.

This version of the centralized installation manager also changes the repository layout. With this version, there is one local repository for the deployment manager or job manager that holds only Installation Manager binaries (Installation Manager installation kit), but that does not require you to configure any additional software. In previous versions, you needed the Installation Factory to set up the local repository.

All other software, such as fix packs or the WebSphere Application Server product, is kept inside the Installation Manager repository, which can be accessed through the Internet or from a specified server in the intranet. You can also use a local repository for each Installation Manager instance on targets, but this method requires more configuration and relevant disk space volume.

The centralized installation manager in WebSphere Application Server V8 also requires that you push only the Installation Manager binaries to the remote target. After the Installation Manager is installed, the centralized installation manager instructs Installation Manager on the target to download only the specified software from its repository and to install that software on the target.

Instructions for Installation Manager are written in a standard response file and are sent using jobs. As a result, the job manager or deployment manager receives the information about the target status and uses the response file for further administration purposes.

WebSphere Application Server V8 supports both of the models illustrated in Figure 1-13 on page 28 and Figure 1-14 on page 28. Furthermore, you can use the central installation manager with a mixed WebSphere Application Server environment that consists of Version 6.1, Version 7, and Version 8.

1.7 Java Management Extensions (JMX)

Deep dive: Extensive knowledge of JMX is not required to administer WebSphere Application Server. However, familiarity with some basic concepts, such as MBeans, can be useful when you are writing scripts for `wsadmin`.

The system management functionality of WebSphere Application Server is based on the use of Java Management Extensions (JMX). JMX is a framework that provides a standard way of exposing Java resources, for example, application servers, to a system management infrastructure. The JMX framework allows a provider to implement functions, such as listing the configuration settings, and allows users to edit the settings. It also includes a notification layer that can be used by management applications to monitor events, such as the startup of an application server. The use of JMX opens the door to third-party management tool providers. Users of WebSphere Application Server are no longer restricted to IBM-supplied management tools.

JMX is a Java specification (JSR-003) that is part of J2SE 1.5. A separate specification defines the J2EE management API (JSR-77) for managing a J2EE conforming application server. WebSphere Application Server provides *managed objects (MOs)* as defined in the JSR-77 specification and hence is manageable from third-party management products that deliver J2EE management capabilities.

IBM WebSphere Application Server V6, V6.1, V7, and V8 implements JMX 1.2, while Version 5.x implements JMX 1.1. Due to the evolution of the JMX specification, the serialization format for JMX objects differs between the two specifications, such as `javax.management.ObjectName`. The WebSphere Application Server JMX run time has been enhanced to be aware of the version of the client with which it is communicating. It makes appropriate transformations on these incompatible serialized formats so as to allow the different version run times to communicate with each other. This makes it possible for a Version 5.x administrative client to call a Version 8 deployment manager, node, or server. Similarly, a Version 8 administrative client can call a Version 5.x node or server.

For more information about the JMX architecture, see the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.nd.multiplatform.doc/info/ae/ae/cxml_javamanagementx.html

1.7.1 JMX MBeans

Resources are managed by JMX MBeans. These are not EJBs, but simple Java beans that conform to certain design patterns outlined in the JMX specification.

Providers that want to instrument their systems with JMX need to provide a series of MBeans. Each MBean is meant to wrap, or represent, a certain runtime resource. For example, to expose an application server as a manageable resource, WebSphere needs to provide an application server MBean.

External applications can interact with the MBeans through the use of JMX connectors and protocol adapters. Connectors are used to connect an agent with a remote JMX-enabled management application. This form of communication involves a connector in the JMX agent and a connector client in the management application.

Each JMX-enabled JVM contains an MBean server that registers all the MBeans in the system. It is the MBean server that provides access to all of its registered MBeans. There is only one MBean server per JVM.

WebSphere Application Server provides a number of MBeans, each of which can have different functions and operations available. For example:

- ▶ An application server MBean might expose operations such as start and stop.
- ▶ An application MBean might expose operations such as install and uninstall.

1.7.2 JMX usage scenarios

Some of the more common JMX usage scenarios you might encounter are:

- ▶ Internal product usage:

All WebSphere Application Server administration clients use JMX:

- WebSphere administrative console
- `wsadmin` scripting client
- Admin client Java API

- ▶ External programmatic administration:

In general, most external users are not exposed to the use of JMX. Instead, they access administration functions through the standard WebSphere Application Server administration clients.

However, external users need to access JMX in the following scenarios:

- External programs written to control the WebSphere Application Server run time and its resources by programmatically accessing the JMX API.
- Third-party applications that include custom JMX MBeans as part of their deployed code, allowing the applications components and resources to be managed through the JMX API.

1.8 IBM Support Assistant

IBM Support Assistant is a tool provided by IBM at no charge to troubleshoot a WebSphere Application Server environment. IBM Support Assistant is composed of the following components:

- ▶ IBM Support Assistant Workbench
- ▶ IBM Support Assistant Agent Manager
- ▶ IBM Support Assistant Agent

The IBM Support Assistant Workbench is an Eclipse-based client application that provides the following features:

- ▶ Search capabilities
- ▶ Product information
- ▶ Media viewer
- ▶ Data collection
- ▶ Guided troubleshooter

The majority of the workbench features are only available on the local machine where the workbench is installed, which means that all data that is collected must be manually transferred to the system where the workbench is installed. To extend the capabilities built into the workbench to import and export data to remote systems, the IBM Support Assistant manager and agent have to be installed.

The IBM Support Assistant Agent Manager needs to be installed only once within your network. After being installed, the workbench can authenticate itself against the manager, and collect data from remote servers through the use of the IBM Support Assistant Agent.

The IBM Support Assistant Agent is installed on all systems that you are interested in troubleshooting remotely. After installation, the agent must be registered with the IBM Support Assistant agent manager. The agent manager then exposes the agent to the workbench, which is now capable of transferring files to the agent, remotely collecting data from the agent, and also gathering inventory reports, as shown in Figure 1-15.

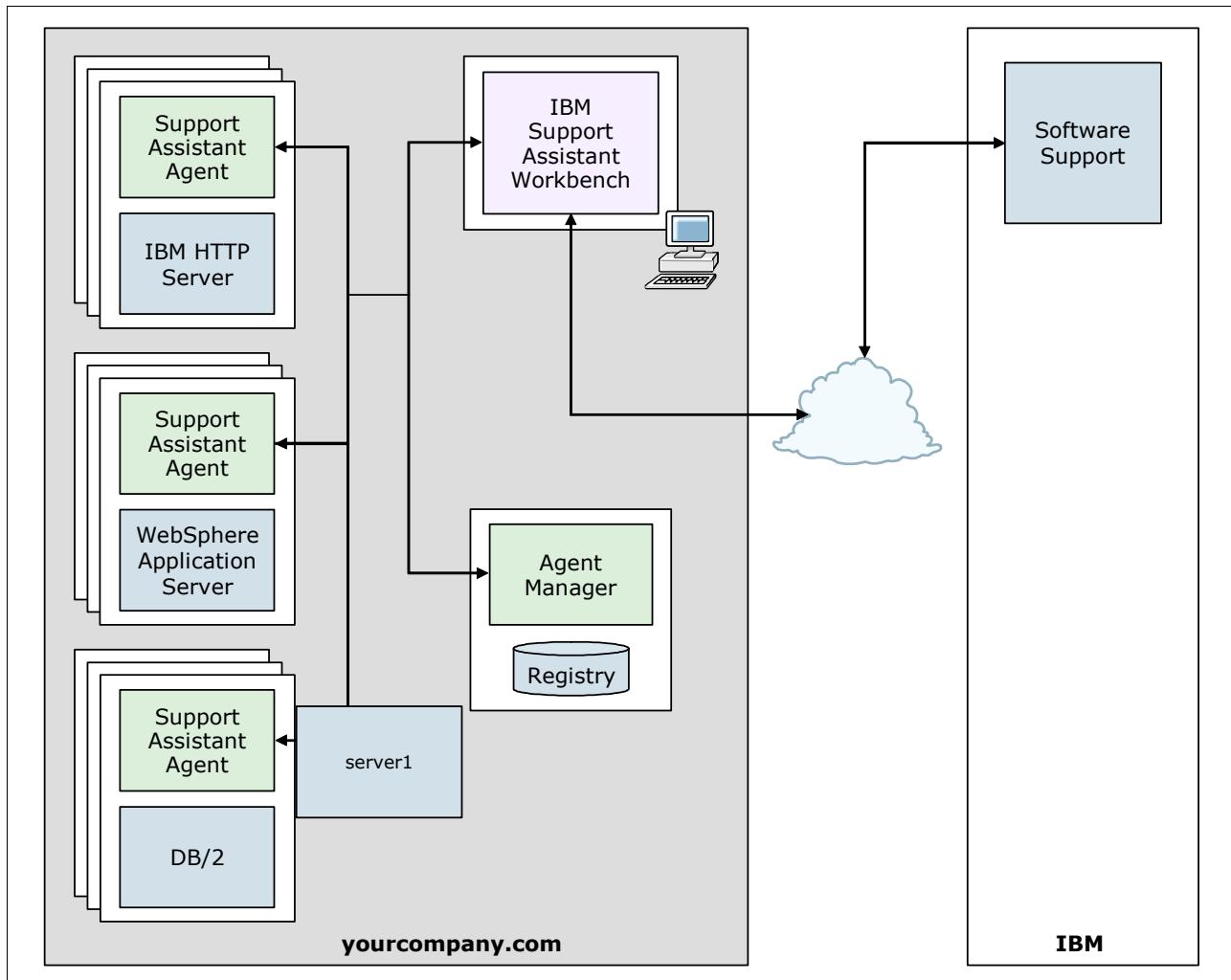


Figure 1-15 IBM Support Assistant

For installation instructions and more details about the IBM Support Assistant, see the following website:

<http://www.ibm.com/software/support/isa/>

If you are new to using IBM Support Assistant, visit the IBM Education assistant tools on IBM Support Assistant for help in getting started at the following website:

http://publib.boulder.ibm.com/infocenter/ieduasst/v1r1m0/topic/com.ibm.iea.selfassist/selfassist/ISAv41_Task.html

THIS PAGE INTENTIONALLY LEFT BLANK

IBM Training: A bright choice for your business *and* your career.



Completing this IBM Redbooks® publication is a great start toward building a solid set of IBM WebSphere® skills. For your next step, reinforce and extend what you've just learned with training from IBM.

The IBM WebSphere Application Server curriculum includes introductory to advanced training for both developers (Java, EJB, web services and AJAX courses) and administrators (problem determination, performance tuning, security and more).

To view abstracts and enroll in IBM WebSphere Application Server courses, visit:

ibm.com/training/websphere/redbook/was/

Energize your career





Installing WebSphere Application Server on distributed systems

This chapter provides a look at the Installation Manager and using the Installation Manager to install WebSphere Application Server V8 and many of its components.

In this chapter, we describe how to install and use the Installation Manager. This chapter contains the following topics:

- ▶ IBM Installation Manager overview
- ▶ Installation Manager installation
- ▶ Using the Installation Manager
- ▶ Working with the Installation Manager
- ▶ Installing WebSphere Application Server
- ▶ WebSphere Customization Toolbox

2.1 IBM Installation Manager overview

Installation Manager is an installation management tool that installs and maintains Installation Manager-based software packages. It is the Eclipse-based tool that enables you to install and modify packages, search for updates, uninstall, and roll back. Installation Manager makes it easier for you to download and install code for a number of IBM software packages.

Starting with WebSphere Application Server V8, Installation Manager V1.4.3.1 or later is used for installation and replaces InstallShield MultiPlatform (ISMP) and Update Installer, which were used to install, update, and uninstall previous versions of WebSphere Application Server. It also replaces the functionality previously provided by the installation factory. Installation Manager V1.4.3.1 comes with WebSphere Application Server V8, but you can also use newer version of Installation Manager to install WebSphere Application Server V8.

Installation Manager was originally introduced to support installation of IBM Rational® products and is currently available for all platforms and supports installation of WebSphere, Rational, and other products. A single instance of Installation Manager can manage the product life cycle for any Installation Manager based product from WebSphere, Rational, IBM Lotus® and any other brand within IBM. It provides the following benefits:

- ▶ Consistency across all platforms using the same methodology
- ▶ Lifecycle management of any Installation Manager installed products
- ▶ Several methods for performing lifecycle management activities
- ▶ Common packaging
- ▶ Validation and system checking performed before downloading binaries
- ▶ More efficiency when delivering new fixes and files for rollback

2.1.1 Terminology overview

In this chapter, we use the following concepts and terminology:

- ▶ A *package* is a software product that can be installed by Installation Manager. It is a separately installable unit that can operate independently from other packages of that software. It can be a product, a group of components, or a single component that can be installed using the Installation Manager. Each package has a name, version, and an identifier as noted here:
 - Package name: com.ibm.websphere.ND.v80
 - Package version: 8.0.0.20110503_0200
 - Package identifier: com.ibm.websphere.ND.v80_8.0.0.20110503_0200
- ▶ The packages are installed to a defined directory location in the file system. Installation Manager allows you to control where products are installed and at which level.
- ▶ A *package group* is used when more than one product is installed at the same location. Package group names are set automatically by Installation Manager. Some packages support installing to the same package group and other packages must be installed to a new package group.
- ▶ A *repository* is a place where the packages to be installed can be found. The repository includes metadata that describes the software version and how it should be installed. It has a list of files organized in a tree structure and can reside on a local directory or on a remotely reachable server.
- ▶ *Shared resources* provide a place where software files and plug-ins are stored and shared by packages. You can only specify the shared resources directory the first time you install a package and you cannot change the location while packages are being installed.

2.2 Installation Manager installation

Installation Manager comes in the form of an installation kit, which contains a set of Installation Manager binaries and a flat-file repository for the Installation Manager product. The installation kit is only used for setup and maintenance of the Installation Manager.

You need to run Installation Manager only on those systems on which you install or update product code. You normally need only one Installation Manager on a computer because one Installation Manager can track any number of product installations.

To begin an installation, you need to obtain the Installation Manager product packages. Product packages can be obtained in one of the following ways:

- ▶ Access the physical disk media and use a local installation.
- ▶ Download the files from the IBM Passport Advantage® site and use a local installation. The website can be found at:
<http://nasoftware.ibm.com/imts/us.nsf/doc/RHIS-7GUMXE>
- ▶ Download the files from an IBM repository site and use a local installation. The website can be found at:
http://www-947.ibm.com/support/entry/portal/Downloads/Software/Software_support_%28general%29

The physical disk contains the rollup of all Installation Manager versions for each supported operating system. When you insert a disk into a local computer, the installation process starts automatically because it recognizes your local operating system. If you are not using a disk on a local computer, you must start the installation process yourself using the appropriate installation file.

You can also download the needed files from an IBM repository site. After the download, extract the downloaded files to a common location, and then determine the files to use to start the installation.

You cannot have multiple versions of Installation Manager installed on the same physical machine. If you try to install on the same physical machine, it returns an error that indicates that it is already installed.

2.2.1 Installation options

You can install Installation Manager using one of the following methods:

- ▶ Launchpad installation: A single point of reference for interactively installing the entire application server environment. The launchpad application is included only in the product physical media. When you insert a disk, the launchpad application has the intelligence to start the appropriate launchpad application based on the operating system.
- ▶ GUI installation: You can install using an interactive installation using a graphical user interface.
- ▶ Command-line installation: You can install silently using the Installation Manager command line (`imcl`) installation command.
- ▶ Console installation: You can install interactively using a command. This method does not include a GUI.
- ▶ Silent installation: You can install using a command where you do not have a GUI or interactive installation option.

2.2.2 Installation

Before installing Installation Manager, you must decide in which mode the Installation Manager will run as well as where the binaries and runtime data will reside. You can install Installation Manager in administrator, non-administrator, or group mode. On UNIX systems, you can install in group mode using a predefined user group. All users in the group can then install and run the same instance of the Installation Manager to manage packages.

Only one administrator instance of Installation Manager can be installed. For non-administrators, install only one instance of Installation Manager for each user.

Launchpad installation

The launchpad application is the starting point for installing a WebSphere Application Server environment. The launchpad identifies components on the product disk that you can install or launch. It is a single point of reference for installing the entire application server environment. The launchpad contains links that launch the installer program for the different products listed in the launchpad. If you click a link in the launchpad that points to a product repository on another disk, you are prompted for that disk. The launchpad also has links for connecting to the WebSphere Application Server Information Center and IBM Education Assistant.

To install Installation Manager using the launchpad, you must have a supported web browser, which includes Mozilla Firefox V3.5 or later or Internet Explorer V6 SP 2 or later. On UNIX systems, you must install the Bash shell package to use the launchpad installation method. You can install and update Installation Manager in administrator and non-administrator modes. However, group mode installation is not supported.

Note: You cannot run the launchpad remotely to install a product. Only local use of the launchpad is supported.

Figure 2-1 shows the launchpad application.

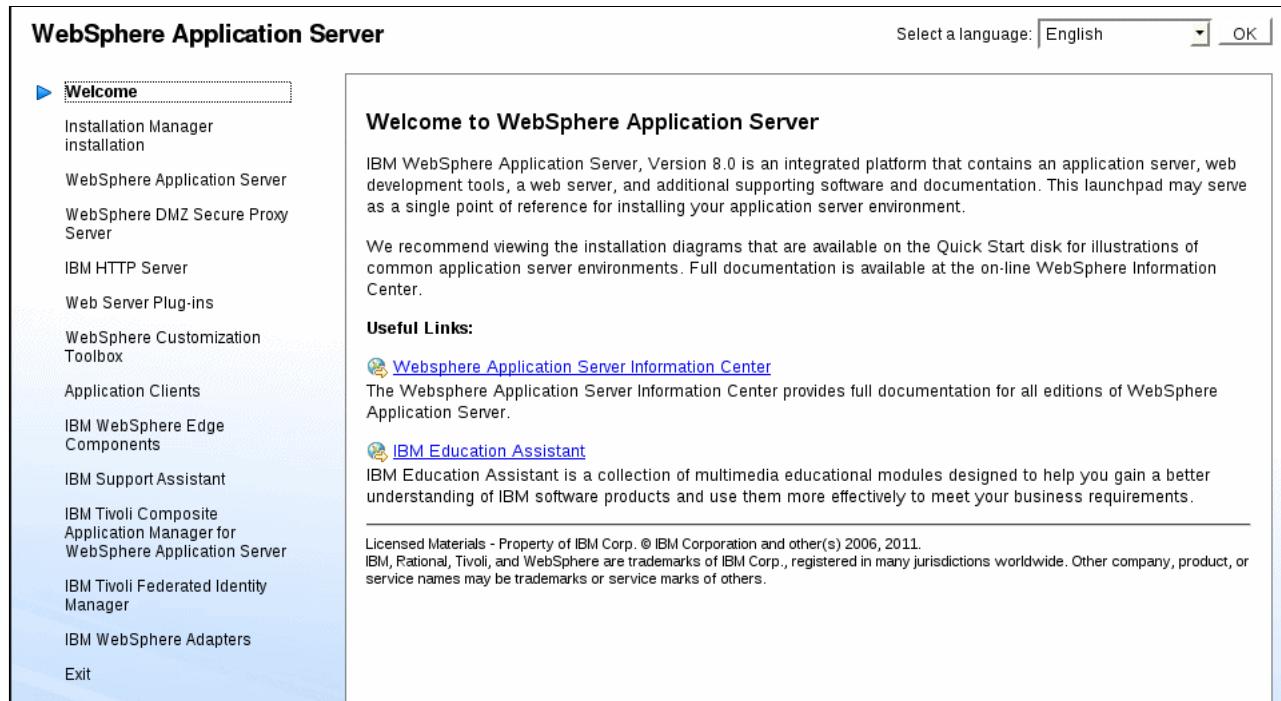


Figure 2-1 Launchpad

GUI installation

To install Installation Manager using the GUI, run one of the following commands:

install	Installs in administrator mode
userinst	Installs in non-administrator mode
groupinst	Installs in group mode

Note: Group mode is not available on Windows and on IBM i.

To begin the installation using the GUI, run the appropriate install command. The following installation is performed as an administrator on a Windows machine:

1. Run the **install.exe** command.
2. The Installation Manager package (Figure 2-2) is selected by default. The status indicates the package will be installed. Click **Next**.

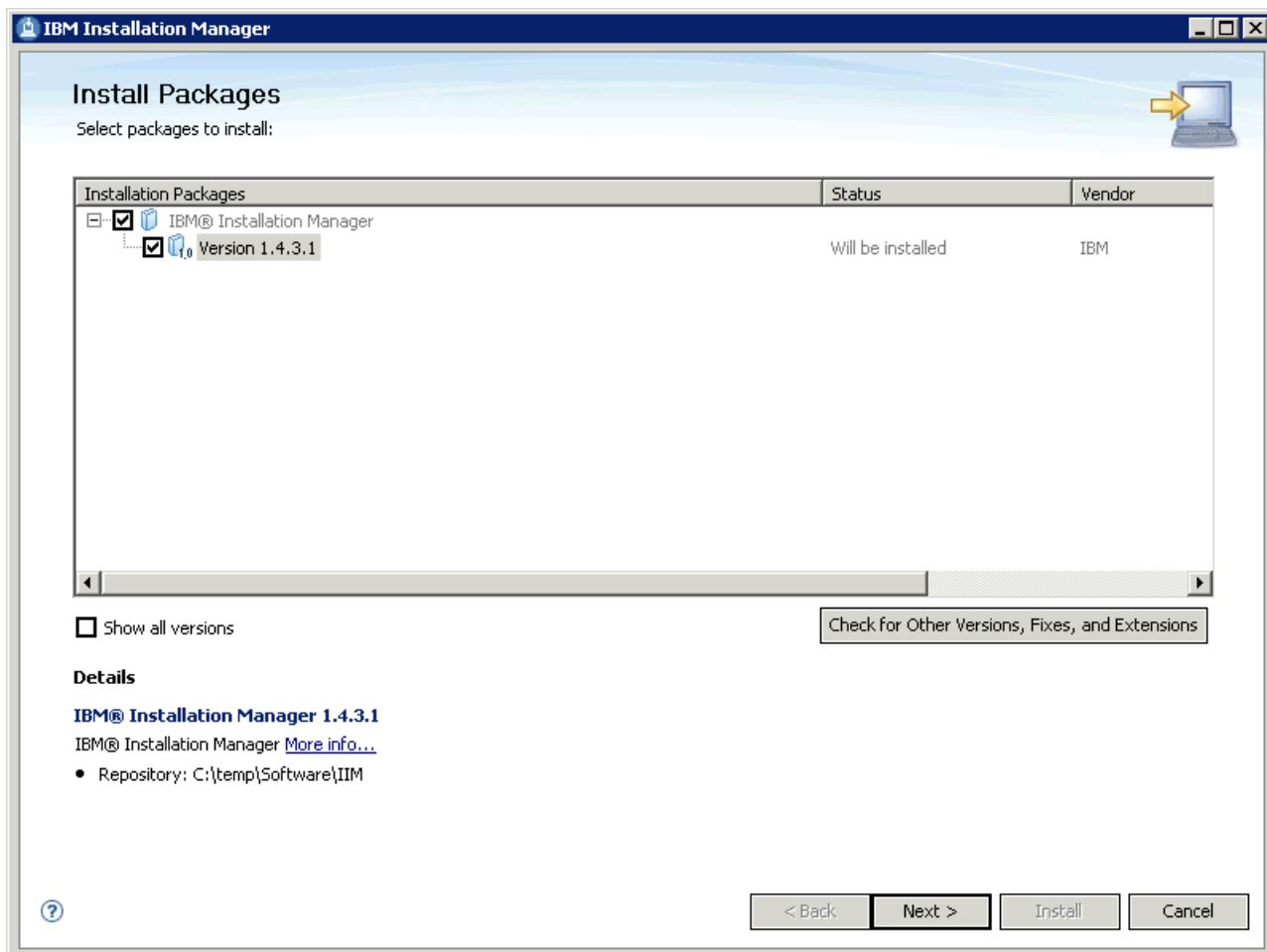


Figure 2-2 Install packages

3. In the Licenses window (Figure 2-3), you can read the license agreement. Click **I accept the terms of the license agreement** and click **Next**.

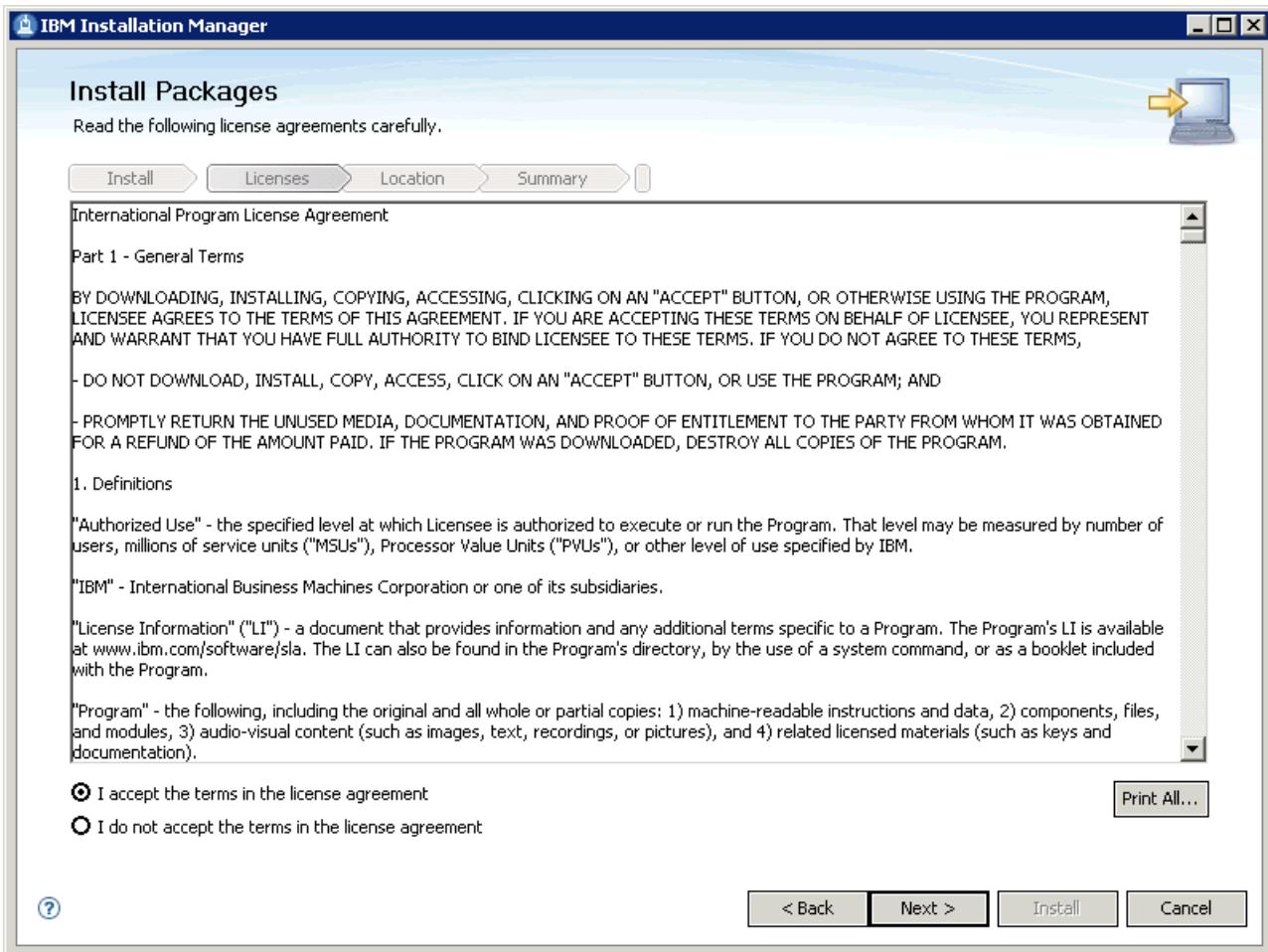


Figure 2-3 License agreement

4. In the Location window (Figure 2-4), the installation directory is entered. Enter an Installation Manager Location directory, or keep the default directory. The installation location must be a directory named `eclipse`. The default directory location differs depending upon the operating system of the machine. In this example, the directory location has been modified. Click **Next**.

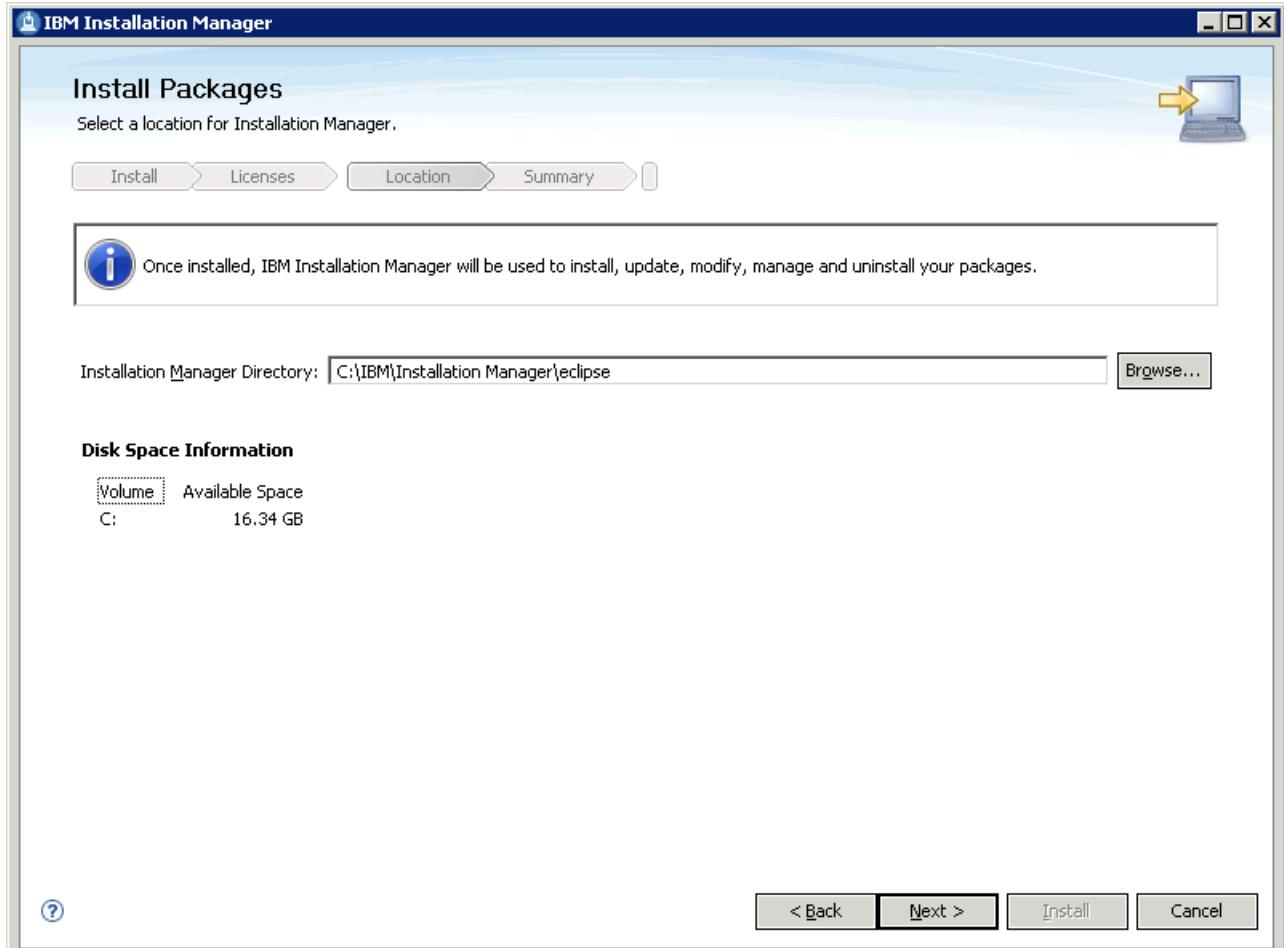


Figure 2-4 Installation directory

5. In the Summary window (Figure 2-5), review the packages to be installed and click **Install** to begin the installation.

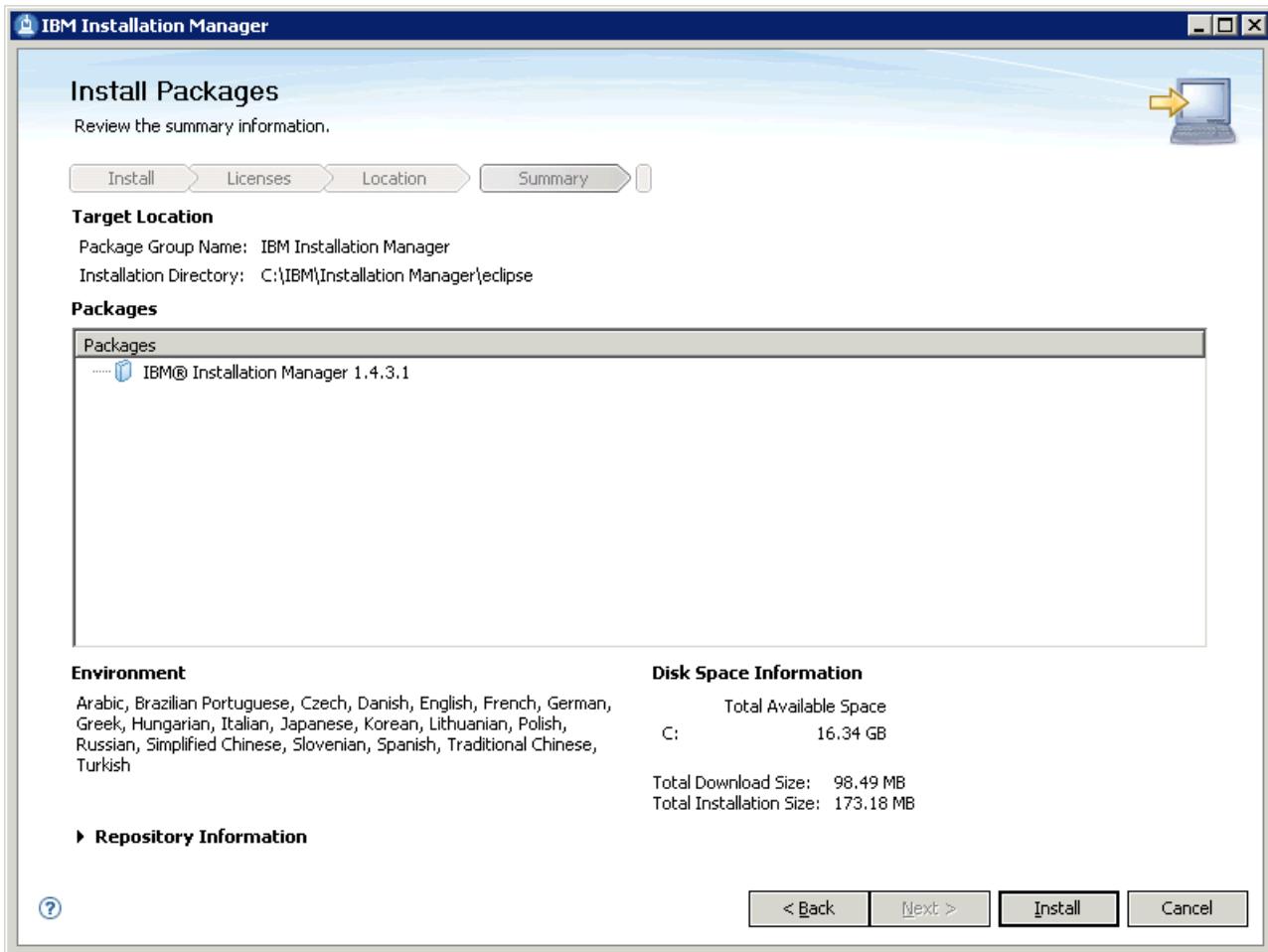


Figure 2-5 Install summary

6. During the installation process, you can observe the progress of the installation. At any time, you can select to pause or cancel the installation. When the installation completes, a message appears indicating the status.

In this window (Figure 2-6), you can view the log file for the installation. Click **Reset Installation Manager** to restart the Installation Manager in GUI mode.

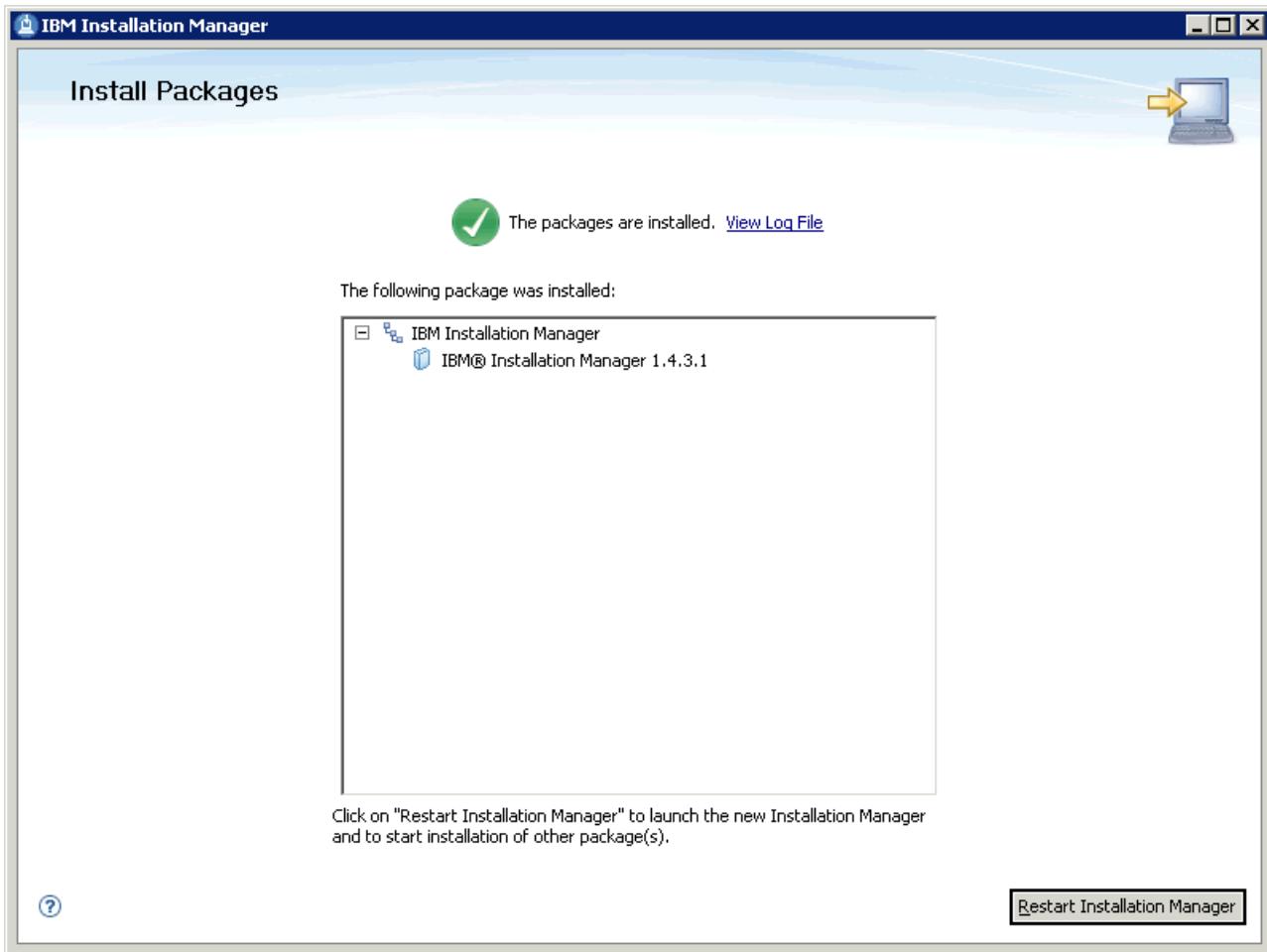


Figure 2-6 Installed packages summary

Command-line installation

To install Installation Manager using the command line, run the `imcl` command as the administrator, non-administrator, or group. The `imcl` command can be found in the `<IM_install>\eclipse\tools` directory.

When using the `imcl` command, you must identify the following installation attributes in the command line:

<code>package_id_version</code>	Indicates the package ID / feature ID that is defined in the <code>install.xml</code> file. It is required because it specifies the offering to be installed.
<code>repository</code>	Indicates the source repository for the installation.
<code>installationDirectory</code>	Indicates the installation directory for Installation Manager, which must include a path that contains spaces in quotation marks.
<code>accessRights</code>	Defines the user you are using to install. If this is not defined, it defaults to admin.
<code>acceptLicense</code>	Indicates that you accept the license agreement.

You can obtain a full listing of supported attributes for `imcl` by running the `imcl "help"` command.

Example 2-1 shows how to install Installation Manager using the command line as an administrator where the software package has been downloaded to the local machine.

Example 2-1 Command-line installation

```
C:\temp\software\IIM\tools>imcl.exe install com.ibm.cic.agent -repositories  
C:\temp\software\IIM\repository.config -installationDirectory  
C:\IBM\InstallationManager\eclipse -accessRights admin -acceptLicense
```

Console mode installation

The console mode is a non-graphical, text-based, and interactive installation method.

To install Installation Manager using console mode, run one of the following commands:

<code>installc -c</code>	Installs in administrator mode
<code>userinstc -c</code>	Installs in non-administrator mode
<code>groupinstc -c</code>	Installs in group mode

Example 2-2 shows the installation of Installation Manager using the console mode as an administrator where the software package has been downloaded to the local machine. In console mode, a selected option is indicated by [X].

Example 2-2 Console mode installation

```
C:\temp\software\IIM>installc.exe -c  
Preprocessing the input.  
Loading repositories...  
Preparing and resolving the selected packages...  
  
=====> IBM Installation Manager> Install  
  
Select packages to install:  
  1. [X] IBM? Installation Manager 1.4.3.1  
  0. Check for Other Version, Fixes, and Extensions  
  
      N. Next, C. Cancel  
----> [N] N  
  
=====> IBM Installation Manager> Install> Licenses  
Read the following license agreements carefully.  
View a license agreement by entering the number:  
  1. IBM Installation Mnager - License Agreement  
  
Options:  
  A. [ ] I accept the terms in the license agreement  
  D. [ ] I do not accept the terms in the license agreement.  
  
  B. Back, C. Cancel  
----> [N] A  
  
=====> IBM Installation Manager> Install> Licenses  
Read the following license agreements carefully.  
View a license agreement by entering the number:  
  1. IBM Installation Mnager - License Agreement
```

Options:

- A. [X] I accept the terms in the license agreement
- D. [] I do not accept the terms in the license agreement.

B. Back, N. Next, C. Cancel

----> [N] N

====> IBM Installation Manager> Install> Licenses

Read the following license agreements carefully.

View a license agreement by entering the number:

- 1. IBM Installation Mnager - License Agreement

Options:

- A. [X] I accept the terms in the license agreement
- D. [] I do not accept the terms in the license agreement.

B. Back, N. Next, C. Cancel

----> [N] N

====> IBM Installation Manager> Install> Licenses > Location

Installation Manager installation location:

C:\Program Files <x86>\IBM\Installation Manager\eclipse

Options:

- L. [] Change Installation Manager installation location

B. Back, N. Next, C. Cancel

----> [N] L

====> IBM Installation Manager> Install> Licenses > Location

Enter IM location

Enter new value for the IM installation location <press <Enter> to cancel>:

----> C:\IBM\InstallationManager\eclipse

====> IBM Installation Manager> Install> Licenses > Location

Installation Manager installation location:

C:\IBM\InstallationManager\eclipse

Options:

- L. [] Change Installation Manager installation location

B. Back, N. Next, C. Cancel

----> [N] N

====> IBM Installation Manager> Install> Licenses > Location > Summary

Target Location:

Package Group Name: : IBM Installation Manager

Installation Directory :C:\IBM\InstallationManager\eclipse

Packages to be installed:

IBM? Installation Manager 1.4.3.1

Options:

- G. [] Generate installation response file

B. Back, I. Install, C. Cancel

```

----> [I] I
-----25%-----50%-----75%
-----100%
-----
=====
====> IBM Installation Manager> Install> Licenses > Location > Summary
      Completion

the install completed successfully.

Options:
  R. Restart Installation Manager
----> [R] R
=====> IBM Installation Manager
Select:
  1. Intall - Install software packages
  2. Update - Find and intall updates and fixes to the installed packages
  3. Modify - change installed software packages
  4. Roll Back - Revert to an earlier version of the intalled software packages
  5. Uninstal - Remove the installed software packages

Other Options:
  L. View Logs
  S. View Installation History
  V. View Installed Packages
-----
  P. Preferences
-----
  E. Export Data for Problem Analysis
  A. About IBM Installation Manager
-----
  X. Exit Installation Manager
----> X

```

Silent installation

To install Installation Manager silently, run one of the following commands:

installc	Installs in administrator mode
installc	Installs in non-administrator mode on Windows and UNIX
userinstc	Installs in non-administrator mode on IBM i
userinstc	Installs as the current user
groupinstc	Installs in group mode

Note: Group mode silent installation is not available on Windows.

Example 2-3 shows how to install Installation Manager silently as an administrator where the software package has been downloaded to the local machine.

Example 2-3 Silent installation

```
C:\temp\software\IIM> installc.exe -acceptLicense
```

2.3 Using the Installation Manager

After you have installed Installation Manager, you can use it to install other packages, update or modify them, and so on. The Installation Manager tracks the packages it installs, including selectable features and maintenance updates for products.

There are a number of ways you can interact with the Installation Manager.

- ▶ Wizard mode: A graphical user interface to run the Installation Manager.
- ▶ Command-line mode: The command-line utility (**imcl**) to manage installations.
- ▶ Console mode: An interactive text-based user interface that is used to run the Installation Manager.
- ▶ Silent mode: Used to perform installations using a response file and run from the command line or a file.

2.3.1 Various modes in Installation Manager

In this section, we provide information about the various modes of Installation manager.

Wizard mode

The Installation Manager includes a number of wizards to help maintain product packages. Installation Manager V1.4.3.1 includes the following wizards:

- ▶ The Install wizard steps you through the installation process of a package. It provides various default settings that can be customized for your environment. You can install multiple packages simultaneously.
- ▶ The Update wizard allows you to update currently installed packages.
- ▶ The Modify wizard allows you to change certain elements of installed packages. After a product package has been installed, it can be modified to add or remove features of the package.
- ▶ The Rollback wizard allows you to revert to a previous version of a package.
- ▶ The Uninstall wizard removes installed packages.

To use the Installation Manager wizard mode, navigate to the Installation Manager install directory and run one of the following commands:

- ▶ Windows: **IBMIM.exe**
- ▶ UNIX: **IBMIM**

Figure 2-7 shows the Installation Manager GUI wizard.

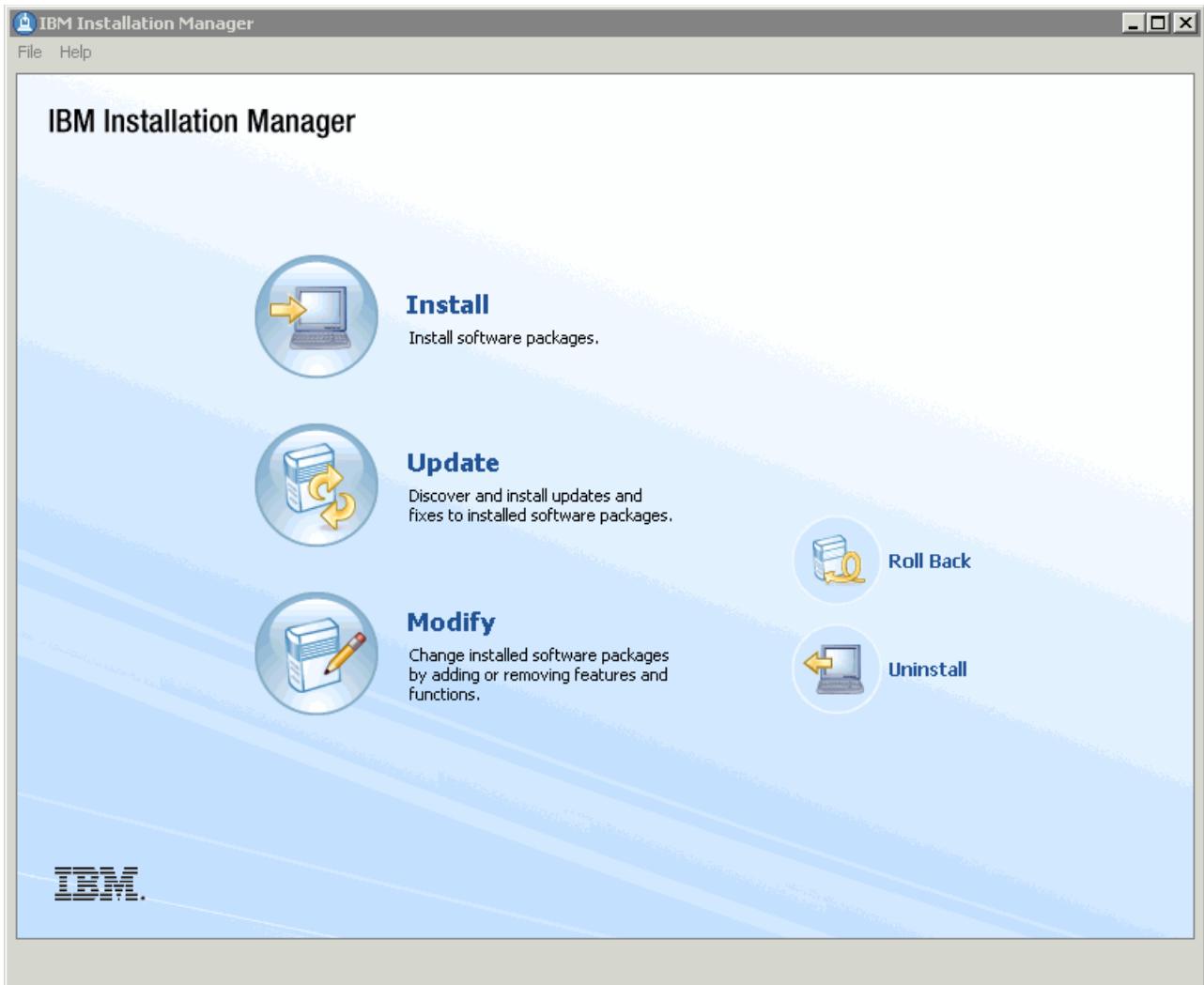


Figure 2-7 Installation Manager GUI

Information: WebSphere Application Server V8 comes with Installation Manager V1.4.3.1. WebSphere Application Server includes IBM Assembly and Deploy Tools for WebSphere Administration, which replaces IBM Rational Application Developer Assembly and Deploy function included in WebSphere Application Server V7. IBM Assembly and Deploy Tools for WebSphere Administration enables rapid assembly and deployment of applications to WebSphere Application Server environments. To install IBM Assembly and Deploy Tools for WebSphere Administrations, Installation Manager V1.4.4 is required. You can update the Installation Manager to Version 1.4.4 automatically or download the fix pack com.ibm.cic.agent.offering_1.4.4000.20110525_1254.zip to update the current installation.

For more information about installing or updating Installation Manager V1.4.4, see the following website:

<https://www-304.ibm.com/support/docview.wss?uid=swg24029226#downloads>

Installation Manager V1.4.4 includes a Manage Licenses wizard that helps you manage licenses for your installed packages.

Command-line mode

If you cannot use the GUI mode, or have a preference for a non-GUI environment, you can use the command-line mode to manage installations. Using the command-line mode, you can install, update, and uninstall packages, list installed features and packages, list available packages, display version information, import a response file to be used for a silent installation, and obtain help for the command line.

The **imcl** command can be found in the <IM_install>/tools/ directory. Example 2-4 illustrates how to obtain help for the command-line mode and provides a listing of all commands and options.

Example 2-4 Command-line mode

```
C:\IBM\InstallationManager\eclipse\tools> imcl.exe help
help, -help, -h, -? (all, <commands>
    Print short info about available commands and exit.
input, -input <script file>
    Execute Installation Manager script file.
install <packageId(_Version)(,featuN, featureM,...)>...
    Install packages or specific features.
installAll, -installAll
    Silently install all available packages.
listAvailableFixes <id>_<version>
    List information about available fixes.
listAvailablePackages
    List information about available packages.
listInstalledPackages
    List information about packages currently installed on the system.
record, -record <recordedFile>
    Record a response file.
uninstall <packageId(_Version)(,featuN, featureM,...)>...
    Uninstall packages or specific features.
updateAll, -updateAll
    Silently update all installed packages.
version, -version
    Print the version of this application and exit.
```

```

-acceptLicense
    Indicate acceptance of the license agreement.
-accessRights, -aR <access rights>
    Define the user as an admin, a nonAdmin or a group. The default value is admin.
    This setting ignores the system status.
-consoleMode, -c
    Run Installation Manager in console mode.
-dataLocation, -dL <data-location>
    Specify a directory to hold internal Installation Manager data.
-log, -l <log file>
    Create a log file from the program script execution.
-passwordKey, -pK <passwordKey>
    Provides password encryption key in UI or silent mode
-showProgress, -sP
    Show progress.
-showVerboseProgress, -sVP
    Show verbose progress.
-silent, -s
    Run Installation Manager in silent mode.

```

Console mode

Another option for interacting with the Installation Manager to manage installations is through the console mode. Console mode is a non-graphical, command-line, and text-based interactive mode for the Installation Manager. In console mode, you can set and examine preferences, install, update, modify, rollback, and uninstall packages.

To start console mode:

- ▶ Windows: Navigate to <IM_install>\tools\ and run **imcl.exe -c**
- ▶ UNIX: Navigate to <IM_install>/tools/ and run **imcl -d**

Figure 2-8 shows the main screen of console mode.

```

[root@RHEL56 tools]# ./imcl -c
=====> IBM Installation Manager

Select:
  1. Install - Install software packages
  2. Update - Find and install updates and fixes to the installed packages
  3. Modify - Change installed software packages
  4. Roll Back - Revert to an earlier version of the installed software packages
  5. Uninstall - Remove the installed software packages

Other Options:
  L. View Logs
  S. View Installation History
  V. View Installed Packages
  -----
  P. Preferences
  -----
  E. Export Data for Problem Analysis
  A. About IBM Installation Manager
  -----
  X. Exit Installation Manager

----->

```

Figure 2-8 Console mode main display

Note: WebSphere Application Server V8 does not support console mode installation.

Silent mode

Silent mode allows you to install packages in a non-interactive and non-GUI mode. In a silent mode installation, a response file is used to provide the input for the installation. The three steps to a silent installation are as follows:

1. Install Installation Manager.
2. Record a response file.
3. Run the installation in silent mode.

Response files can be used to install, update, modify, roll back, and uninstall software packages.

Creating response files

A response file can be recorded using the Installation Manager or manually created by using a documented list of commands. You can record a response file using the Installation Manager to install, update, modify, roll back, or uninstall a package. To record a response file for the installation of package using the Installation Manager, the **-skipInstall** argument is used in the command line. The **-skipInstall** argument does not install any package or files; it speeds up the installation process by just creating an XML response file. The installation process can then be automated by using this response file.

You can use the Installation Manager GUI to record a response file. Start the GUI with the following options:

- ▶ **-skipInstall**: Indicates to skip the install.
- ▶ **<agentDataLocation>**: Specifies a directory to hold internal Installation Manager data. Create a unique directory for each new installation.
- ▶ **-record <responsefile>**: Specifies the response file to be created.

Example 2-5 shows how to create a response file using the Installation Manager GUI on Windows. After you run this command, it kicks off the GUI, where you tell it what repositories to search through, which packages to select, and to click **Install** to generate the script. Nothing is actually installed; only a response file is created.

Example 2-5 GUI response file creation

```
C:\IBM\InstallationManager\eclipse>IBMIM.exe -skipInstall C:\temp\imRegistry  
-record C:\temp\silent_install_response_file.xml
```

To manually create a response file, you can use a sample response file that is provided and customize it to suit your environment. You can find sample response files located at the following website:

http://publib.boulder.ibm.com/infocenter/install/v1r4/index.jsp?topic=/com.ibm.install112.doc/topics/c_sample_response_files.html

You must record a response file on the same platform that you plan for the installation. If you plan installations for multiple platforms, you must have a response file for each platform. For example, to install on a computer running Microsoft Windows, you must record the response file on a computer that runs Windows.

Using response files

After you create an install response file, you can use it to silently install a package. Start the installation using the command line with the following options:

- ▶ **-input:** Where you identify the response file.
- ▶ **-log:** Where you can provide a file name for the log file to be created for the installation.

Example 2-6 shows how to silently install a package using the Installation Manager command-line mode.

Example 2-6 Silent installation

```
C:\IBM\InstallationManager\eclipse\tools>imcl.exe -acceptLicence input  
C:\temp\silent_install_reponse_file.xml -log C:\temp\install_log_file.xml
```

2.4 Working with the Installation Manager

You can influence the characteristics of Installation Manager by setting preferences. The Installation Manager has a number of preferences defined for repositories, appearance, files for rollback, help, Internet settings, Passport Advantage settings, and updates. You can keep the default settings for preferences or they can be modified to suit your environment.

2.4.1 Installation Manager preferences

The easiest way to examine and modify the Installation Manager preferences is by using the GUI wizard mode. Preferences can also be examined and modified using the console mode. To verify or modify the preferences using the GUI, select **File → Preferences**. Figure 2-9 shows the Preferences page.

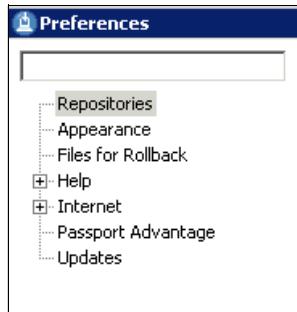


Figure 2-9 Installation Manager preferences

Repositories

This preference identifies any number of repositories to be used by the Installation Manager for installing, modifying, and updating packages.

Appearance

This preference allows you to select whether or not the internal version of the packaging being installed is listed. If this option is selected, the internal version is displayed during the installation process and includes the release number, year, month, day, and package ID. By default, this option is not selected.

Files for Rollback

This preference allows use of the rollback feature to revert to a previous installed version of an updated package.

Help

This preference allows you to indicate how help information is displayed. For example, help contents can be launched in the help browser, which is the default setting, or launched in an external browser. You can also configure access to multiple information centers for products you are installing. You have the option of including local or remote help and setting a priority of which information center is used first.

Internet

This preference allows you to set preferences for proxy servers.

Passport Advantage

This preference is used to provide the settings for a Passport Advantage site. If this option is selected, internet connectivity is required. By default, this option is cleared. Select **Connect to Passport Advantage** to enable and you are prompted for a user name and password for Passport Advantage. There is an option to save this password.

Updates

This preference is used to indicate if the Installation Manager searches for updates to itself when installing, modifying, or updating packages. If this option is selected, internet connectivity is required. By default, this option is cleared.

When making any changes to the Installation Manager preferences using the GUI, click **Apply** to save the changes. Click **OK** to exit the preferences.

2.4.2 Repository overview

The Installation Manager uses a repository to identify the packages or updates to install. A repository is a location that stores data for installing, modifying, rolling back, updating, or uninstalling packages. Each installed package has an embedded location for its default update repository. You can add, edit, or remove repositories to be used by the Installation Manager.

The Installation Manager determines and lists all the available packages to install, which includes products, fix packs, interim fixes, and so on, based on the configured repositories. It checks prerequisites and interdependencies, and installs the selected packages.

The Installation Manager repository contains one or multiple product offerings that have both metadata and actual payload for the offerings. The offering metadata describes such aspects of the offering, such as the following:

- ▶ Name, version, and supported platforms
- ▶ Required and optional features
- ▶ Relationships and dependency between offerings and features of offerings

Normally, an Installation Manager repository contains the full content that is required to install on various platforms, operating systems, and so on.

Repository topologies can be generalized as the following categories:

- ▶ Public repository: Accessible to the general public at an IBM hosted site, such as IBM Passport Advantage.
- ▶ Local repository: Used by single user and not shared with others.
- ▶ Enterprise repository: Located behind the firewall, and accessed by multiple machines within the enterprise.

2.4.3 Repository configuration

In this section, we provide information about the configuration of repositories.

Service repository

By default, the Installation Manager is configured to use a service repository that is made up of repositories at an IBM repository site. In this case, Internet access is required. If a machine does not have Internet access, the Installation Manager can be configured to look for a local repository. Updates can be downloaded and placed in a temporary directory on the machine. The Installation Manager looks in this directory for updates to be installed. You must manually configure local repositories.

The Search service repositories during installation and updates option has the Installation Manager search the service repositories for updates to installed packages. Service repositories contain product updates and each IBM product includes links to service repositories that are specific to the product. Installation Manager searches both the service repositories and the repositories listed in the repository table of the repository preferences panel.

If a machine does not have access to an IBM repository site or you do not want the machine to access that site, clear the **Search service repositories during installation and updates** option. When this option is checked, the Install, Modify, and Update wizards try to access the service repositories. If a connection is not made, Installation Manager times out and then tries to reconnect before starting the installation, update, or modify.

To verify or modify the service repository setting, click **File → Preferences → Repositories** in the Installation Manager GUI. The Repositories tab contains the area for repository configuration.

Figure 2-10 shows the Repositories window for the Installation Manager.

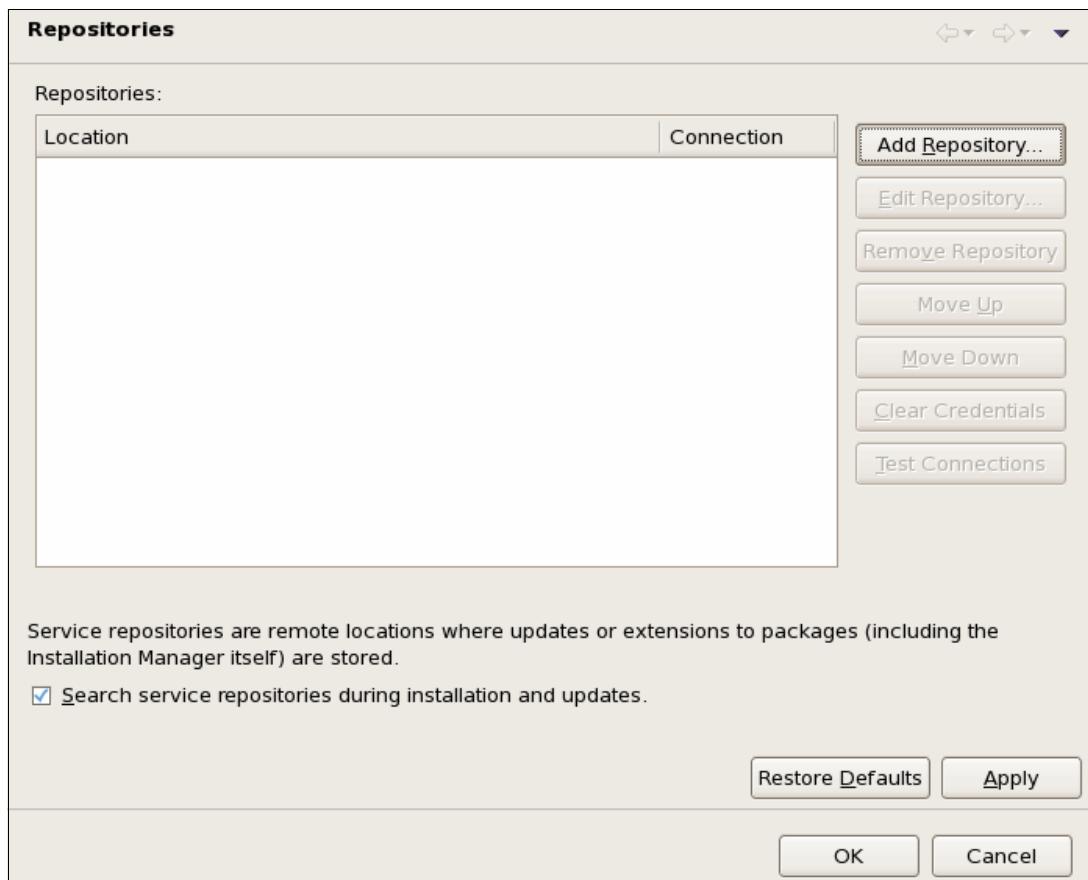


Figure 2-10 Installation Manager repositories window

Adding repositories

You can add a number of repositories that the Installation Manager searches for available packages when installing, modifying, or updating a product package. To add a repository, select **Add Repository** and select the repository location and file type. The repository file types can be any of the following:

- ▶ A repository.config file included in the product repository files.
- ▶ A diskTag.inf file that tells the Installation Manager that the files are from a disk.
- ▶ A JAR file that contains a repository. For example, license kits are distributed using a JAR file.
- ▶ A compressed file that contains a diskTag.inf file. Compressed files should be extracted to the local system prior to use.

The Installation Manager searches the repositories in the order they are listed in the repository window. If two repositories use the same package, the repository listed higher in the order is used. You can move a repository up and down in the order listed by selecting the appropriate repository and clicking either **Move Up** or **Move Down**.

The Installation Manager only searches the repositories that you have selected in the repository page. To select a repository, select the check box before the repository name. If you clear the check box before the repository name, the Installation Manager will not search the repository.

Figure 2-11 shows the Installation Manager repositories window where two repositories have been added.

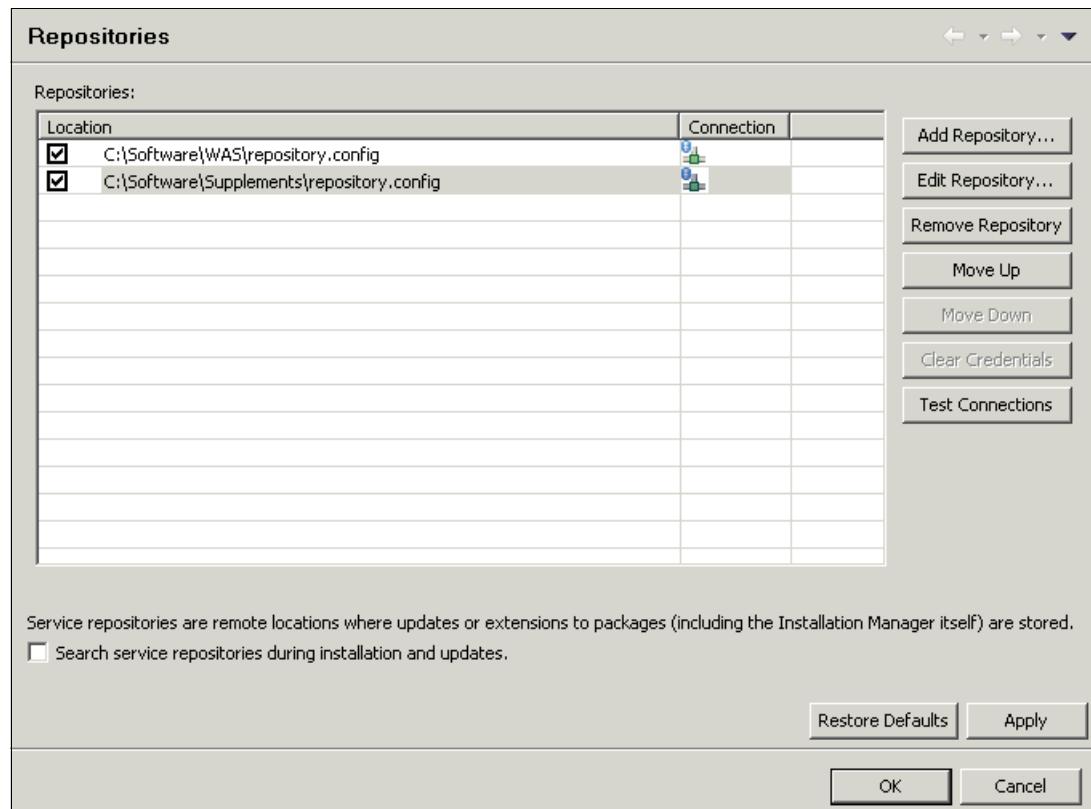


Figure 2-11 Installation Manager repositories window

After you add a repository, you can test the connection to the repository. Select the check box before the repository name and click **Test Connections**. If the Installation Manager can access the repository, the repository is connected and the icon reflects this status. If a connection cannot be made to a repository, a message indicates the failed connection and icon reflects this status.

You can also edit and remove repositories from the repository listing.

2.4.4 Updating the Installation Manager

The Installation Manager can be configured to automatically search for updates to itself. To verify or modify the setting, click **File** → **Preferences** → **Updates** in the Installation Manager GUI. The Search for Installation Manager updates option has the Installation Manager search for updates automatically. If this option is selected, network connectivity is required, and the Installation Manager looks for updates for itself. If it finds an update, when you click the install, modify, or update a package, you get a message that prompts you to update the Installation Manager version.

If the option is cleared, the Installation Manager does not look for updates to itself. You need to download a fix pack to update the current installation or download the full installation of the needed version of Installation Manager.

For more information about installing or updating Installation Manager, see the following website:

<http://publib.boulder.ibm.com/infocenter/install/v1r4/index.jsp>

2.4.5 Key features of Installation Manager

In addition to installing packages, the Installation Manager has a few other key features. The Installation Manager can update, modify, or roll back packages. Prior to updating, modifying, or rolling back a package, verify that all processes and programs for the package have been stopped or closed.

Updating packages

Using the Installation Manager update feature, you can locate and install product updates and new features for packages that have been installed using the Installation Manager. You can update a package by automatically searching for and applying an update if the Search service repositories during installation and updates option is selected, or you can download an update and apply it manually. Updates can be one of the following:

- ▶ Fix packs, which are updates to a software package. The Installation Manager indicates that a new version of the software package is available.
- ▶ Interim fixes, which apply to a specific version of a software package and are typically a fix for a critical issue.

You can check the WebSphere Application Server Support page for information about any fix packs or interim fixes at the following website:

<http://www-01.ibm.com/support/docview.wss?uid=swg27004980>

You can also download fix packs and interim fixes from IBM Fix Central at the following website:

<http://www-933.ibm.com/support/fixcentral/>

Modifying packages

Using the Installation Manager modify feature, you can modify the packages that are installed using the Installation Manager, which includes adding or removing features for an installed package. For example, you want to add an additional language pack to the current installation of WebSphere Application Server.

Rolling back packages

Using the Installation Manager rollback feature, you can remove an update and revert to a previous version. For example, you apply a fix to your existing environment and now want to remove that fix. Simply use the rollback feature to roll back to the previously installed package prior to applying the fix.

The Installation Manager saves earlier versions of a package in files stored on the system and uses these files to roll back. If you removed these files, the Installation Manager must have access to your installation repository or disk media. When you roll back a package, Installation Manager uninstalls the updated resources, and reinstalls the resources from the previous version.

Uninstalling packages

Using the Installation Manager uninstall feature, you can uninstall packages that have previously been installed by the Installation Manager.

2.4.6 Listing packages

When using the Installation Manager, you can list the installed packages and list the available packages that can be installed, updated, modified, and rolled back.

Example 2-7 shows how to list available packages using the Installation Manager command-line mode.

Example 2-7 Listing available packages

```
C:\IBM\InstallationManager\eclipse\tools>imcl.exe listAvailablePackages  
-repositories C:\temp\software\Supplements\repository.conf  
com.bm.websphere.APPCLIENT.v80.8.0.0.20110503_0200  
com.bm.websphere.IHS.v80.8.0.0.20110503_0200  
com.bm.websphere.PLG.v80.8.0.0.20110503_0200  
com.bm.websphere.PLUGCLIENT.v80.8.0.0.20110503_0200  
com.bm.websphere.WCT.v80.8.0.0.20110503_0200
```

Example 2-8 shows how to list installed packages using the Installation Manager command-line mode.

Example 2-8 Listing installed packages

```
C:\IBM\InstallationManager\eclipse\tools>imcl.exe listInstalledPackages  
com.ibm.cic.agent_1.4.3001.20110504_1325
```

You can also list installed packages using the Installation Manager GUI by clicking **File** → **View Installed Packages**.

In Windows, click **Start** → **All Programs** → **IBM Installation Manager** → **View Installed Packages**. The installed packages information opens in a web browser.

2.4.7 Examining the log files

The Installation Manager creates log files that you can use to troubleshoot any installation problems. It is a good idea to consult the log files after any installation to verify that it was successful. You can view log files immediately after installation. When using the GUI installation method, a link to View Log File is on the summary page. This link launches the Installation Log GUI.

You can view logs at any time using the Installation Log GUI. Launch the Installation Manager GUI interface and click **File** → **View Log**. The GUI provides a convenient interface where all log files can be examined. Using the GUI, you can:

- ▶ Export an XML log file to a location in the file system.
- ▶ Filter search contents to narrow down the results displayed. You can filter by the severity level of the event in the log by Error, Warning, Information, and Note. By default, all options are selected except Information.
- ▶ Open a selected log file in a web browser.

Figure 2-12 displays the Installation Manager GUI log file interface.

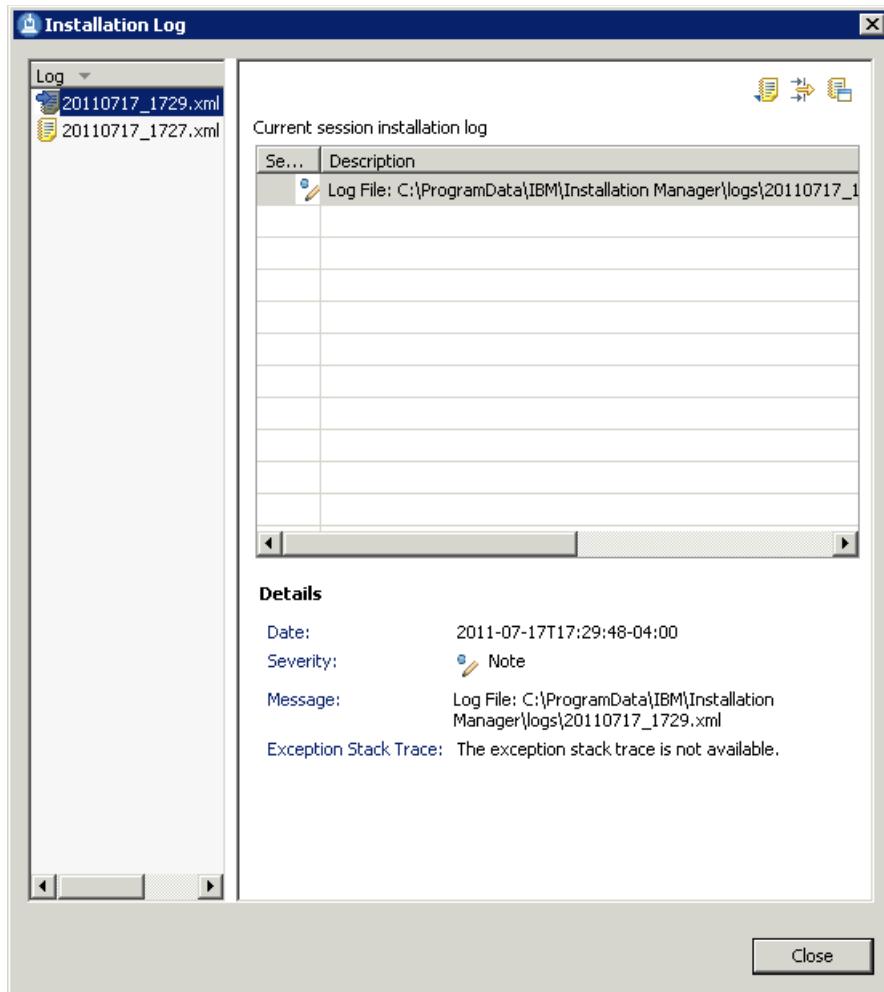


Figure 2-12 Log files

Log files can be examined manually and can be located in the following directories:

- ▶ Windows: C:\ProgramData\IBM\Installation Manager\logs
- ▶ UNIX: /var/ibm/InstallationManager/logs

2.4.8 Uninstalling the Installation Manager

Before you uninstall Installation Manager, you must uninstall all of the packages that were previously installed by Installation Manager. Be sure to also close the Installation Manager before starting the uninstall process. You must also log into the machine as the identity you used when installing Installation Manager.

To uninstall Installation Manager, complete one of the following actions:

- ▶ Windows GUI uninstall:

Click **Control Panel** → **Add or Remove Programs**. Click **IBM Installation Manager** and click **Remove**.

- ▶ Windows silent uninstall:
Navigate to the C:\ProgramData\IBM\Installation Manager\uninstall directory and run **uninstallc.exe** in admin mode or **userinstc.exe** in non-admin mode.
- ▶ For UNIX GUI uninstall:
Navigate to the /var/ibm/Installation Manager/uninstall directory and run **uninstall**.
- ▶ For UNIX silent uninstall:
Navigate to the /var/ibm/Installation Manager/uninstall directory and run **uninstallc**.

2.5 Installing WebSphere Application Server

There are a number of ways you can install WebSphere Application Server using the Installation Manager. Launchpad is a web application that is the starting point for installing WebSphere Application Server. To use the launchpad to install WebSphere Application Server, you must have already installed the Installation Manager. You can also install using the Installation Manager GUI or silently using response files.

Note: WebSphere Application Server V8 does not support console mode installation.

2.5.1 Checking prerequisites

Prior to installation, you can verify the list of hardware and software requirements and supported platforms for WebSphere Application Server V8 at the following website:

<http://www-01.ibm.com/support/docview.wss?rs=180&uid=swg27006921>

Planning information for a WebSphere Application Server installation can be found at the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.installation.nd.doc/info/ae/ae/tins_scenario3.html

2.5.2 Installing WebSphere Application Server using the GUI

Complete the following steps to install WebSphere Application Server in admin mode:

1. Start the Installation Manager using the IBMIM command.
2. Add a repository for the WebSphere Application Server package.
3. Click the **Install** wizard to begin the installation.

4. The Installation Manager searches the defined repositories and lists the packages that it finds. In the Install Packages window (Figure 2-13), the WebSphere Application Server package is listed by default. Select the package and the status indicates the package will be installed. Click **Next**.

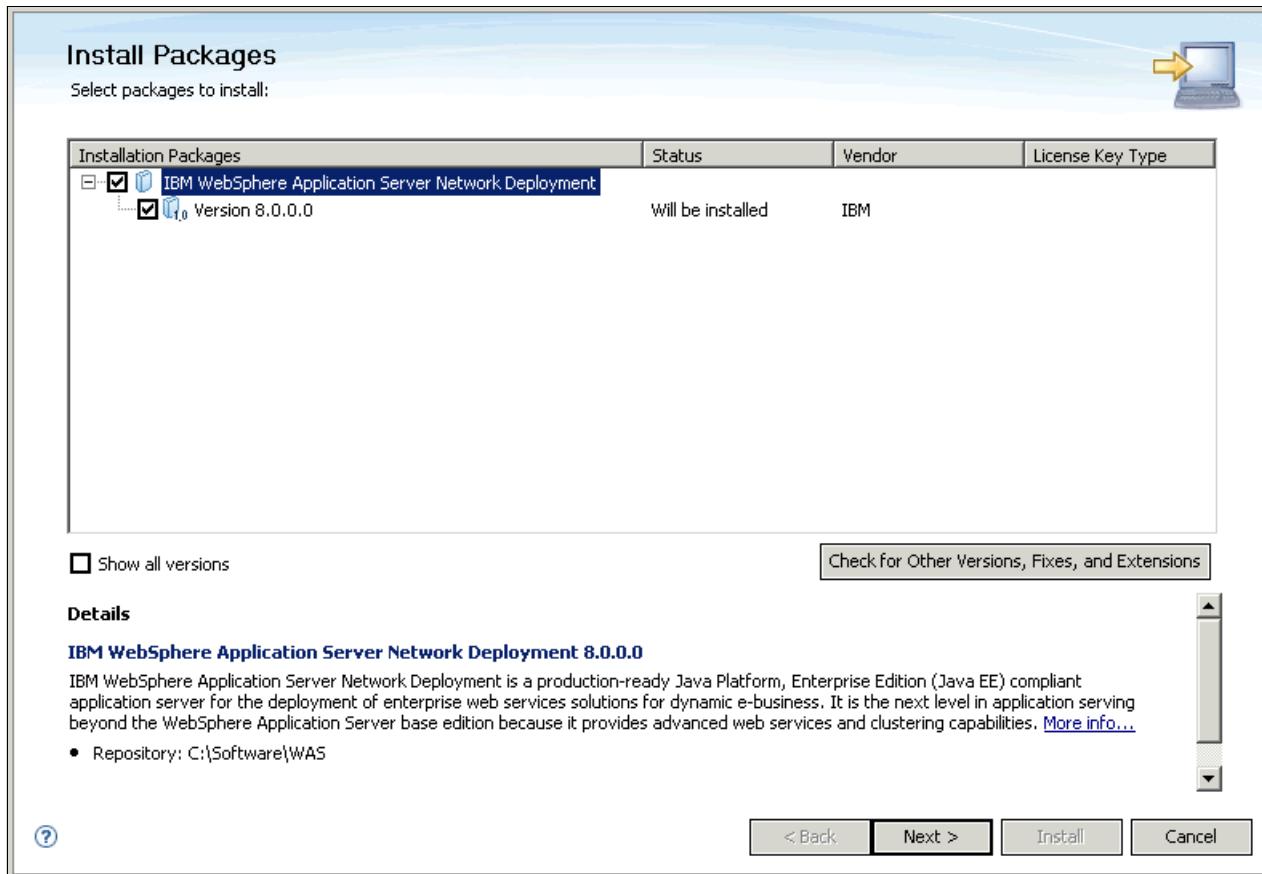


Figure 2-13 Install Packages window

Note: If WebSphere Application Server is already installed on the machine, the status indicates Installed. You can install another instance of WebSphere Application Server on the same machine. When you select the package, a message appears indicating the package is already installed. Click **Continue** to install the package. The package must be installed to a new package group.

5. In the Licenses window, you can read the license agreement. Click **I accept the terms of the license agreement** and then click **Next**.

6. In the Location window (Figure 2-14), you must provide a directory location for shared resources. The shared resources directory is used by multiple packages and is configured only during the first product package installation. You cannot change this directory after the package has been installed. Keep the default directory or modify it to suit your environment and click **Next**.

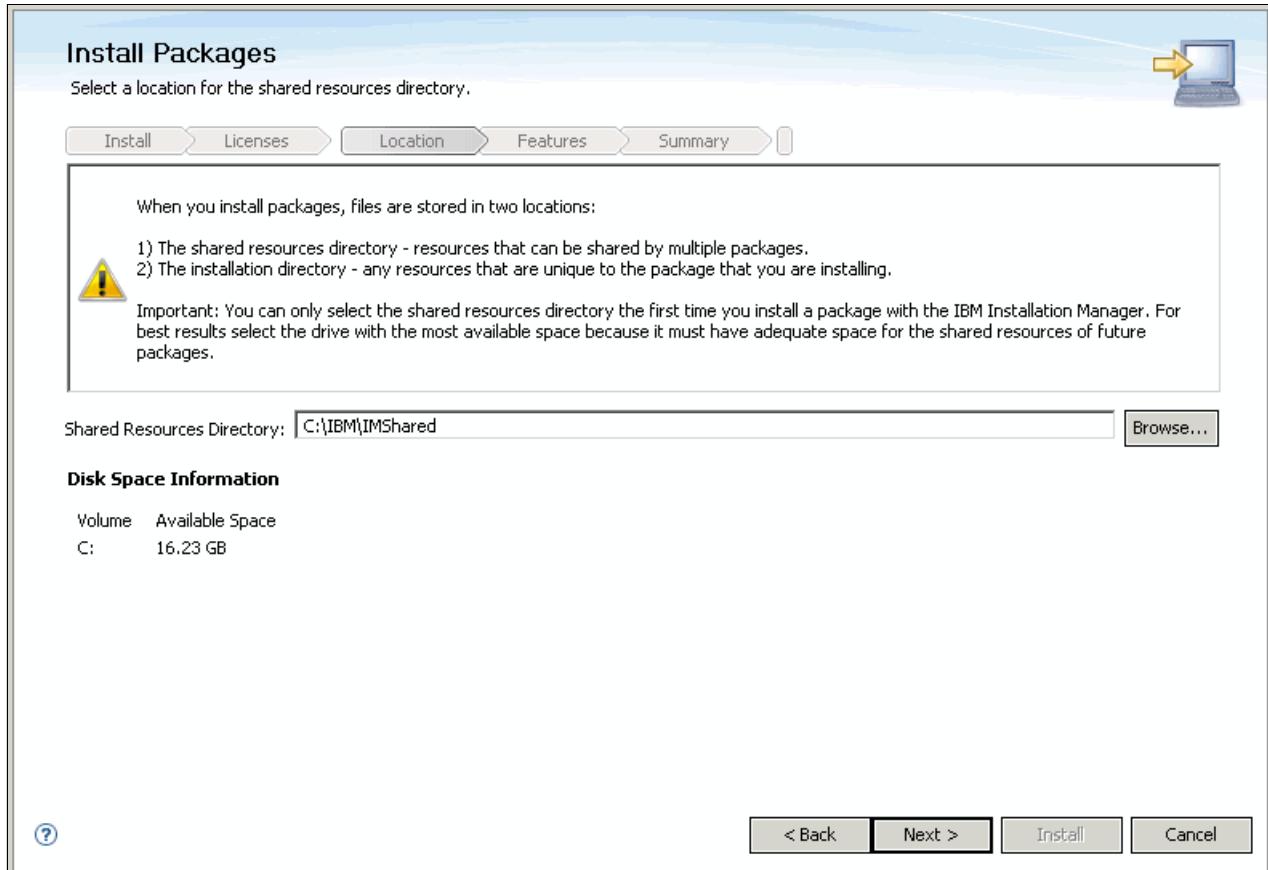


Figure 2-14 Shared resources directory location

7. In the Location window (Figure 2-15), the installation directory is entered. Enter a WebSphere Application Server Installation directory, or keep the default directory. The default directory location differs depending upon the operating system of the machine. In this example, the directory location has been modified. Click **Next**.

Note: If a non-administrator installs WebSphere Application Server V8 on a Windows Vista, Windows 7, or Windows Server 2008 operating system into the Program Files or Program Files (x86) directory with User Account Control (UAC) enabled, WebSphere Application Server will not function correctly.

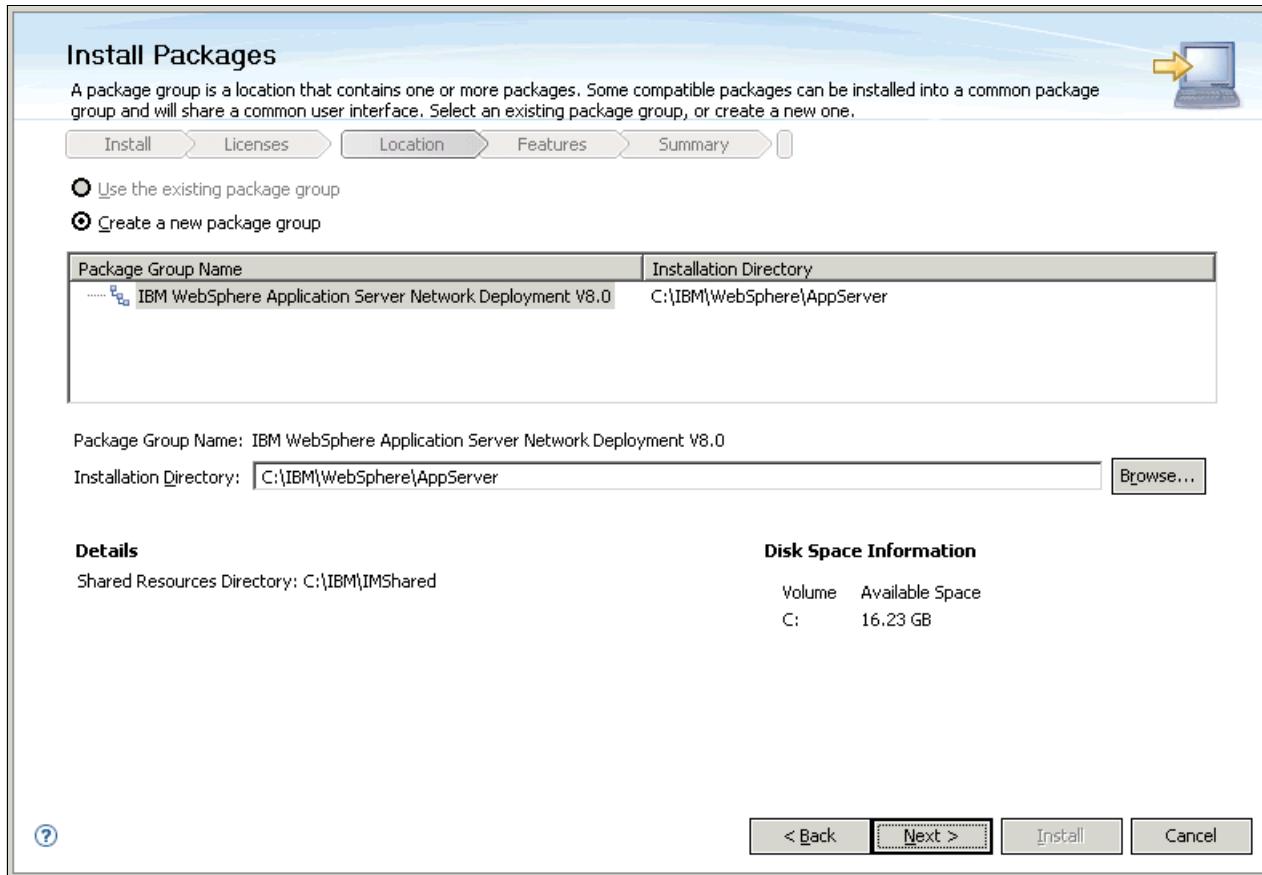


Figure 2-15 Location directory window

8. In the Features window (Figure 2-16), select the language translation to install. This action indicates the individual language packs for the WebSphere Application Server runtime environment and administrative console. English is selected by default. Click **Next**.

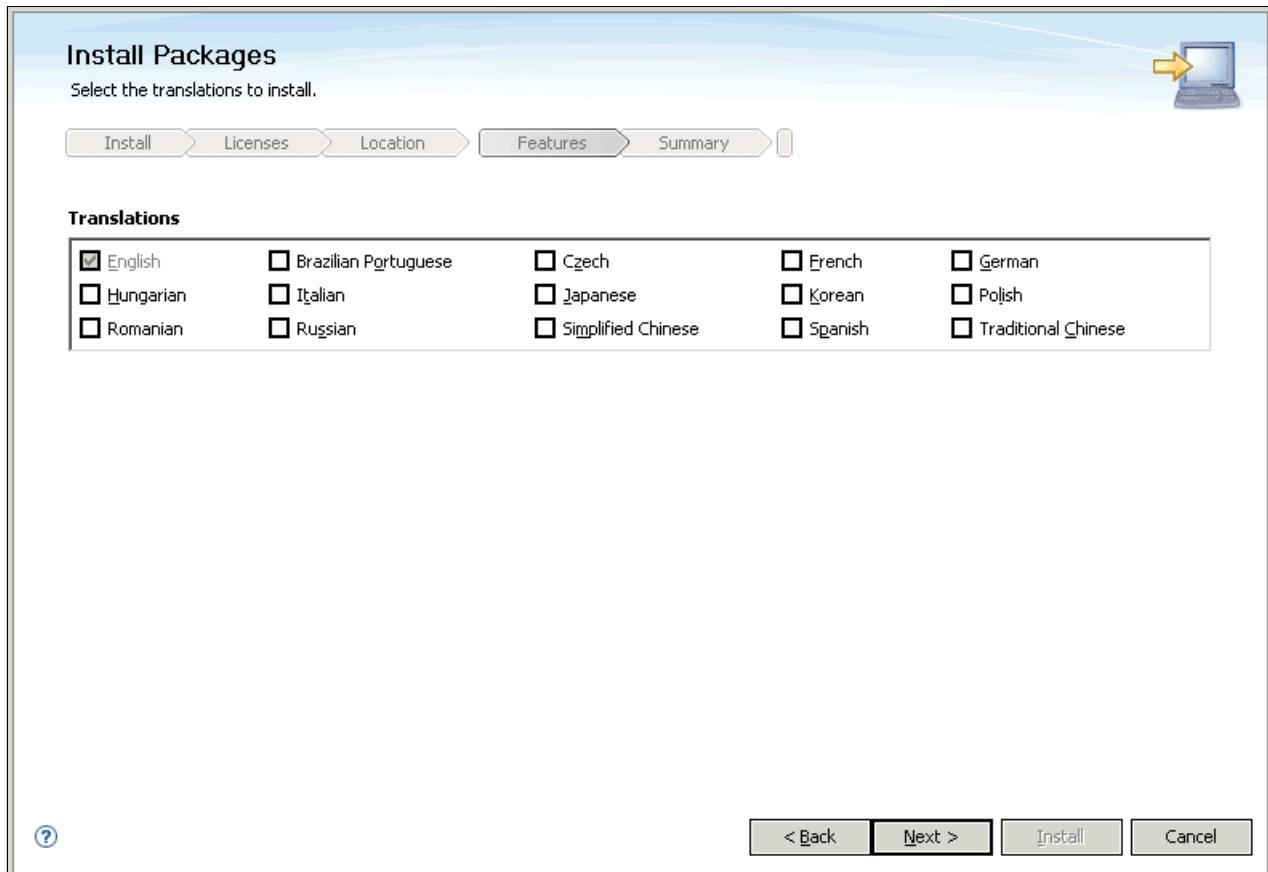


Figure 2-16 Features window

9. In the Features window, a listing of all the features that can be installed are listed. You can select the features you want to install.

New in Version 8: The EBJDeploy tool for pre-EJB modules is no longer automatically installed, it is an optionally installable feature.

Another optionally installable feature is the sample application. In Version 8, the only sample application shipped with the product is PlantsByWebSphere (PBW), and it has been updated to use Java Platform, Enterprise Edition (JEE 6) technology. You can select to deploy the sample application now or modify the installation later to add the sample application. You can no longer deploy samples during profile creation. All previous sample applications included in Version 7 that were still relevant, as well as several new samples, have been placed online for download at the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.samples.doc/info/ae/ae/welcome_samples.html

In the Features window, the Software Development Kit (SDK) feature only shows up when installing on a 64-bit operating system. Either the IBM 32-bit SDK for Java Version 6 or IBM 64-bit SDK for Java Version 6 feature must be selected. After this feature is installed, it cannot be modified.

If the installation is being performed on a 32-bit operating system, this feature choice between IBM 32-bit SDK for Java Version 6 or IBM 64-bit SDK for Java Version 6 is not available, and the installation will automatically default to the IBM 32-bit SDK for Java Version 6. In WebSphere Application Server V7, there are separate installable packages for 32-bit WebSphere Application Server installable images and 64-bit WebSphere Application Server installable images. In Version 8, they have been combined into one single installable package.

Note: This option does not apply to Solaris x86 64-bit systems.

By selecting the IBM 32-bit SDK for Java Version 6 feature in this example, it is equivalent to installing a 32-bit WebSphere Application Server on a 64-bit operating system.

Select the features to install and click **Next**. See Figure 2-17.

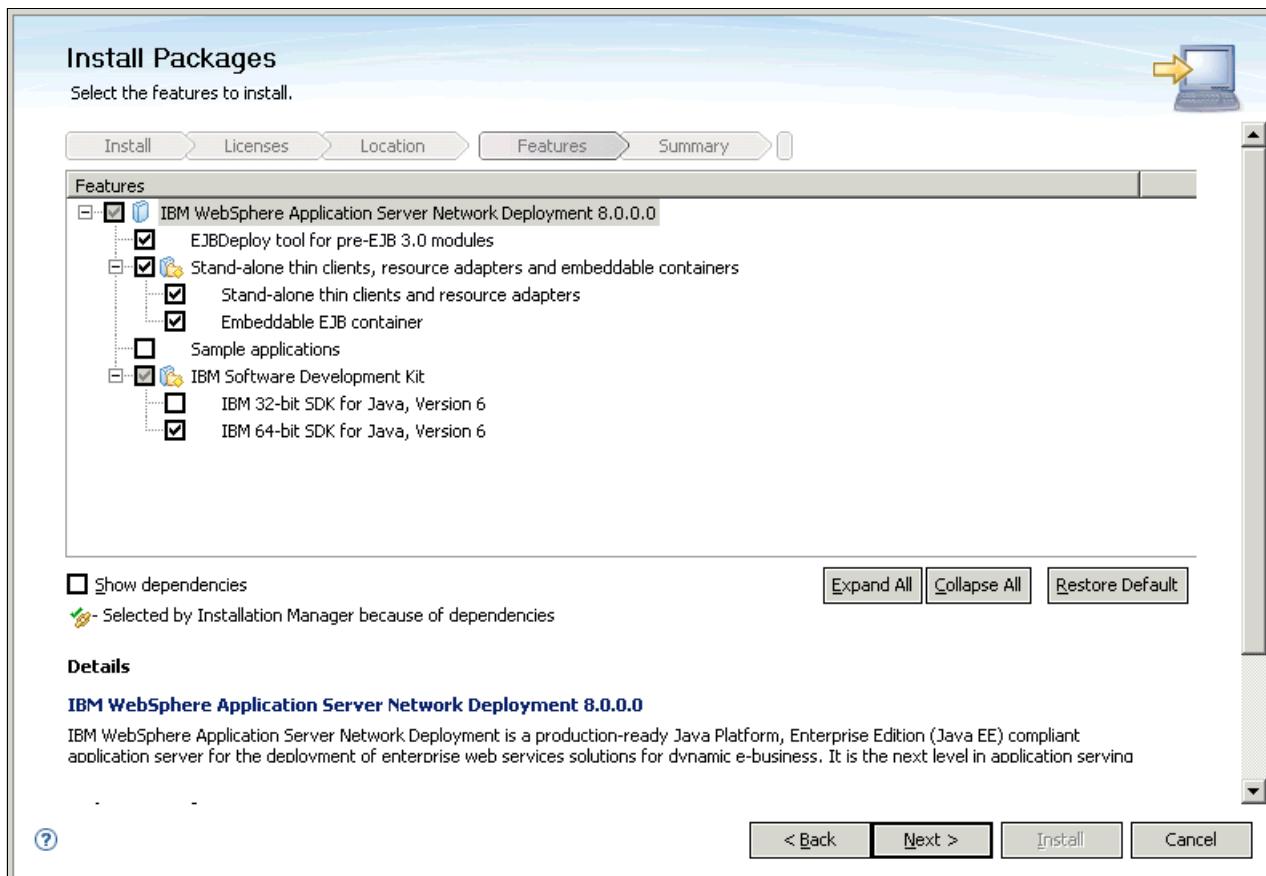


Figure 2-17 Features selection window

10. In the Summary window (Figure 2-18), review the summary information and click **Install**.

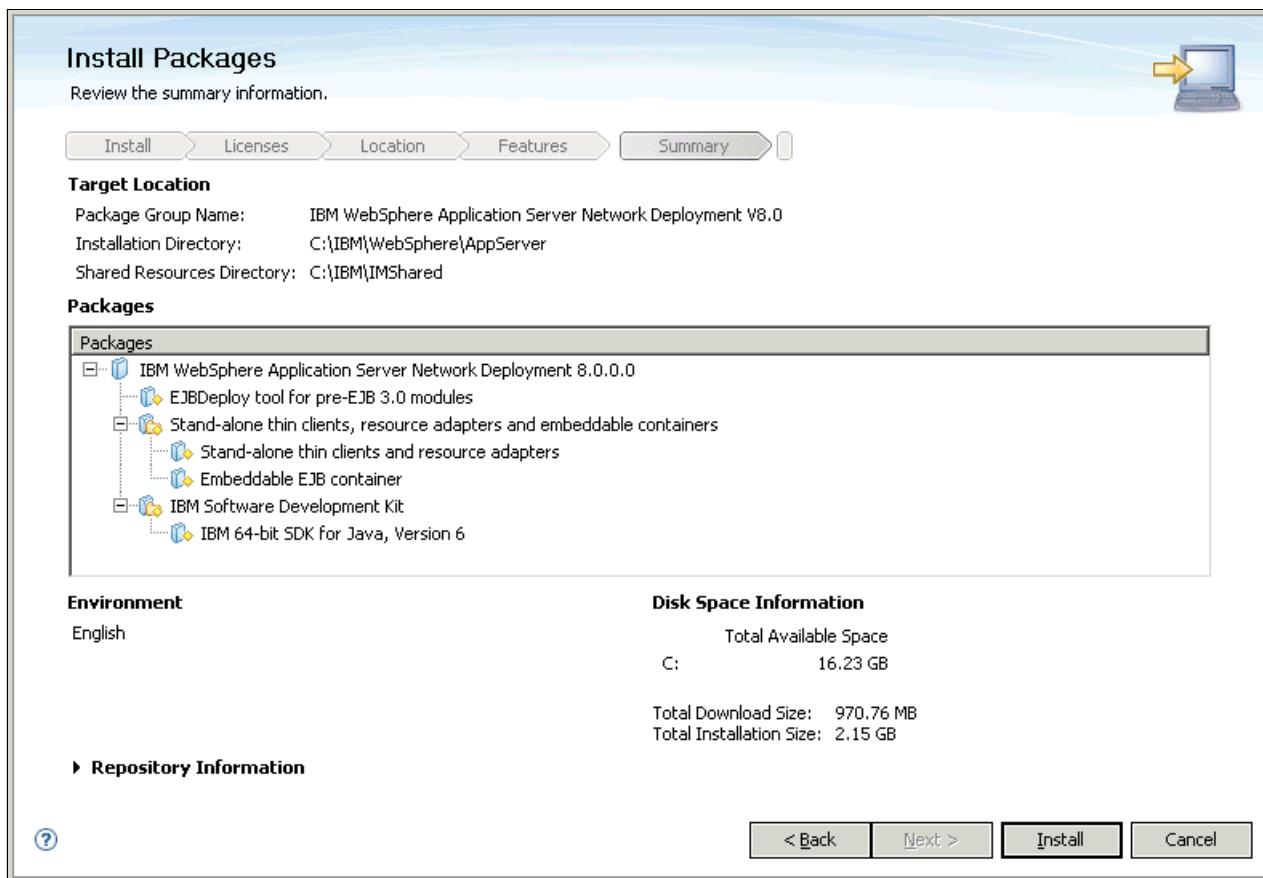


Figure 2-18 Install packages summary

11. During the installation process, you can observe the progress of the installation. At any time, you can select to pause or cancel the installation. See Figure 2-19.

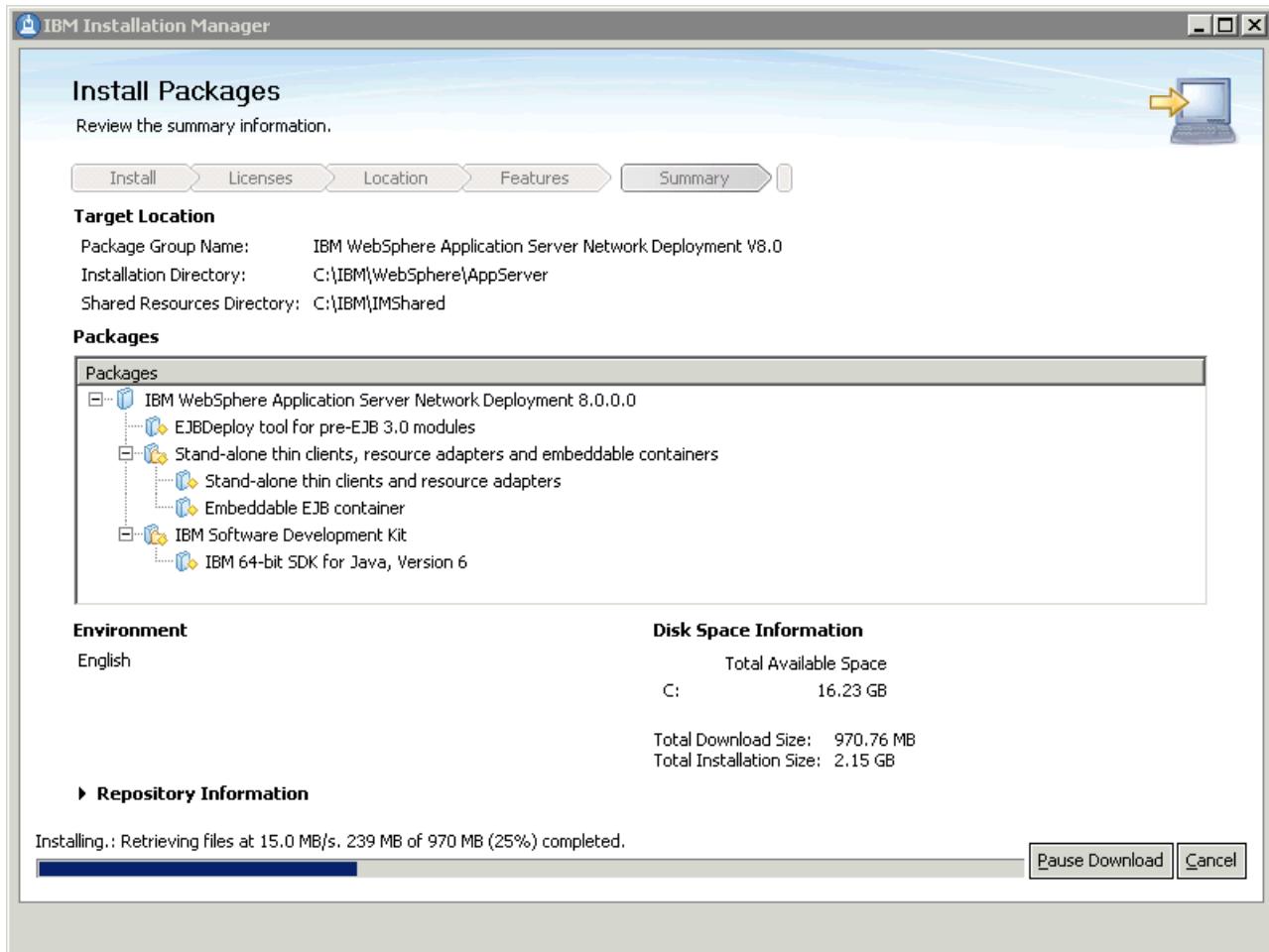


Figure 2-19 Installation progress

12. When the installation completes, the installation results are displayed. You can review the installation log file to troubleshoot any problems that you may have by selecting the **View Log File** link.

From the summary window, you can start another tool or exit the installation process. You have the following options:

- Profile Management Tool, to create a profile that allows you to create a new profile using the Profile Management Tool.
- Profile Management Tool, to create an application server profile for a development environment that allows you to create a new application server profile with settings for a development environment. If the application server will be used primarily for development purposes, select this option to create it from the development template. The development template reduces startup time and allows the server to run on less powerful hardware. Do not use this for production servers.
- None, which indicates that you do not want to create a profile at this time.

Select an option and click **Finish**. See Figure 2-20.

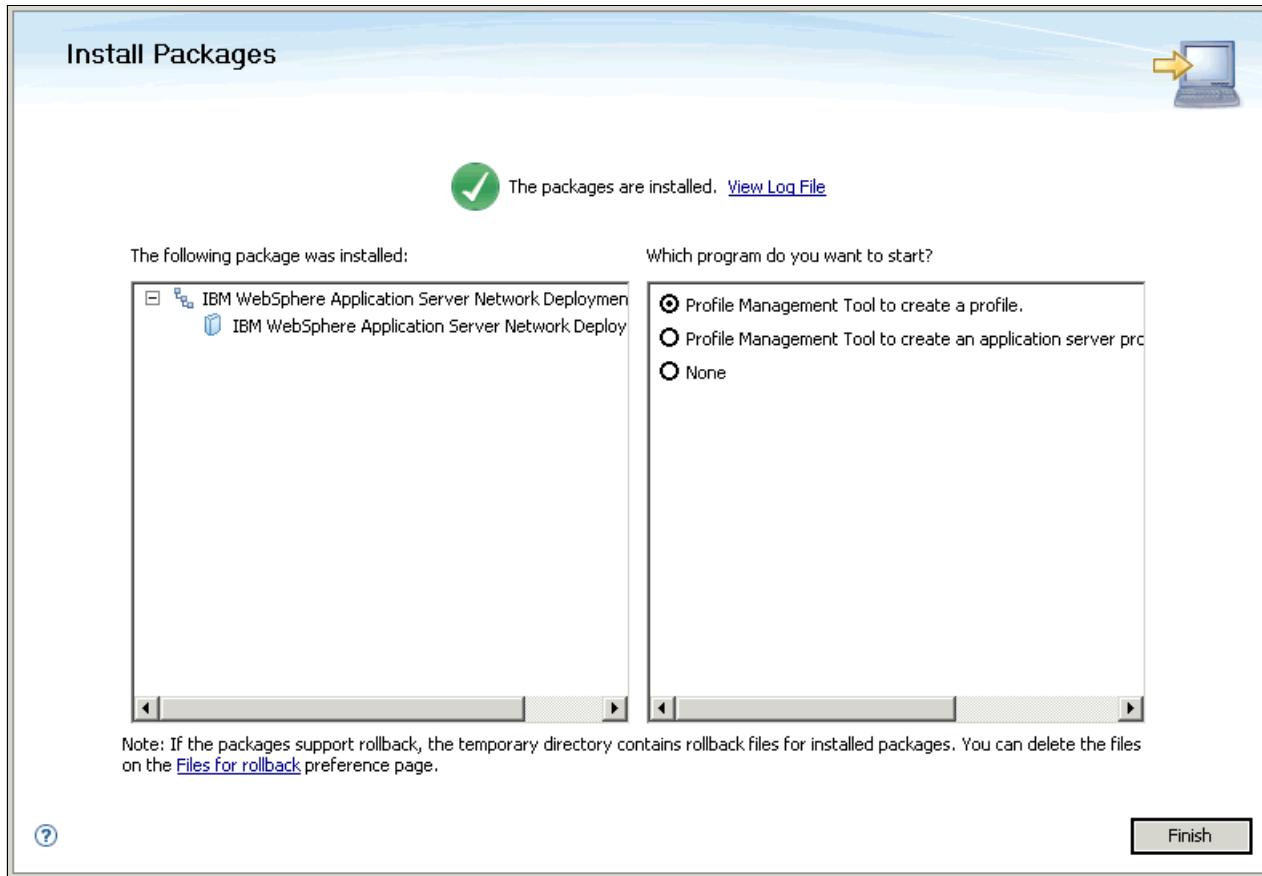


Figure 2-20 Installation summary window

2.5.3 Creating a response file

In WebSphere Application Server V8, the recommended method for creating a response file is using the Installation Manager GUI with the **-skipInstall** option.

Complete the following steps to create an installation response file for WebSphere Application Server using the command-line mode:

1. Install Installation Manager.
2. Navigate to the <IM_install>\eclipse\tools directory.
3. Run **imcl listAvailablePackages**, pointing to the WebSphere Application Server repository to obtain the package name. See Example 2-9.

Example 2-9 Listing available packages

```
C:\IBM\InstallationManager\eclipse\tools>imcl.exe listAvailablePackages  
-repositories C:\temp\software\WebSphere\repository.conf  
com.bm.websphere.ND.v80.8.0.0.20110503_0200
```

You can also run **imcl listInstalledPackages** to verify the package has not already been installed.

4. Run the command to create a response file using Installation Manager GUI, including the **-skipInstall** option. Because this is the first product packaged installed by the Installation Manager, the Shared Resources Directory is also identified. See Example 2-10.

Example 2-10 Command-line response file creation

```
C:\IBM\InstallationManager\eclipse>IBMIM.exe -skipInstall C:\temp\imRegistry  
-record C:\temp\websphere_silent_install_reponse_file.xml
```

- The Installation Manager GUI is launched. Add a repository for the WebSphere Application Server package. See 2.4.3, “Repository configuration” on page 52 for details about adding a repository.
- Click the **Install** wizard to begin the installation.
- The Installation Manager searches the defined repositories and lists the packages that it finds. In the Install window, the WebSphere Application Server package is listed. Select the package and the status indicates the package will be installed. Click **Next**.
- Read and accept the license agreement. Click **Next**.
- Enter the shared resources directory location and click **Next**.
- Enter the WebSphere installation directory location and click **Next**.
- Select the language translation to install and click **Next**.
- Select the packages to install and click **Next**.
- Review the settings and click **Install**.

- The final window is the summary indicating the package is installed. However, no package was installed; instead, a recording of the installation was created. You can see at the top of the Installation Manager GUI that it indicates Recording. See Figure 2-21.

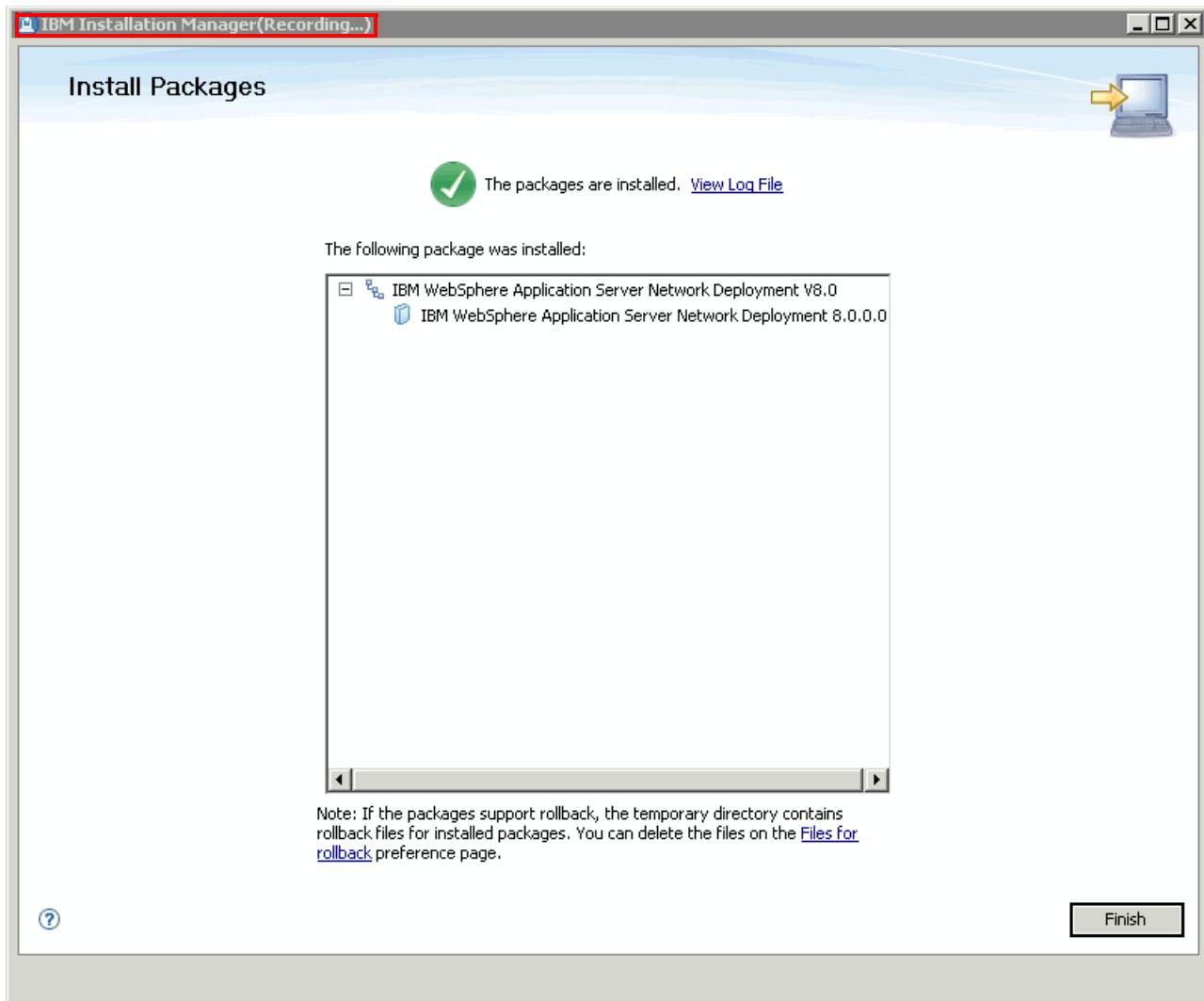


Figure 2-21 Installation recording summary

You can view the log file or click **Finish**.

- The Installation Manager is still in recording mode, so exit the GUI. Click **File** → **Exit**.
- Examine the created response file and modify it if needed. Example 2-11 shows the contents of the response file that was just created.

Example 2-11 WebSphere installation response file

```
<?xml version="1.0" encoding="UTF-8"?>
<agent-input>
<server>
    <repository location='C:\temp\Software\WAS' />
</server>
<profile id='IBM WebSphere Application Server Network Deployment V8.0'
    installLocation='C:\IBM\WebSphere\AppServer'>
    <data key='eclipseLocation' Value='C:\IBM\WebSphere\AppServer' />
```

```

        <data key='user.import.profile' value='false'/>
</profile>
<install modify='false'>
    <offering id='com.ibm.comwebsphere.ND.v80' version='8.0.0.20110503_0200'
    profile="IBM WebSphere Application Server Network Deployment V8.0"
    features="core.feature,ejbdeploy,thinclient,embeddablecontainer,com.ibm.sdk.6_6
    4bit"
    installFixes="none"/>
</install>
<preference name="com.ibm.cic.common.core.preferences.eclipseCache"
value="C:\IBM\IMShared"/>
<preference name="com.ibm.cic.common.core.preferences.connectTimeout"
value="30"/>
<preference name="com.ibm.cic.common.core.preferences.readTimeout" value="45"/>
<preference name="com.ibm.cic.common.core.preferences.downloadAutoRetryCount"
value="0"/>
<preference name="offering.service.repositories.areUsed" value="true"/>
<preference name="com.ibm.cic.common.core.preferences.ssl.nonsecureMode"
value="false"/>
<preference
name="com.ibm.cic.common.core.preferences.http.disablePreemptiveAuthentication"
value="false"/>
<preference name="http.ntlm.auth.kind" value="NTLM"/>
<preference name="http.ntlm.auth.enableIntegrated.win32" value="true"/>
<preference
name="com.ibm.cic.common.core.preferences.preserveDownloadedArtifacts"
value="true"/>
<preference name="com.ibm.cic.common.core.preferences.keepFetchedFiles"
value="false"/>
<preference name="PassportAdvantageEnabled" value="false"/>
<preference name="com.ibm.cic.common.core.preferences.searchForUpdates"
value="false:/>
<preference name="com.ibm.agent.ui.displayInternalVersion" value="false"/>

```

2.5.4 Installing silently

Complete the following steps to install WebSphere Application Server silently through a response file using the command-line mode:

1. Run the silent installation command, indicating the response file. See Example 2-12.

Example 2-12 Silent installation

```
C:\IBM\InstallationManager\eclipse\tools>imcl.exe -acceptLicence input
C:\temp\websphere_silent_install_reponse_file.xml -log
C:\temp\install_nd_log_file.xml
```

You can include the **-showProgress** argument to see the progress in the command line for the silent installation.

- Run `imcl listInstalledPackages` to verify that the package has been installed. See Example 2-13.

Example 2-13 Listing installed package

```
C:\IBM\InstallationManager\eclipse\tools>imcl.exe listInstalledPackages
com.ibm.cic.agent_1.4.3001.20110504_1325
com.ibm.websphere.ND.v80_8.0.0.20110503_200
```

- Examine the installation log file to verify that the installation was successful. You can view the log file using a text editor, browser, or using the Installation Manager GUI. To use the Installation Manager GUI, click **File → View Log**. Select the log file in the page to view the log file.

Note: Do not use the same response files that are used with WebSphere Application Server V7 or earlier to install or uninstall V8 silently. Use the response files that are based on Installation Manager to install, update, or uninstall WebSphere Application Server V8.

2.6 WebSphere Customization Toolbox

New in Version 8: The WebSphere Customization Toolbox for WebSphere Application Server V8 includes tools for managing, configuring, and migrating various parts of your WebSphere Application Server environment. It is an Eclipse framework application that existed in Version 7, but had a vastly smaller function (it was used only to configure z/OS servers). WebSphere Customization Toolbox is a container framework that holds various tools that are used for configuring your application server environment.

WebSphere Customization Toolbox is available as two different offerings, and each offering has various combinations of tools on different platforms. The two offerings are as follows:

- ▶ Embedded
- ▶ Stand-alone

Each offering for WebSphere Customization Toolbox is installed, modified, rolled back, and updated using the Installation Manager. WebSphere Customization Toolbox can be installed silently, using the command line, or interactively using the GUI or console mode.

2.6.1 Embedded WebSphere Customization Toolbox

Embedded WebSphere Customization Toolbox comes as a part of the WebSphere Application Server V8 package. It is installed on all platforms wherever WebSphere Application Server V8 is installed and supported.

Tools included in the embedded WebSphere Customization Toolbox are as follows:

- ▶ Profile Management Tool (PMT)
- ▶ Configuration Migration Tool (CMT)

When installing the embedded WebSphere Customization Toolbox offering, both tools are automatically installed. The tools are not listed for selection in the Installation Manager.

New in Version 8: WebSphere Application Server V8 has added GUI support on IBM AIX® 64-bit systems.

Note: AIX requires the GNU Toolkit (GTK) to be installed to run the GUI. If you do not have the GTK installed, you receive an Eclipse error when trying to launch the GUI. For details about configuring AIX to support the GUI, refer to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.installation.nd.doc/info/ae/ae/tins_aixsetup.html

Supported platforms include HPUX 64-bit, Windows 2K (32-bit and 64-bit), Linux on x86 (32-bit and 64-bit), Linux on Power (32-bit and 64-bit), Linux on s390 (32-bit and 64-bit), Solaris Sparc, and AIX (32-bit and 64-bit).

2.6.2 Stand-alone WebSphere Customization Toolbox

Stand-alone WebSphere Customization Toolbox comes as its own product offering. It can be found in the WebSphere Applications Server V8 supplements package and is installed using the Installation Manager.

Tools included in the embedded WebSphere Customization Toolbox are as follows:

- ▶ Web Server Plug-ins Configuration Tool (PCT)
- ▶ Remote Installation Tool for IBM i
- ▶ z/OS Profile Management Tool (zPMT)
- ▶ z/OS Migration Management Tool (zMKT)

When installing the stand-alone WebSphere Customization Toolbox offering, you select the tools you want to install. However, the zPMT and zMMT have dependencies that are recognized by the Installation Manager and must be installed together.

Supported platforms include HPUX, Windows, Linux on x86, Linux on Power, Linux on s390, Solaris (Sparc and x86), and AIX. Regardless of a 32-bit or 64-bit operating system, stand-alone WebSphere Customization Toolbox operates as a 32-bit component on a 32-bit JDK.

2.6.3 Overview of the tools in the WebSphere Customization Toolbox offerings

The tools in the WebSphere Customization Toolbox offerings include:

- ▶ Profile Management Tool (PMT)
The Profile Management Tool provides a user interface for profile creation and augmentation.
- ▶ Configuration Migration Tool (CMT)
The Configuration Migration Tool provides a graphical interface to the migration tools included in WebSphere Application Server.

- ▶ Web Server Plug-ins Configuration Tool (PCT)

The Web Server Plug-ins Configuration Tool is a new tool that allows you to configure your web server plug-ins on distributed operating systems for communicating with the application server. If possible, it creates a web server configuration definition in the application server.

- ▶ Remote Installation Tool for IBM i

The Remote Installation Tool for IBM i can be used on an Intel-based operating system only to install Installation Manager or a WebSphere Application Server component from a Windows machine to a remote target IBM i system.

- ▶ z/OS Profile Management Tool (zPMT)

The z/OS Profile Management Tool can be used on an Intel-based or Linux operating system to generate jobs and instructions for creating profiles for WebSphere Application Server on z/OS systems. The jobs are then uploaded and run on a target z/OS system.

- ▶ z/OS Migration Management Tool (zMMT)

The z/OS Migration Management Tool can be used on an Intel-based or Linux operating system to create migration definitions that are used to migrate a WebSphere Application Server on z/OS node. Each migration definition is a set of jobs and instructions that can then be uploaded and run on a target z/OS system.

To discover more about the Profile Management Tool, refer to Chapter 3, “Working with profiles on distributed systems” on page 77.

To discover more about the Web Server Plug-ins Tool, refer to Chapter 12, “Configuring and managing web servers” on page 443.

To discover more about the z/OS Profile Management Tool, refer to Chapter 4, “Installing WebSphere Application Server on z/OS systems” on page 141.

2.6.4 Installing the stand-alone WebSphere Customization Toolbox

To install the stand-alone WebSphere Customization Toolbox offering, you first must have Installation Manager installed. Also, make sure the Installation Manager preferences points to a repository containing WebSphere Customization Toolbox. You can install WebSphere Customization Toolbox using the GUI, command-line, or silent installation methods. You can also create a response file using the Installation Manager and use this file to install WebSphere Customization Toolbox.

Note: WebSphere Application Server V8 does not support console mode installation.

The installation process for WebSphere Customization Toolbox is similar to installation of other packages using Installation Manager. However, you have the option of selecting the tools you want to install in the stand-alone offering. If you do not install all the available tools during the initial installation process, you can modify the installation to add any additional tools.

Figure 2-22 shows the Install Packages window where you can select individual tools for installation.

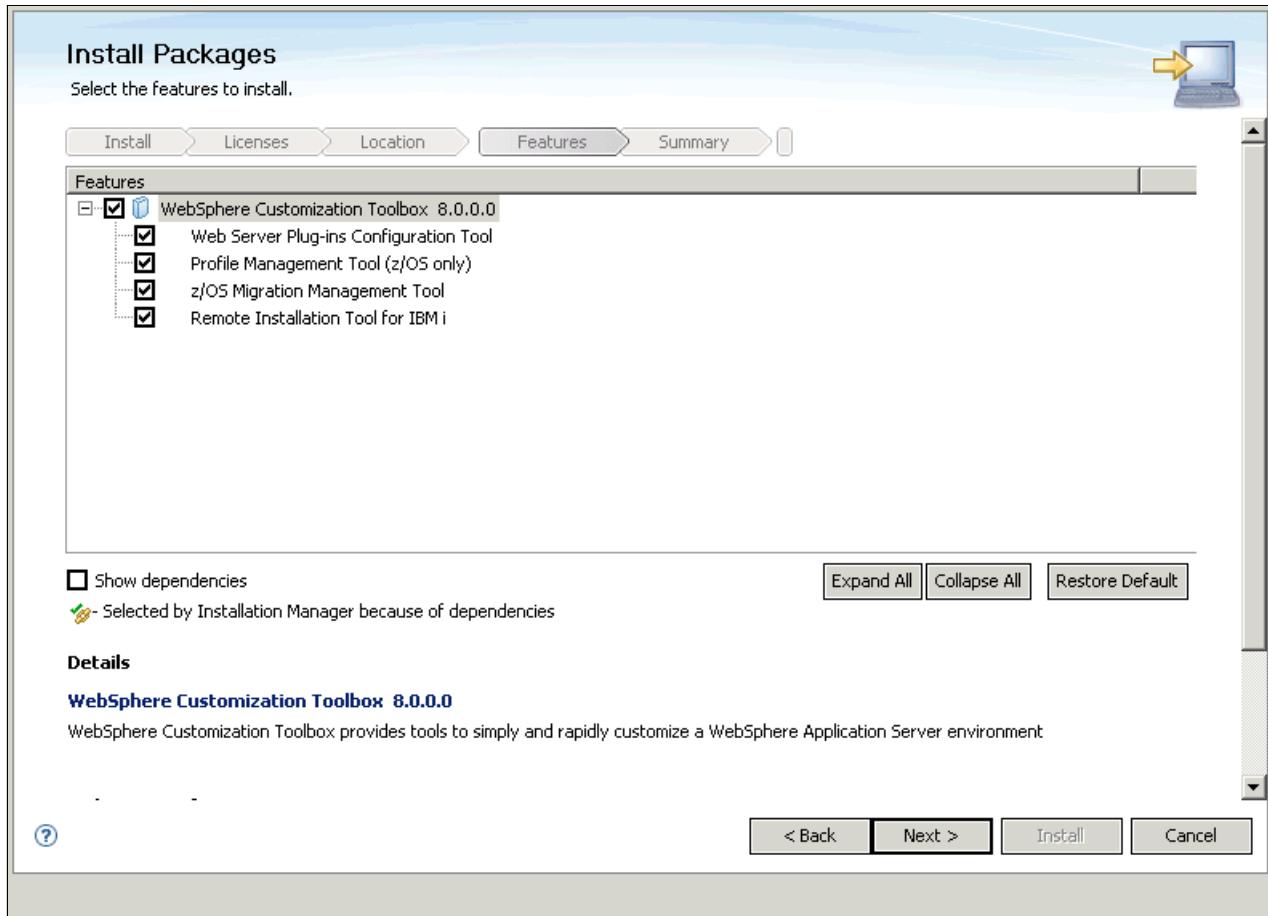


Figure 2-22 Install Packages window

If you install the z/OS Profile Management Tool, you must also install the z/OS Migration Management Tool. There is a dependency between these two tools; the Installation Manager recognizes this situation and installs both tools automatically. Click the **Show dependencies** option to see a listing of dependencies for a particular tool. See Figure 2-23.

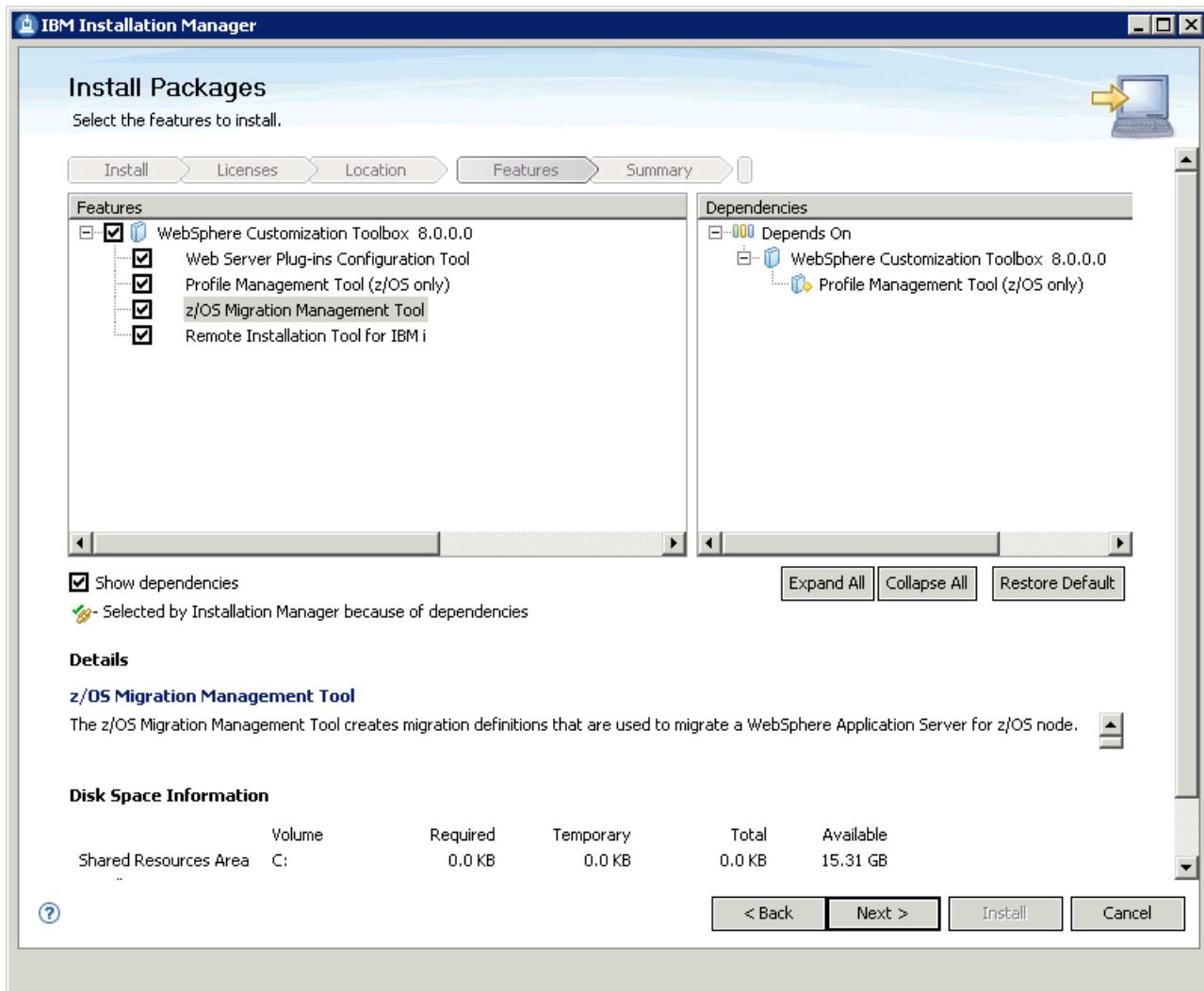


Figure 2-23 Package dependencies

2.6.5 Starting the WebSphere Customization Toolbox

There are a few ways you can start the WebSphere Customization Toolbox, which depends on the WebSphere Customization Toolbox offering being used and the operating system. To start the WebSphere Customization Toolbox, use one of the following methods:

- ▶ On Windows and Linux platforms, a start menu shortcut is created for both the embedded and stand-alone offering.
- ▶ Launch `wct` from `<was_install>\bin\ProfileManagement\WCT\` for the embedded offering.
- ▶ Launch `wct` from `<wct_install>\WCT\` for the stand-alone offering.

New in Version 8: The `pmt` command is still available for backward compatibility. However, it has been deprecated in WebSphere Application Server V8.

Figure 2-24 shows the embedded WebSphere Customization Toolbox GUI.

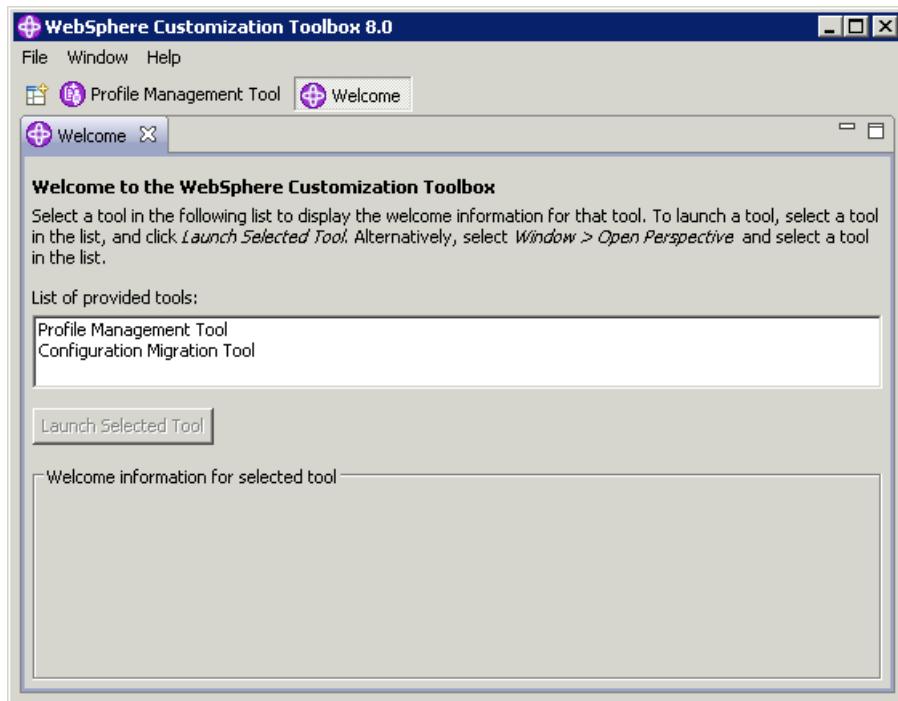


Figure 2-24 Embedded WebSphere Customization Toolbox

Figure 2-25 shows the stand-alone WebSphere Customization Toolbox GUI.

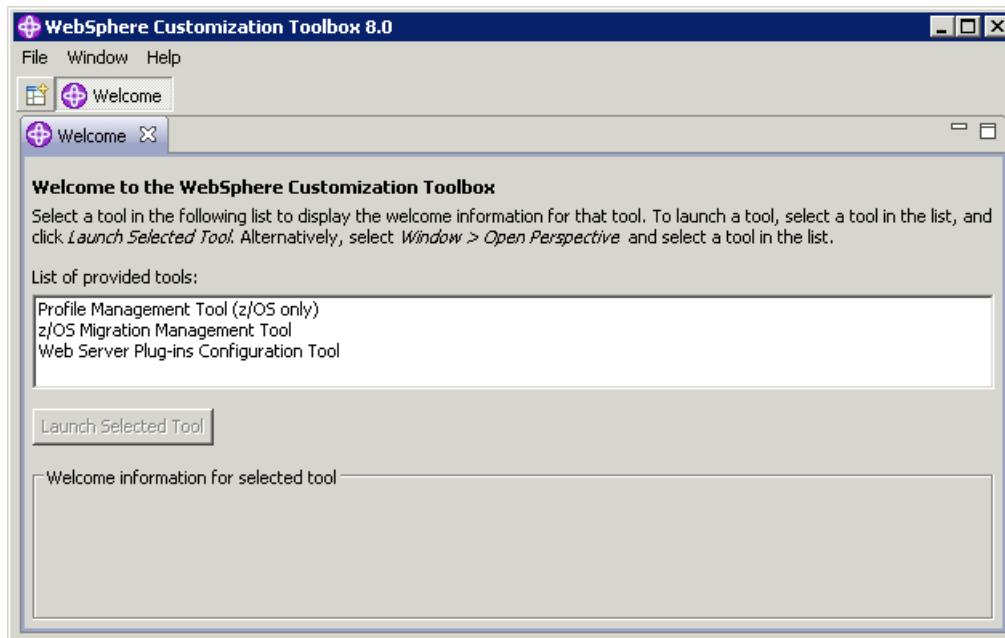


Figure 2-25 Stand-alone WebSphere Customization Toolbox

WebSphere Customization Toolbox maintains a repository to store its metadata. The repository location is tied to the user ID used to launch WebSphere Customization Toolbox. Multiple launches of WebSphere Customization Toolbox are allowed as long as workspaces do not collide, which means they must be different user IDs. Therefore, two different users can launch WebSphere Customization Toolbox from the same installation location without conflict.

Repository information can be located at the following directories:

- ▶ Windows: C:\Documents and Settings\Administrator\AppData\Local\IBM\WebSphere
- ▶ UNIX: /root/.ibm/WebSphere/AppServer

2.6.6 WebSphere Customization Toolbox command line tool

The stand-alone WebSphere Customization Toolbox offering also includes a command-line utility that launches a command line version of the Web Server Plug-ins Configuration Tool (PCT).

The syntax for the **wct** command-line tool is shown in Example 2-14.

Example 2-14 Command-line tool

Function:

Invokes the WebSphere Customization Toolbox command line tool specified by the **-tool** parameter.

Syntax:

```
wctcmd -tool <toolId> -<argument> <argument parameter> ...
wctcmd -tool <toolId> -<wctcmd specific argument> <argument parameter>
    -response <response file containing tool arguments>
```

Arguments:

The following command line arguments are required:

- tool** <argument parameter>: The name of the tool to launch, as it is registered with the WCT command line framework
- defLocName** <argument parameter>: The name of the definition location as it resides in the definition location registry.
- defLocPathname** <argument parameter>: The absolute path name of the definition location to use when the specified tool is launched.

The following command line arguments are optional:

- createDefinitionLocations** <argument parameter>: Indicates that the WebSphere Customization Toolbox command line should create a definition location.
- importDefinitionLocation** <argument parameter>: Indicates that the WebSphere Customization Toolbox command line should import a definition location.
- removeDefinitionLocation** <argument parameter>: Indicates that the WebSphere Customization Toolbox command line should remove a definition location from the definition list.
- listDefinitionLocations** <argument parameter>: Indicates that the WebSphere Customization Toolbox command line should list the existing definition locations in the definition location list.
- defLocVersion** <argument parameter>: The version of the definition location to create

Note: command line arguments are case sensitive.

Note: If argument accepts a parameter containing spaces, the parameter must be enclosed in “double quotes”.



Working with profiles on distributed systems

Installing a WebSphere Application Server environment requires careful planning. A major decision point is the topology for the system. Decisions include, for example, whether you will have a stand-alone server, a distributed managed server environment, and whether you will use the flexible management options.

Planning for topology design is covered in *IBM WebSphere Application Server V8 Concepts, Planning, and Design Guide*, SG24-7957. That book is designed to help you select a topology and develop a clear idea of what steps are needed to set up your chosen environment. Your options depend on your WebSphere Application Server edition. The installation process is well documented in the installation guide packaged with the product.

The purpose of this chapter is to help you build your initial WebSphere Application Server environment after you have installed the product. In this chapter, we cover the following topics:

- ▶ Types of profiles
- ▶ Planning for profiles
- ▶ Building systems with profiles
- ▶ Managing profiles

3.1 Types of profiles

The WebSphere Application Server installation process simply lays down a set of core product files required for the runtime processes. After installation, you need to create one or more *profiles* that define the run time to have a functional system. The core product files are shared among the runtime components defined by these profiles.

With the Base and Express packages, you can have stand-alone application servers. Each application server is defined within a single cell and node. The administration console is hosted within the application server and can only connect to that application server. You can consolidate administration for multiple stand-alone servers by registering the node for each application server to an administrative agent.

The application server profile defines the stand-alone environment. You can also create stand-alone application servers with the Network Deployment package, though you would most likely do so with the intent of federating that server into a cell for central management.

With the Network Deployment package, you have the option of defining multiple application servers with central management capabilities. The administration domain is the cell, consisting of one or more nodes. Each node contains one or more application servers and a node agent that provides an administration point management by the deployment manager.

The deployment manager can be located on the same machine as one or more of the application servers: a common topology for single machine development and testing environments. In most production topologies, the deployment manager should be placed on a separate dedicated machine.

The basis for this runtime environment starts with the deployment manager that provides the administration interface for the cell. As you would expect, the deployment manager is defined by a deployment manager profile.

Nodes can be added to the cell in one of two ways:

- ▶ You can create an application server profile, then federate it to the cell. When a node is added to a cell, a node agent is created on the node, and configuration files for the node are added to the master configuration repository for the cell. The deployment manager then assumes responsibility for the configuration of all servers on the node.
- ▶ You can define a custom profile to create an empty node for federation to the cell. After federation, you further configure the node by creating application servers and clusters from the deployment manager administrative console.

The job manager and administrative agent profile types can be created to enhance the administration capabilities.

3.1.1 Application server profile

The application server profile defines a single stand-alone application server. Using this profile gives you an application server that can run stand-alone, or unmanaged. The environment has the following characteristics:

- ▶ The profile consists of one cell, one node, and one server. The cell and node are not relevant in terms of administration, but you see them when you administer the server through the administrative console scopes.
- ▶ The PlantsByWebSphere sample application is installed on the server (optional).
- ▶ The server has a dedicated administrative console.

The primary uses for this type of profile are:

- ▶ To build a stand-alone server in a Base or Express installation.
- ▶ To build a stand-alone server in a Network Deployment installation that is not managed by the deployment manager (a test machine, for example).
- ▶ To build a server in a distributed server environment to be federated and managed by the deployment manager. If you are new to WebSphere Application Server and want a quick way of getting an application server complete with samples, this is a good option. When you federate this node, the default cell becomes obsolete, the node is added to the deployment manager cell, and the administrative console is removed from the application server.

3.1.2 Deployment manager profile

The deployment manager profile defines a deployment manager in a distributed server environment. Although you could conceivably have the Network Deployment edition and run only stand-alone servers, this action bypasses the primary advantages of Network Deployment, which is workload management, failover, and central administration.

In a Network Deployment environment, you should create one deployment manager profile for each cell. This setup gives you:

- ▶ A cell for the administrative domain
- ▶ A node for the deployment manager
- ▶ A deployment manager with an administrative console
- ▶ No application servers

After you have the deployment manager, you can:

- ▶ Federate nodes built either from existing application server profiles or custom profiles.
- ▶ Create new application servers and clusters on the nodes from the administrative console.

3.1.3 Custom profile

A custom profile is an empty node, intended for federation to a deployment manager. This type of profile is used when you are building a distributed server environment. You use a custom profile as follows:

1. Create a deployment manager profile.
2. Create one custom profile on each node on which you will run application servers.
3. Federate each custom profile to the deployment manager, either during the custom profile creation process or later by using the **addNode** command.
4. Create new application servers and clusters on the nodes from the administrative console.

3.1.4 Cell profile

A cell profile is actually a combination of two profiles: a deployment manager profile and an application server profile. The application server profile is federated to the cell. The deployment manager and application server reside on the same system. This type of profile lets you get a quick start with a distributed server environment and is especially useful for test environments that typically have all nodes on one test system.

3.1.5 Administrative agent profile

The administrative agent profile provides enhanced management capabilities for stand-alone application servers. An administrative agent profile is created on the same node as the stand-alone servers and can manage only servers on that node. The node configuration for each stand-alone server is totally separate from any other servers on the system, but it can be managed using the administrative console on the administrative agent.

To participate in flexible management, stand-alone base servers first register themselves with the administrative agent. When a base application server registers with an administrative agent, much of the administrative code that was in the base server is consumed by the administrative agent. This action results in a significantly smaller and faster starting base server. Figure 3-1 shows this concept.

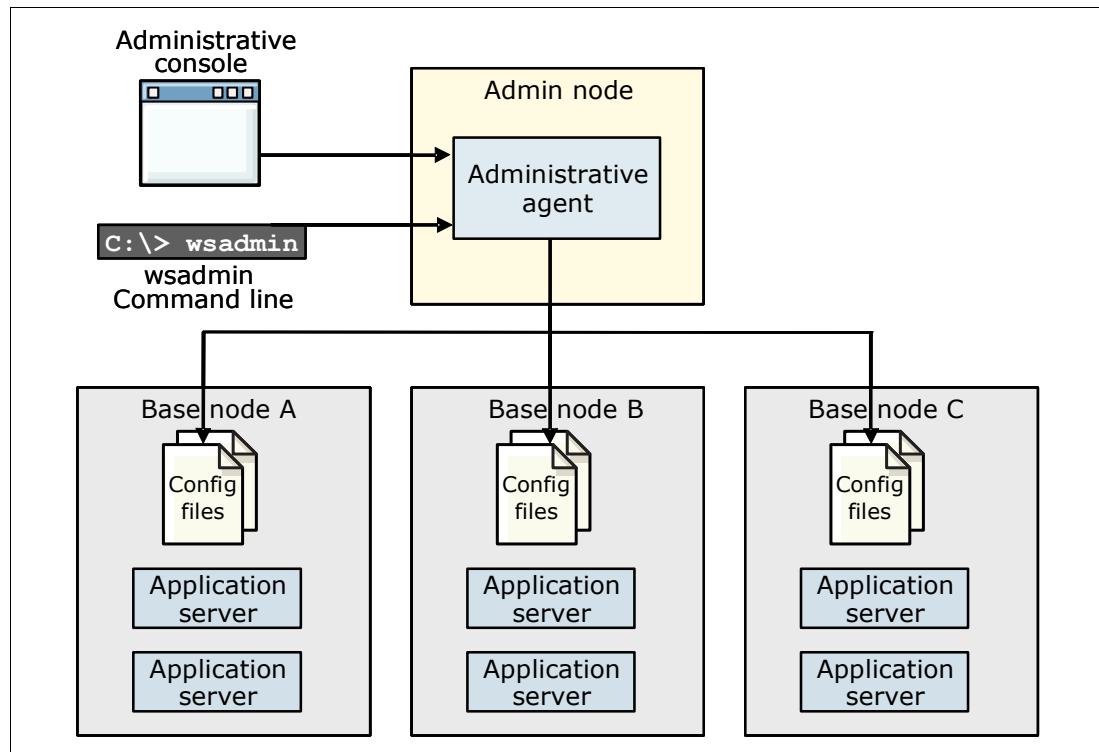


Figure 3-1 High-level overview of a administrative agent profile architecture

3.1.6 Job manager profile

The job manager is a server type that was added in WebSphere Application Server V7 to support flexible management. A job manager is defined by a job manager profile.

To participate in flexible management, a stand-alone application server first registers itself with the administrative agent. The administrative agent must then register the node for the application server with the job manager. If a deployment manager wants to participate in an environment controlled by a job manager, the deployment manager registers directly with the job manager; no administrative agent is involved in this case.

The main use of the job manager is to queue jobs to application servers in a flexible management environment. These queued jobs are pulled from the job manager by the administrative agent and distributed to the appropriate application server or servers.

Both deployment manager and administrative agents retain autonomy and can be managed without the job manager. A job manager can submit jobs to one or more administrative agents or deployment managers, and an administrative agent or a deployment manager can register with more than one job manager, if desired.

The units of work that are handled by the flexible management environment are known as jobs. The semantics of these jobs are typically straightforward, and the jobs require few parameters. The jobs are processed asynchronously and can have an activation time, expiration time, and a recurrence indicator. You can specify that an -mail notification be sent upon completion of a job. Additionally, you can view the current status of a job by issuing a status command.

In Figure 3-2, we see that the administrative agent looks like it communicates directly to the job manager node. In practice, the individual application server that is managed by the administrative agent is registered with the job manager directly.

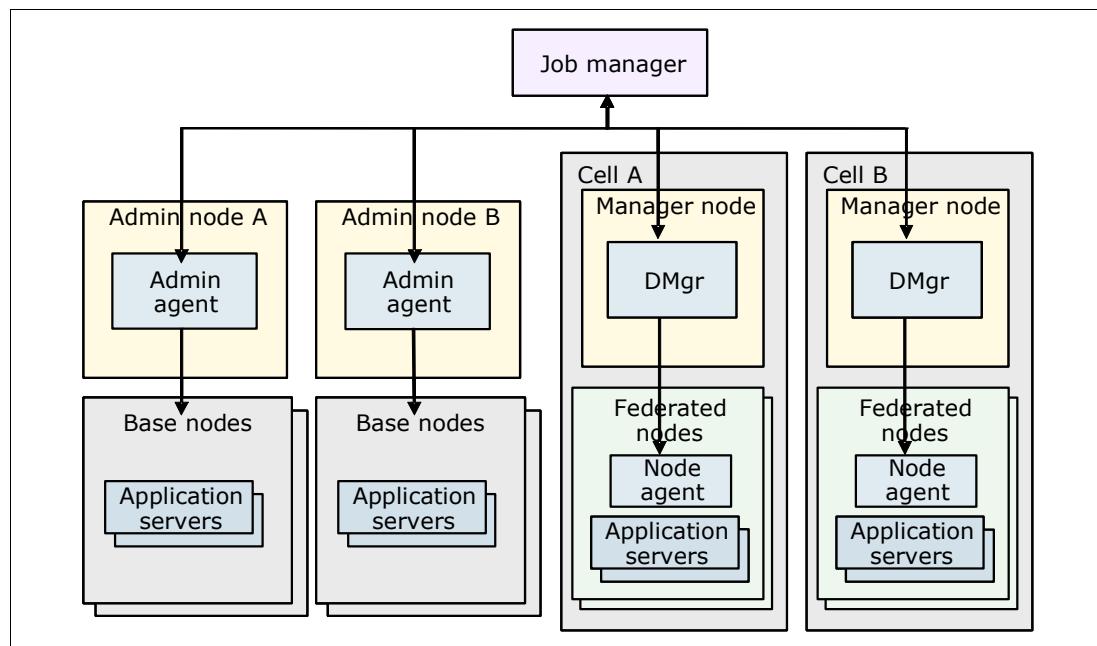


Figure 3-2 High-level overview of a job manager architecture

New in Version 8: In WebSphere Application Server V8, you can complete job manager actions and run jobs from a deployment manager.

3.1.7 Profile generation

Profiles can be created at any time during or after installation using graphical or command-line tools. WebSphere Application Server provides the following profile management tools:

- ▶ The **manageprofiles** command: A command-line interface for profile management functions.
- ▶ Profile Management Tool (PMT): A GUI interface in the WebSphere Customization Toolbox that gathers user input and invokes the **manageprofiles** command-line tool to manage the profiles.

3.2 Planning for profiles

Profiles can be created using the WebSphere Customization Toolbox Profile Management Tool graphical interface or in silent mode using the `manageprofiles` command.

Regardless of the method you use, a minimum amount of space must be available in the directory where you create a profile. This minimum requirement is documented in the Information Center at the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.nd.doc/info/ae/ae/rpro_diskspace.html

Note: When you use the Profile Management Tool with the Motif graphical user interface on the Solaris operating system, the default size of the Profile Management Tool might be too small to view all the messages and buttons of the Profile Management Tool. To fix the problem, add the following lines to the `app_server_root/.Xdefaults` file:

```
Eclipse*spacing:0  
Eclipse*fontList:-misc-fixed-medium-r-normal-* -10 -100 -75 -75 -c -60 -iso8859-1
```

After adding the lines, run the following command before launching the Profile Management Tool:

```
xrdb -load user_home/.Xdefaults
```

Profiles grow when applications and associated log files are created, and therefore these increases must be considered at the planning stages.

An error can occur when you do not provide enough space to create a profile. Verify that you have, in addition to the minimum space required for a particular profile, an additional 40 MB of space, which is used for log files and temporary files.

3.3 Building systems with profiles

This section shows how to use the WebSphere Customization Toolbox Profile Management Tool (PMT) to create profiles on distributed systems. Information about creating profiles in silent mode is provided in 3.4.4, “Creating a profile with the `manageprofiles` command” on page 137.

3.3.1 Starting the WebSphere Customization Toolbox Profile Management Tool

The first steps in creating a profile are common, regardless of the type of profile you are going to create.

Complete the following steps to create the profile:

1. Start the WebSphere Customization Toolbox using one of the following methods:
 - a. At the end of the installation process using the Installation Manager install wizard, select the option to start the Profile Management Tool to create a profile.

- b. Select the WebSphere Customization Toolbox option from the First steps console.
- Windows only:
From the Start menu, select **Start → Programs → IBM WebSphere → Application Server Network Deployment V8.0 → WebSphere Customization Toolbox**
 - For Linux only:
From the operating system menu to start programs, select **Applications → IBM WebSphere Application Server Network Deployment V8.0 → WebSphere Customization Toolbox → Profile Management Tool**.
 - For all platforms:
Use the **wct.bat(sh)** command in the `<install_root>/bin/ProfileManagement` directory.

Note: A **pmt.bat(sh)** shell script is provided for backward compatibility, but is deprecated in WebSphere Application Server V8.

2. When you start the wizard, you see the **Profile Management Tool** tab. Then you see a list of existing profiles. Click **Create** to start the profile creation process, as shown in Figure 3-3.

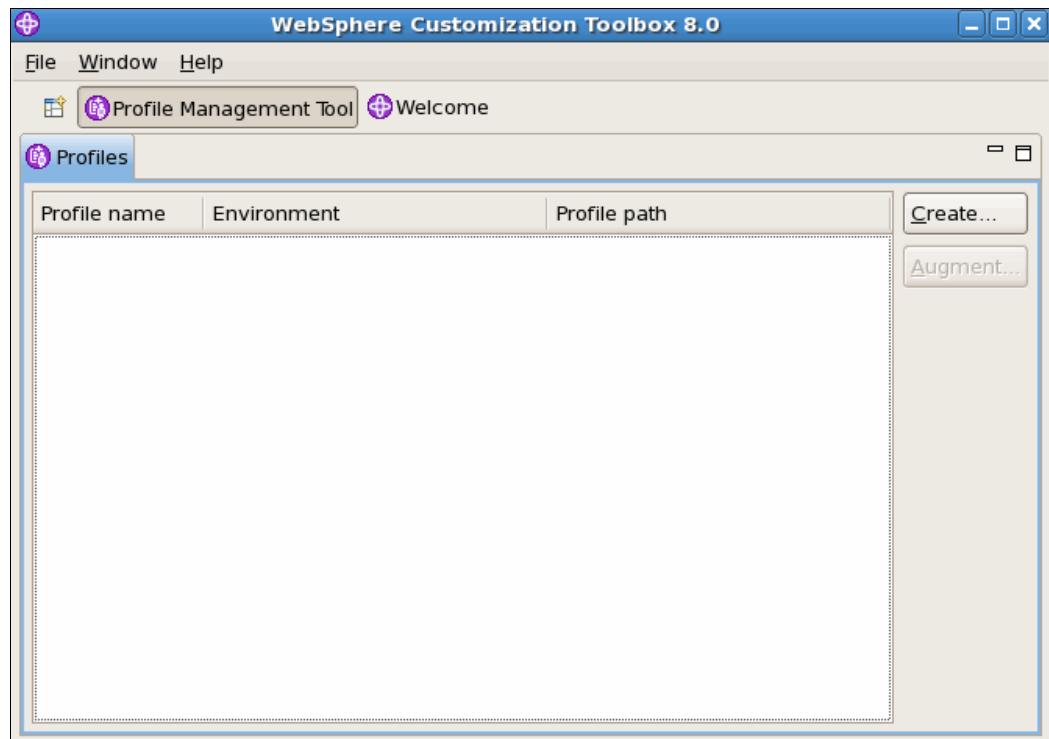


Figure 3-3 Profile Management Tool main display

3.3.2 Common windows and steps for all profiles

Many of the options that you have when you create a profile are the same, regardless of the type of profile.

Environment selection

The Profile Management Tool provides multiple profile templates, including the cell template, which has the ability to create a cell in a single step. During profile creation, you will be asked to select the type of profile to create, as shown in Figure 3-4.

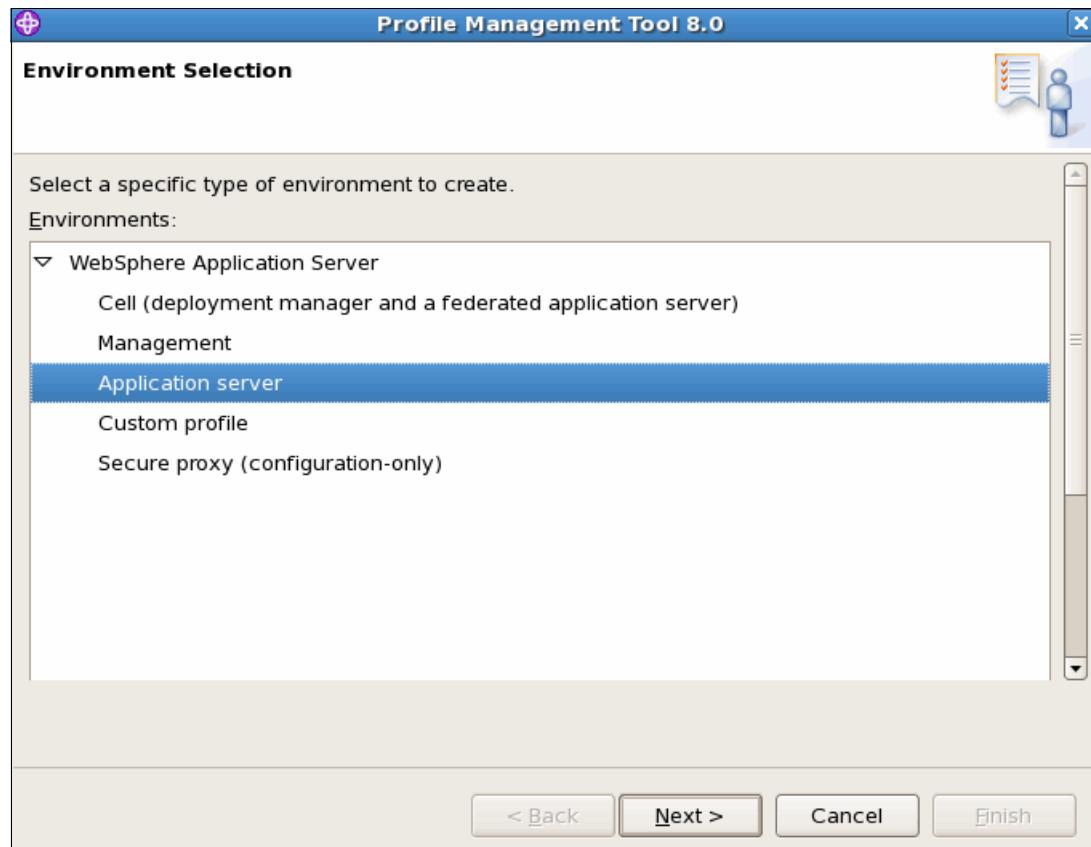


Figure 3-4 Profile type selection

The profile options are listed next. Note that the deployment manager profile is under the Management option, along with the profile types for flexible management (administrative agent and job manager):

- ▶ Cell (deployment and a federated application server)
- ▶ Management
 - Administrative agent
 - Deployment manager
 - Job manager
- ▶ Application server
- ▶ Custom profile
- ▶ Secure proxy (configuration-only)

Profile creation options

While creating profiles, you are presented with a choice (Figure 3-5) of following the “Typical” path, where a set of default values for most settings will be used, or an “Advanced” path, which lets you specify values for each option.

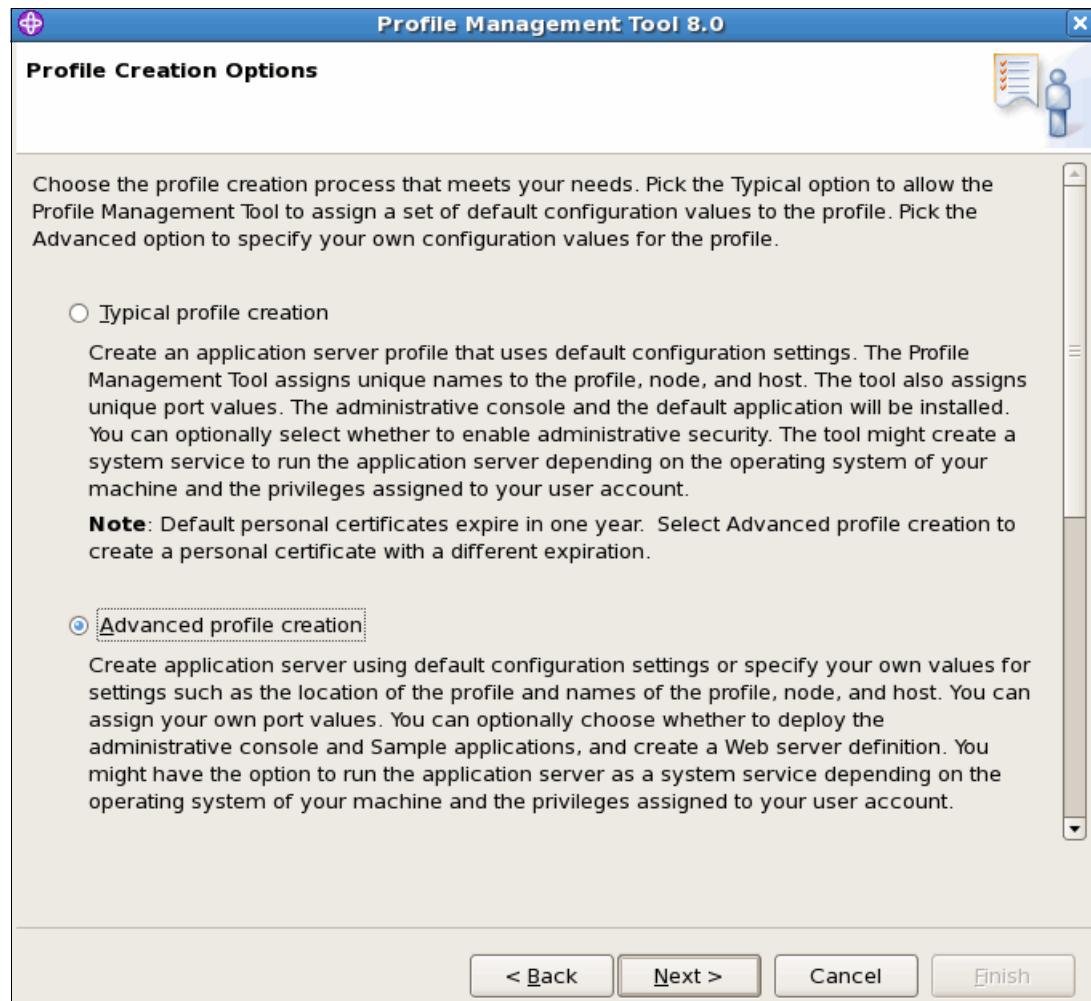


Figure 3-5 Profile creation path (typical versus advanced option)

The Advanced path is preferred because it gives you additional control over names and settings. An overview of the Advanced profile options is shown in Table 3-1.

Table 3-1 Options available in the typical versus advanced path

Option	Application server	Deployment manager	Administrative agent	Job manager	Custom	Cell
Deploy Administrative Console	Yes	Yes	Yes	Yes	No	Yes
Default Application	Yes	No	No	No	No	Yes
Profile Name and Location	Yes	Yes	Yes	Yes	Yes	Yes

Option	Application server	Deployment manager	Administrative agent	Job manager	Custom	Cell
Node and Host Names	Yes	Yes	Yes	Yes	Yes	Yes
Administrative Security	Yes	Yes	Yes	Yes	Yes (Federation)	Yes
Certificates(parts 1 and 2)	Yes	Yes	Yes	Yes	Yes	Yes
Port Assignment	Yes	Yes	Yes	Yes	Yes	Yes (Part 1 and part 2 ; one for dmgr, the other for App Server)
Windows Services (Windows only)	Yes	Yes	Yes	Yes	Yes	Yes
Linux Service (Linux only)	Yes	Yes	Yes	Yes	Yes	Yes
Web Server definition (parts 1 and 2)	Yes	No	No	No	No	Yes
Summary	Yes	Yes	Yes	Yes	Yes	Yes

Profile name and location (and default profiles)

The wizard asks for a profile name and a location where you want the profile configuration files stored.

Directory location

By default, profiles are stored in `install_root/profiles/profile_name`. The logs for the process defined by the profile reside within this directory structure, but you can easily change this setting if space is a concern.

Default profile

The first profile that you create on a machine is the default profile. The default profile is the default target for commands that are issued from the `bin` directory in the product installation root when the `-profileName` argument is not used.

You can make another profile the default profile when you create that profile by checking **Make this profile** the default on the Profile name and location window of the Advanced profile creation path. You can also make another profile the default profile using the `manageprofiles` command after you create the profile.

Profile name

The profile name must be unique within the installation and should follow an appropriate naming convention so you can easily identify it by the name it is given. The guidelines are as follows:

- ▶ Double-byte characters are supported.

- ▶ The profile name can be any unique name with the following restrictions:
 - Do not use any of the following characters when naming your profile:
 - Spaces
 - Special characters that are not supported within the name of a directory on your operating system, such as *&?
 - Slashes (/) or (\)

Administrative security

When you create a profile for a process with the administrative functions (basically everything but a custom profile), you have the opportunity to enable administrative security. Enabling administrative security prevents any user from gaining unauthorized access to the administrative tools. If you enable administrative security during profile creation, you are asked for a user ID and password that will be added to a file-based user registry with the Administrator role, as shown in Figure 3-6.

You should enable administrative security. An XML file-based user repository is created during profile creation and can be federated with other repositories later to provide a robust user registry for both administrative and application security. If you do not want to use the file-based repository, do not enable administrative security during profile creation or change it afterwards.



Figure 3-6 Enable administrative security

Note: If you are going to create a job manager and register a deployment manager, keep in mind that you cannot register a deployment manager that has security enabled to a job manager that does not. So, you should plan for administrative security across the WebSphere environment.

You can find more information about administrative security in 6.2, “Securing the administrative console” on page 247.

Certificates

Each profile contains a unique chained certificate signed by a unique long lived root certificate that is created when the profile was created. When a profile is federated to a deployment manager, the signer for the root signing certificate is added to the common truststore for the cell, establishing trust for all certificates signed by that root certificate.

For a full description of the certificates and the keystore password, see the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.nd.doc/info/ae/ae/csec_7ssldefault_chainedcert_config.html

Two windows are used during profile creation to manage the import or creation of these certificates.

The first window (Figure 3-7) allows you to create the certificates or import existing certificates.

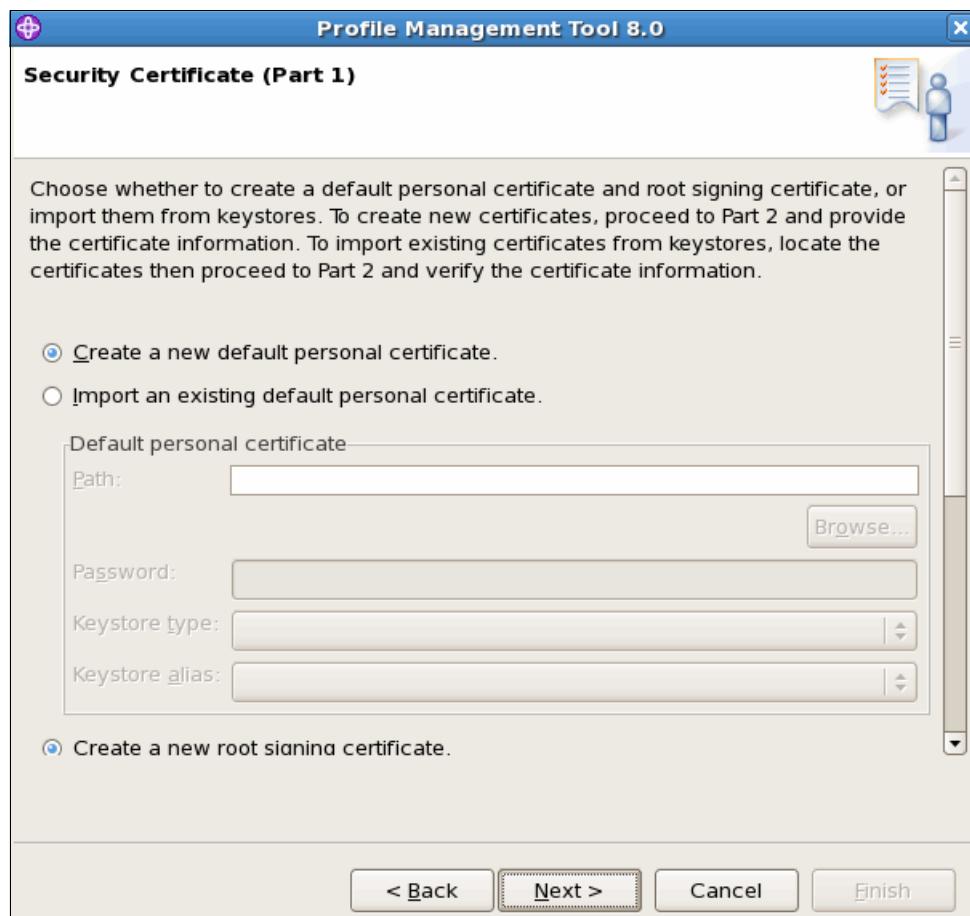


Figure 3-7 Create certificates or import existing personal or root certificates

Note: In WebSphere Application Server V8, signer and personal certificates can be either created or imported during profile creation. If you have new certificates created, you can choose the correct DN during profile creation.

The second window (Figure 3-8) is used to modify the certificate information to create new certificates during profile creation. Review the expiration period and provide a new password for the default keystore. The default password is WebAS.

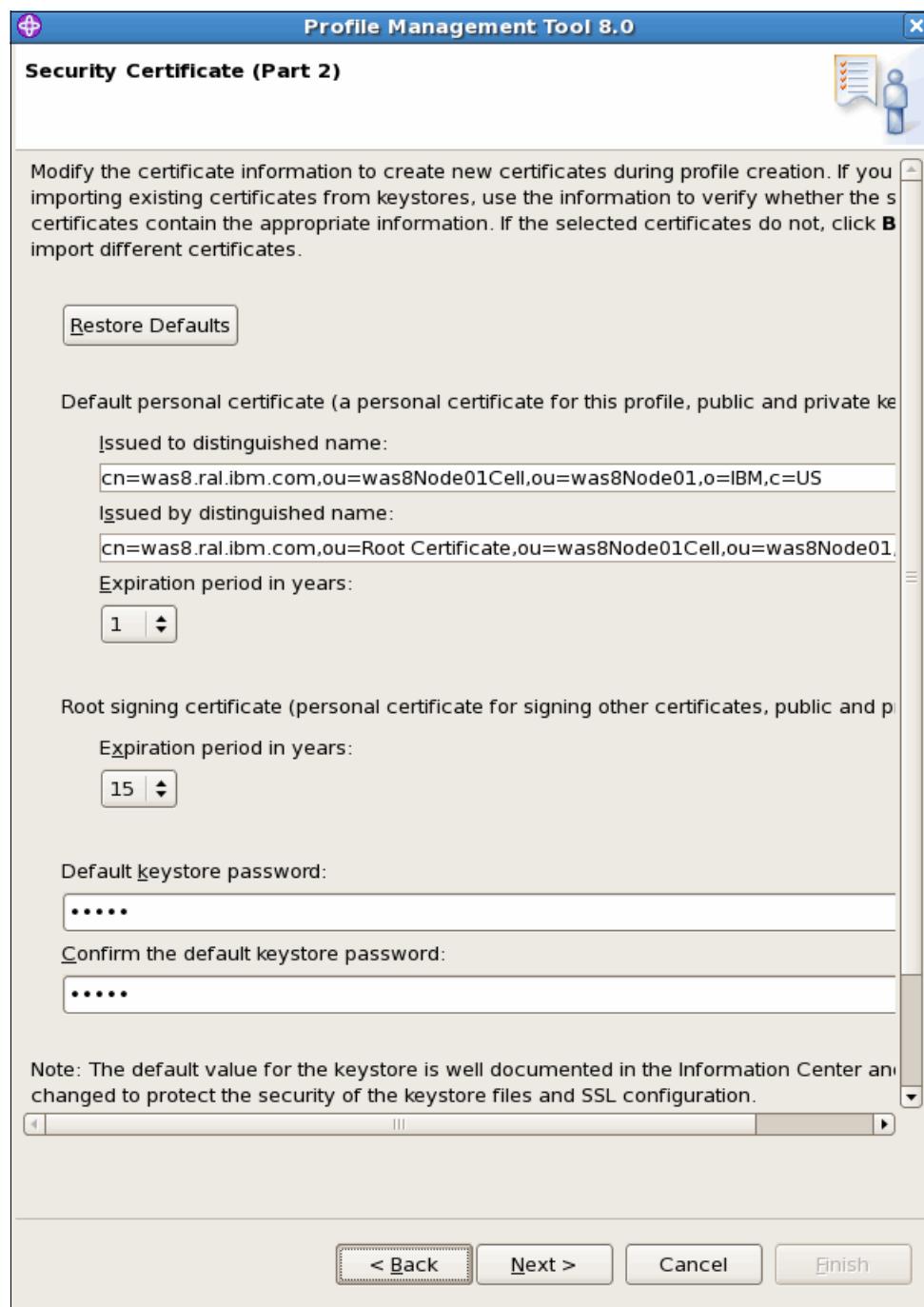


Figure 3-8 Modify certificate information at profile creation time

Port assignments

Every process uses a set of ports at run time. These ports must be unique to a system. For the default port assignment for the distributed platform, see the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.migration.base.doc/info/aes/ae/rmig_portnumber.html

The profile management tool wizard assigns unique port numbers to each profile if multiple profiles are installed in the same system. Careful planning is needed so that there are no port conflicts with other software installed on the same systems.

When you take the Advanced path through the profile wizard, you have three options:

- ▶ Use the default set of port numbers.
- ▶ Use the recommended set of port numbers. These have been selected as unique to the WebSphere installation.
- ▶ Customize the port numbers.

After profile creation, you can obtain port numbers by looking in the following location:

profile_home/properties/portdef.props

Information: You can also use the PortManagement command group for the AdminTask object in **wsadmin** to list application and server ports and modify server ports. For more information about the PortManagement command group, see the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.nd.doc/info/ae/ae/rxml_atportmgt.html

Running as a Windows service

When you create a profile on a Windows system, you have the option of running the application server as a Windows service. This action provides you with a simple way of automatically starting the server process when the system starts.

If you would like to run the process as a Windows service, select the check box and enter the values for the logon and startup type. Note that the window lists the user rights that the user ID you select needs to have. If the user ID does not have these rights, the wizard automatically adds them.

When you take the Typical path through the profile creation wizard, the default is to define the process as a Windows service (Figure 3-9).

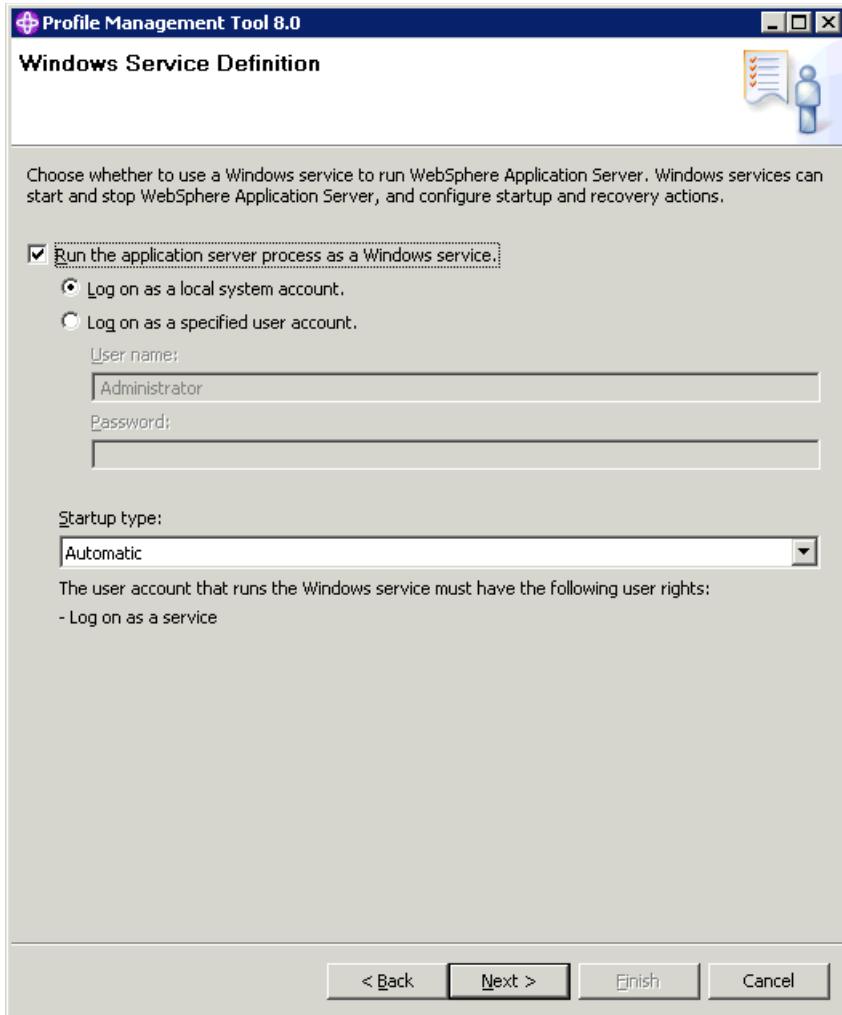


Figure 3-9 Run as a Windows service

If you do not register the process as a Windows service during profile creation, you can do that later using the **WASService** command. For more information about the **WASService** command, see the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.base.doc/info/aes/ae/rins_wasservice.html

Running as a Linux service

When you create a profile on a Linux system, you have the option of running the application server as a Linux service. This action provides you a simple way of automatically starting the server process when the system starts.

If you would like to run the process as a Linux service, select the check box and enter the values for the logon and startup type. Note that the window lists the user rights that the user ID you select needs to have. If the user ID does not have these rights, the wizard automatically adds them.

When you take the Typical path through the profile creation wizard, the default is to not define the process as a Linux service (Figure 3-10).

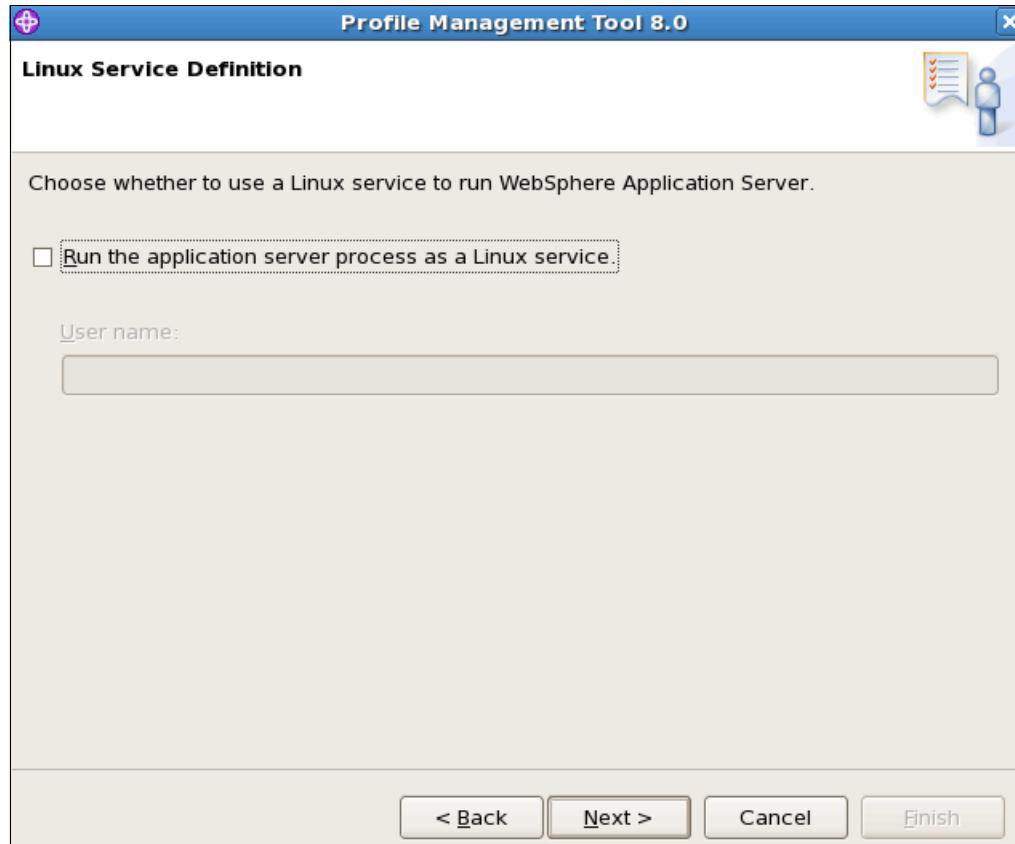


Figure 3-10 Run as a Linux service

If you do not register the process as a Linux service during profile creation, you can do that later using the `WASService` command. This command enables to you create a service for a Java process on both Linux and Windows operating systems.

Verification steps

The profile is stored in the directory structure you selected. In this book, we refer to this directory as `profile_root`. This is where you can find, among other things, the `config` directory containing the configuration files, the `bin` directory for entering commands, and the `logs` directory where information is recorded.

At the end of the profile creation, you have the opportunity to start the First steps console. This interface helps you start the server process and has other useful links, such as opening the administrative console, an Information Center and IBM Education Assistant link, starting the WebSphere Customization Toolbox, and installation verification.

After you create a new profile, you can complete the following steps to verify that the profile is working correctly:

1. View the messages produced by the profile creation. First, note the messages that result from the profile creation (Figure 3-11).

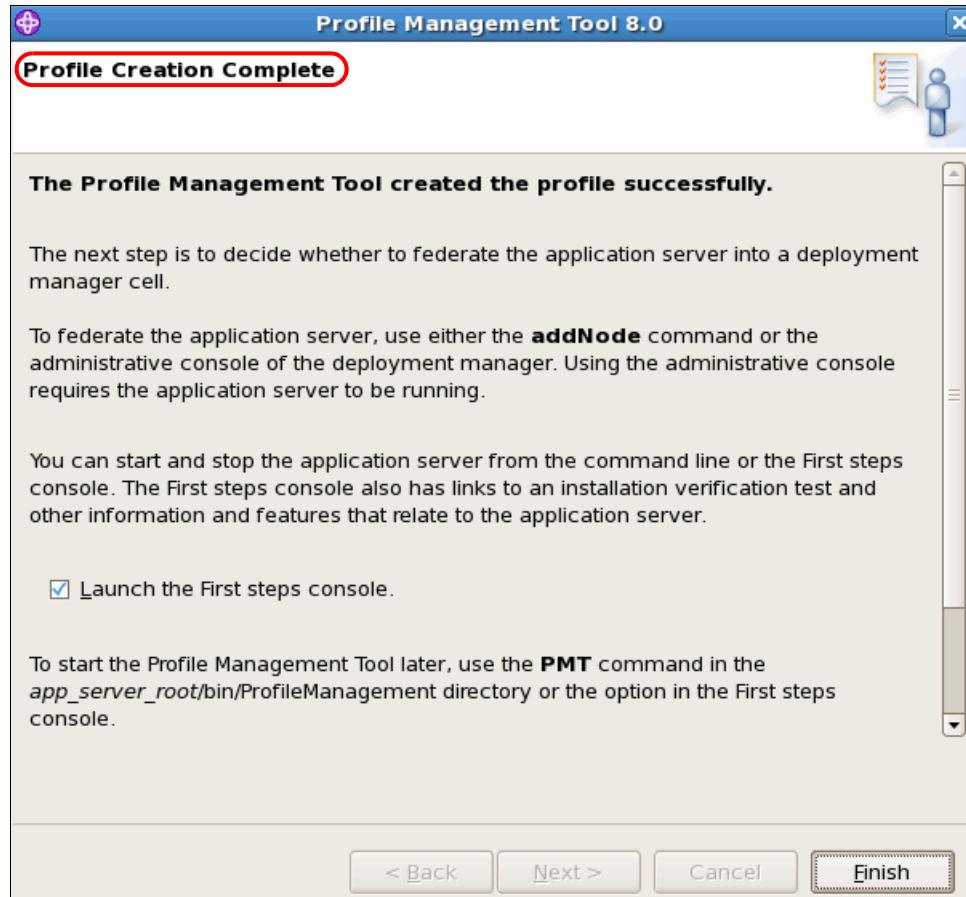


Figure 3-11 Profile creation complete - Messages

If the messages indicate that the profile was not created successfully, then look in *install_root/logs/manageprofiles/profile_name_create.log* to determine what went wrong.

2. Run the installation verification test.

Each profile has its own installation verification tests (IVT) program that starts the process defined by the profile and runs a series of verification tests. The IVT program scans the *SystemOut.log* file for errors and verifies the core functionality of the profile.

Use the First Steps console to run the IVT. The First Steps console starts by default when you click **Finish** at the end of the profile creation. You can start the First Steps console any time by using the **firststeps** command located in the *profile_root/firststeps* directory. The options on this console vary depending on the profile type.

Alternatively, the **ivt** command can be executed from *profile_root/bin*. The IVT program verifies that the installation of the application server or deployment manager profile was successful.

Messages from the IVT are displayed on the First Steps window and logged in the following places:

- *profile_root/logs/server_name/startServer.log*
- *profile_root/logs/server_name/SystemOut.log*

3. If applicable, log in to the administrative console hosted by the process. You can access the console from the First Steps menu or by accessing its URL from a web browser:

`http://server_host:<admin_console_port>/ibm/console`

Here is a sample URL:

`http://localhost:9060/ibm/console/`

The administrative console port is selected during profile creation (see Figure 3-16 on page 99).

Click the **Log in** button. If you did not enable security, you do not have to enter a user name. If you choose to enter a name, it can be any name. It is used to track changes you make from the console. If you enabled administrative security, enter the user ID and password you specified.

Figure 3-12 shows the First steps console for a stand-alone application server.

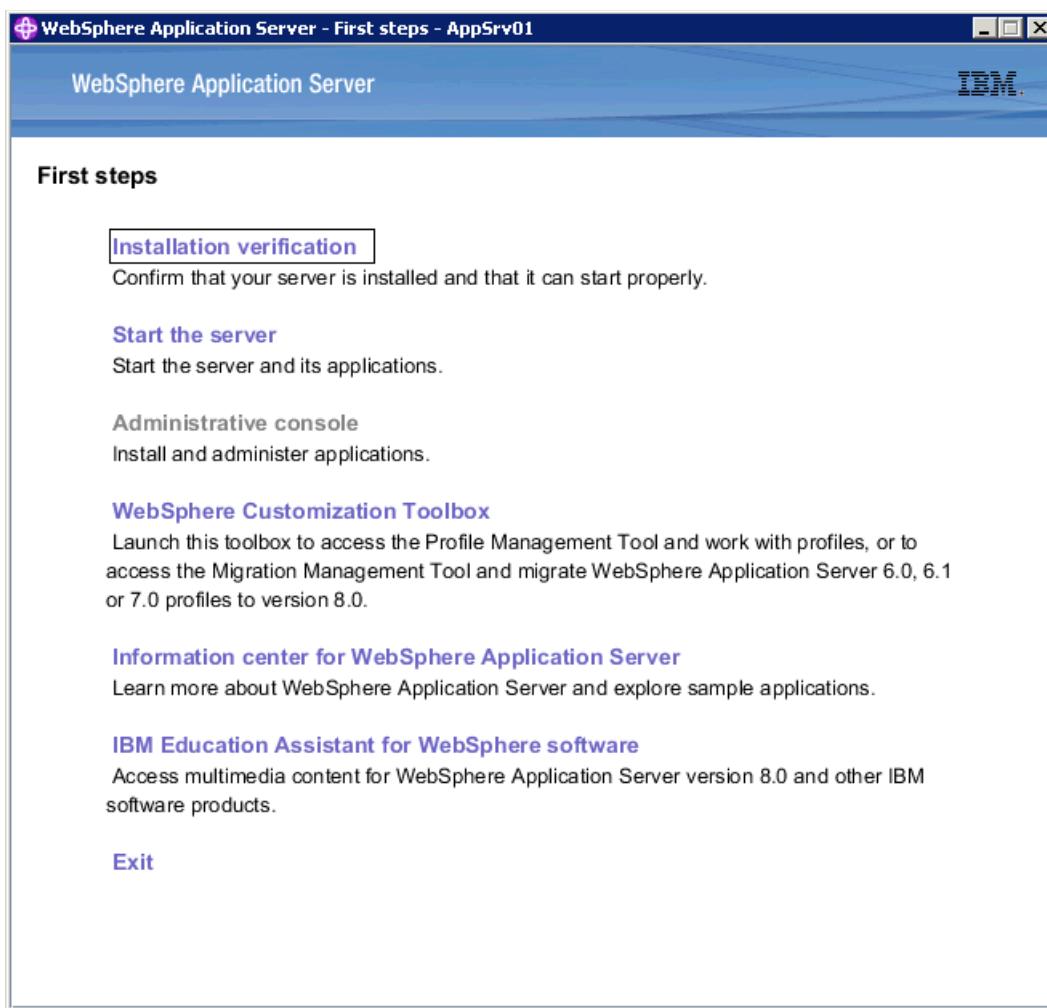


Figure 3-12 First steps

3.3.3 Creating an application server profile

An application server profile defines a new stand-alone application server. This server can be run stand-alone or can be later federated to a deployment manager cell for central management.

This section takes you through the steps of creating the application server profile using the WebSphere Customization Toolbox Profile Management Tool. It shows the steps in the Advanced path through the profile creation.

To create the profile, complete the following steps:

1. Start the WebSphere Customization Toolbox. The **Profile Management Tool** window opens.
2. Click the **Create** button.
3. Select **Application server** as the profile type and click **Next**.
4. Select **Advanced**. Click **Next**.
5. Select the applications you want to deploy (Figure 3-13).

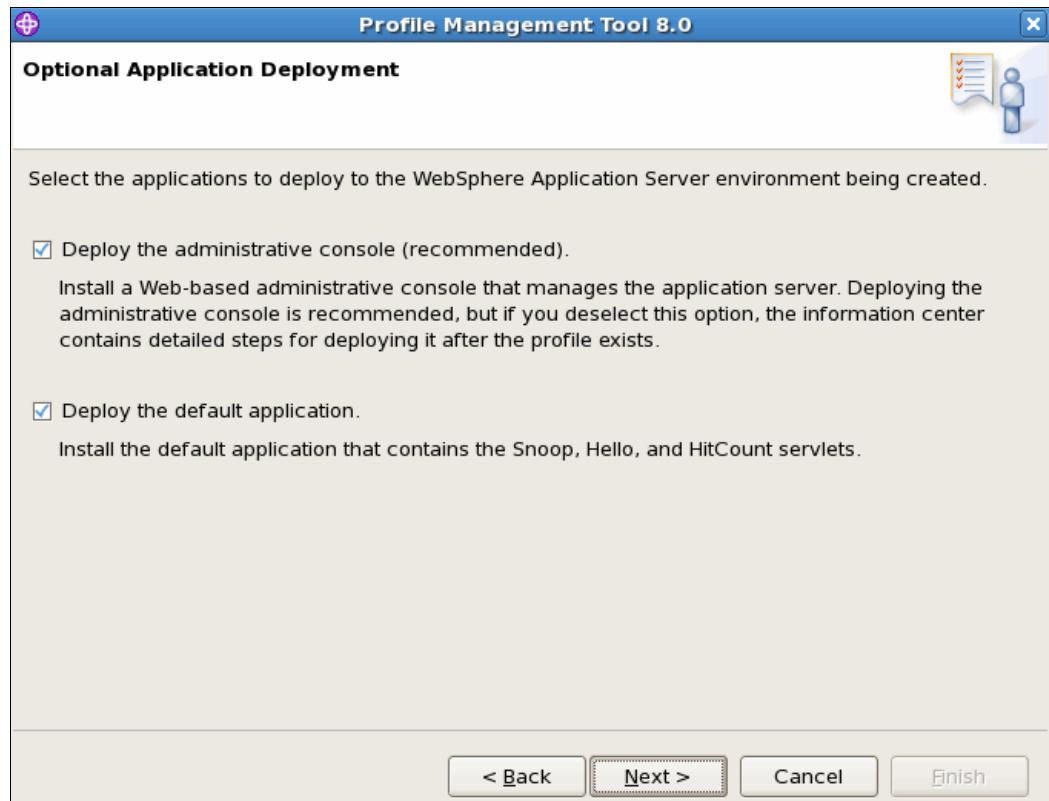


Figure 3-13 Application selection

Installing the administrative console is recommended. However, there might be some circumstances when you would not want to install an administrative console, such as though you plan to control all administrative tasks through scripting. If you do not install the administrative console during profile creation, you can install using the `deployConsole.py` script at a later time.

Information: To install the administrative console after profile creation, complete the following steps:

1. Navigate to *profile_root/bin*.
2. Start the server using the following command:

```
startServer.bat(sh) server1
```

3. Enter the following command to install the application:

```
wsadmin.bat(sh) -lang jython -f deployConsole.py install
```

If you configured administrative security during profile creation, you are prompted for an administrative user ID and password when running the **wsadmin** install command.

You have the option of deploying the default application. This is a default application that can be used to verify that your application server is running and serving application content. The default application contains a web module called DefaultWebApplication and an EBJ called Increment. The application includes a number of servlets that retrieve information that can be used for verification.

New in Version 8: In WebSphere Application Server V8, you no longer have the option of deploying sample applications during profile creation. The Deploy the Sample applications option has been removed from profile creation. During product installation, you can choose to install the PlantsByWebSphere sample application. This is the only sample application included with the product. You can find other sample applications for downloading at the Information Center. For information about the samples available and how to install them, see the topics *Accessing the samples* and *Learn about WebSphere applications* in the Information Center at the following websites:

- ▶ http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.nd.doc/info/ae/ae/covr_samples.html
- ▶ <http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.nd.doc/info/ae/ae/we1c6tech.html>

Click **Next**.

6. Enter a unique name for the profile or accept the default. Enter a unique directory path for the profile directory or accept the default.

New in Version 8: A server runtime performance tuning option has been introduced in WebSphere Application Server V8. Select a performance tuning setting for the profile, or accept the default setting of standard (Figure 3-14). You have three performance tuning options to choose from:

- ▶ Standard, which is the standard default configuration settings that are optimized for general purpose usage. These settings are conservative.
- ▶ Peak, which is appropriate for a production environment where application changes are rare and optimal runtime performance is important.
- ▶ Development, which is appropriate for a development environment where frequent application updates are performed and system resources are at a minimum. Do not use the development setting for production servers.

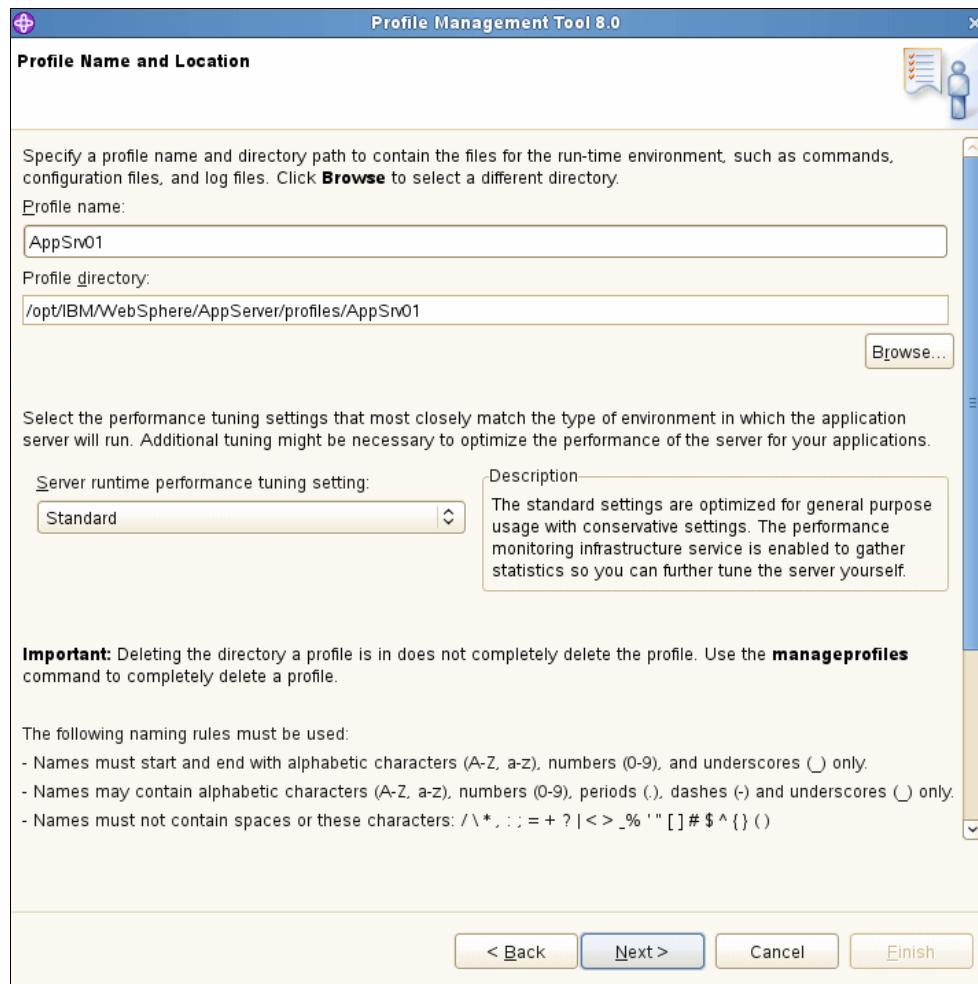


Figure 3-14 Profile name and location settings

Click **Next**.

7. Enter the new node name and the system host name. The node name will default to a name based on the host name of your system. The wizard recognizes if there are existing nodes in the installation and takes this situation into account when creating the default node name (Figure 3-15).

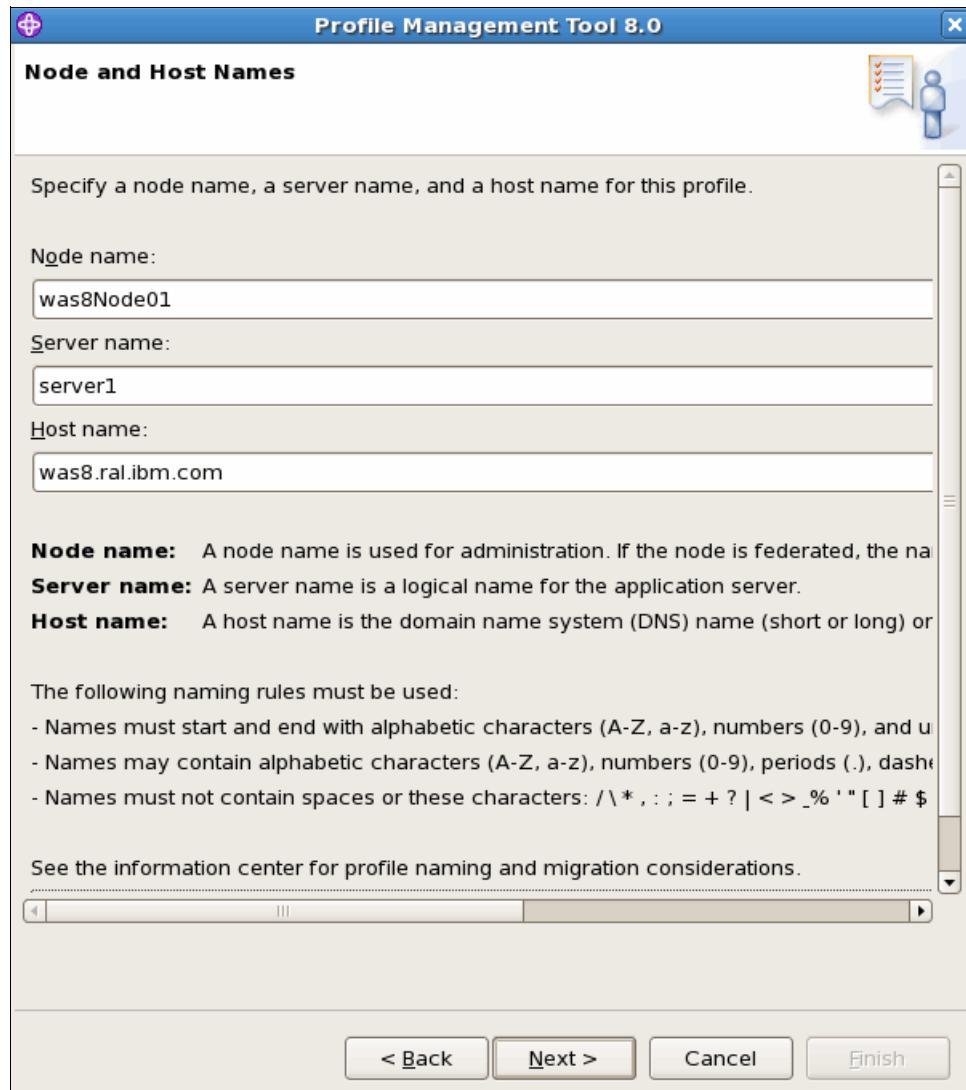


Figure 3-15 Enter host and node names

Click **Next**.

Note: If you are planning to create multiple stand-alone application servers for federation later to the same cell, make sure that you select a unique node name for each application server.

8. Choose whether to enable administrative security. If you enable security, you are asked for a user ID and password that will be added to a file-based user registry with the Administrator role.

Important: It is a good idea to make a note of the user ID and passwords you enter here. Without the administrator ID, you will not be able to manage the application server.

Click **Next**.

9. Elect to either create new default personal and root signing certificates or to import them.

Click **Next**.

10. Review and modify the certificate information as needed. Click **Next**.

11. The wizard presents a list of TCP/IP ports for use by the application server. If you already have existing profiles on the system (within this installation), this situation is taken into account when the wizard selects the port assignments, but you should verify that these ports will be unique on the system. You can select a different port by using the up and down arrows next to the port number.

Figure 3-16 shows typical port assignments.

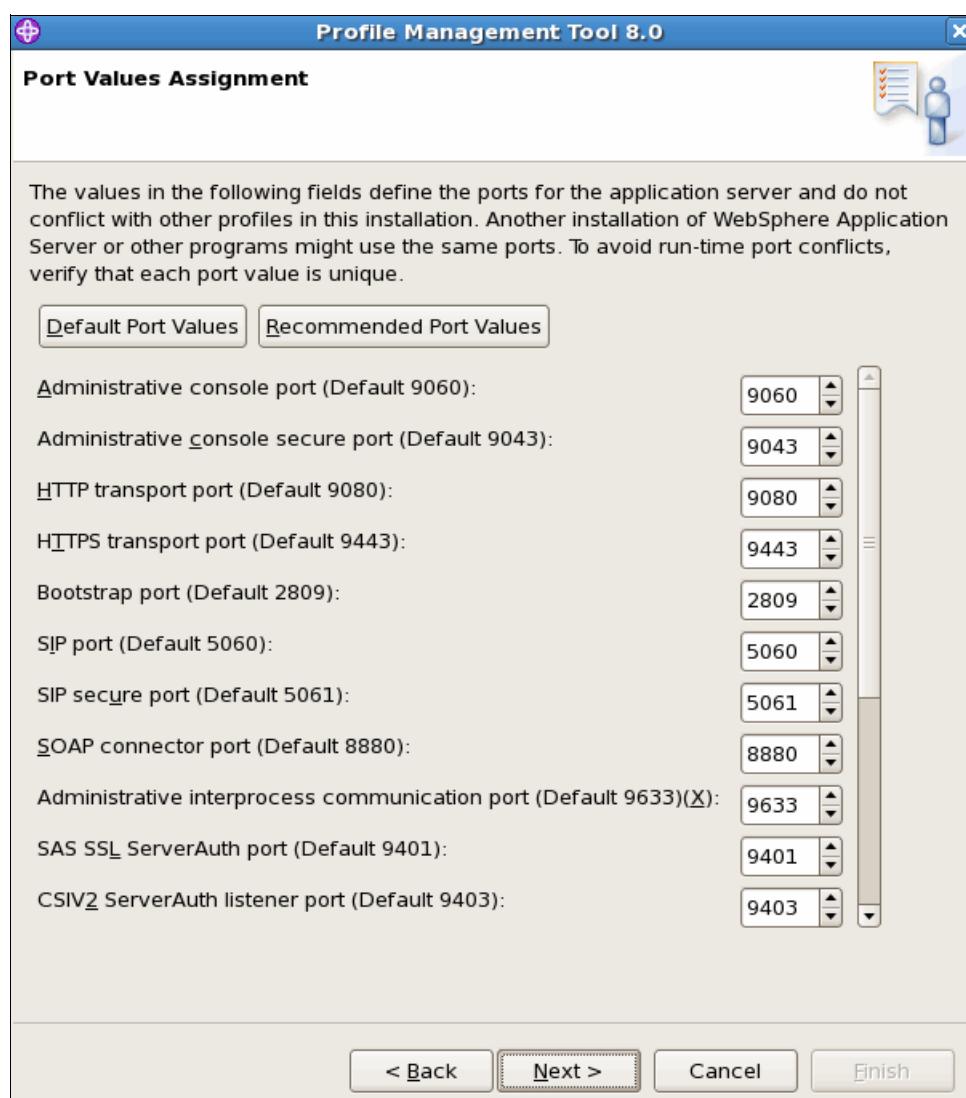


Figure 3-16 Select ports

Attention: Note the following port numbers for later use:

- ▶ *SOAP connector port*: If you plan to federate this node to a deployment manager later using the deployment manager administrator console, you need to know this port number. This port is also the port that you connect to when using the `wsadmin` administration scripting interface.
- ▶ *Administrative console port*: You need to know this port to access the administrative console. When you turn on security, you need to know the *Administrative console secure port*.
- ▶ *HTTP transport port*: This port is used to access applications running on the server directly versus going through a web server. This port is useful in test environments.

Click **Next**.

12. If the profile is being created on a Windows system, select whether you want the server to run as a Windows service. Click **Next**.

The wizard allows you to create an optional web server definition (Figure 3-17). Web server definitions define an external web server to WebSphere Application Server. This setup allows you to manage web server plug-in configuration files for the web server and, in some cases, to manage the web server. If you have not installed a web server or want to perform this action later, you can easily do it from the administrative console.

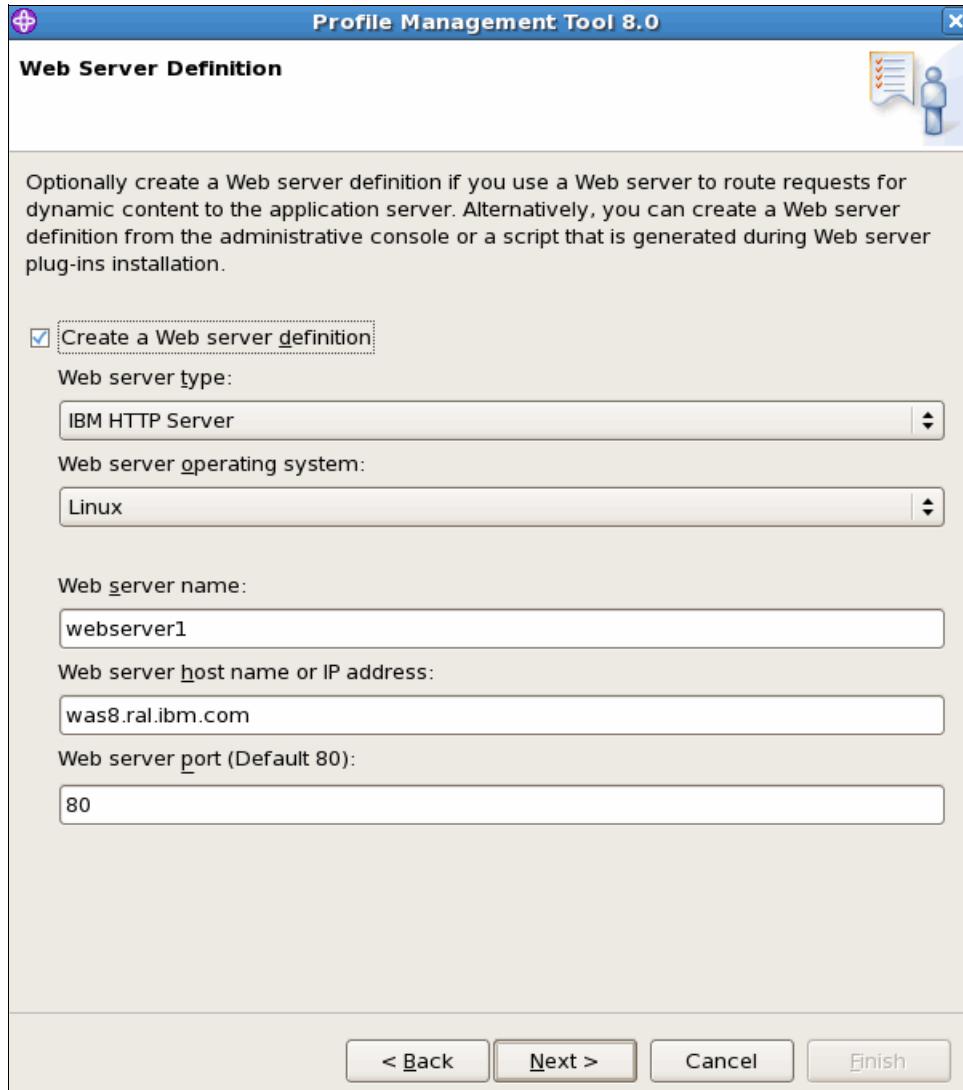


Figure 3-17 Creating a web server definition

Click **Next**.

13. If you elect to create a web server definition, the next window shows the default locations of the web server installation and web server plug-in path. Change these settings to match your installation (Figure 3-18).

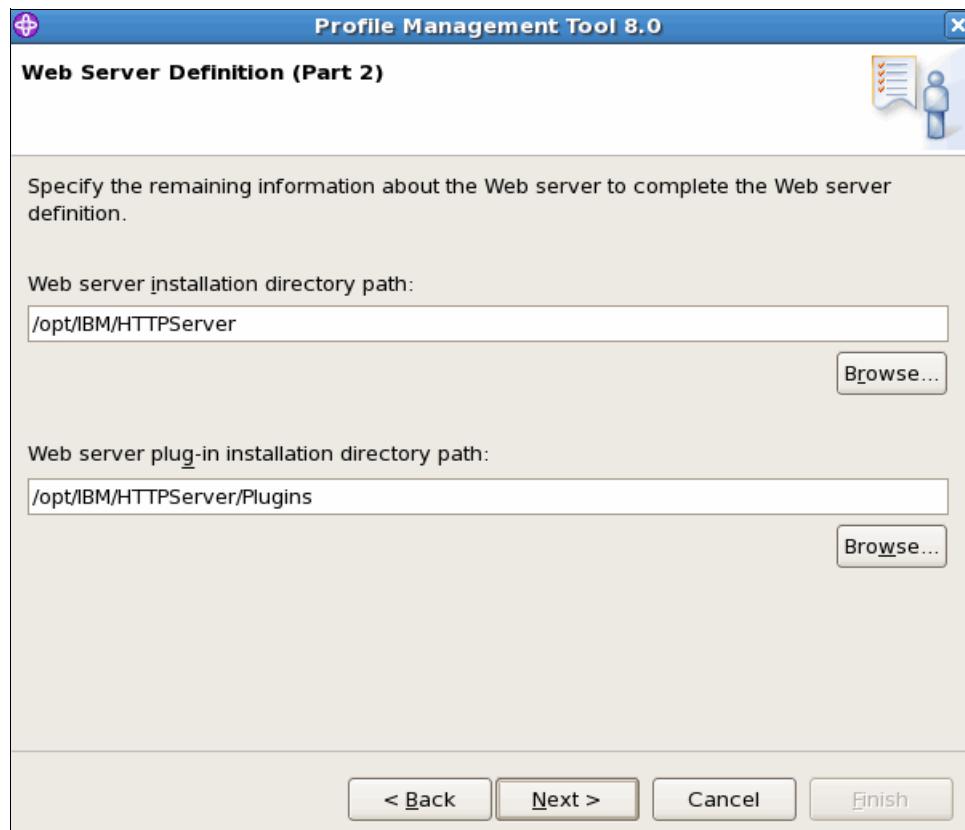


Figure 3-18 Location of the web server definition

Click **Next**.

14. Review the options you have chosen and click **Create** to create the profile.
15. The final window indicates the success or failure of the profile creation. If you have errors, check the log at:
install_root/logs/manageprofiles/profile_name_create.log
You can also find logs for individual actions stored in:
profile_root/logs
16. Click **Finish** to close the wizard and start the First Steps application.
17. Use the First Steps console to verify the installation, start the server, and log in to the administrative console.
18. Display the configuration from the console. You should be able to see the Application servers item in the administrative console.

Click **Servers** → **Server Types** → **WebSphere Application servers**. You should see the application server in the list (Figure 3-19).

The screenshot shows the WebSphere Application servers page. The left sidebar has a tree view with 'Server Types' expanded, showing 'WebSphere application servers'. The main content area displays a table with one row:

Name	Node	Host Name	Version
server1	was8Node01	was8.ral.ibm.com	ND 8.0.0.0

Total 1

The right sidebar contains help sections: 'Field help', 'Page help' (with a link to 'More information about this page'), and 'Command Assistance' (with a link to 'View administrative scripting command for last action').

Figure 3-19 Application server defined by the application server profile

Working with application servers: For information about starting, stopping, and viewing application servers, see Chapter 6, “Administration consoles and commands” on page 223.

3.3.4 Creating a deployment manager profile

This section takes you through the steps of creating the deployment manager profile using the WebSphere Customization Toolbox Profile Management Tool. It shows the steps in the Advanced path through the profile creation.

To create the profile, complete the following steps:

1. Start the WebSphere Customization Toolbox. You see the Profile Management Tool window.
2. Click **Create**.
3. Click **Management**. Click **Next**.
4. Click **Deployment manager**. Click **Next**.
5. Select whether to take the typical settings or to go through the advanced windows. The options that you see next depend on the path you take.
 - If Typical is selected, then you only see one more option (to enable security).
 - If Advanced is selected, continue with the following steps.
6. Select the option to deploy the administrative console (the default) and click **Next**.

7. Enter a unique name for the profile or accept the default. The profile name becomes the directory name for the profile files. Select the check box if you want this to be the default profile for receiving commands. Select the location for the profile and click **Next** (Figure 3-20).

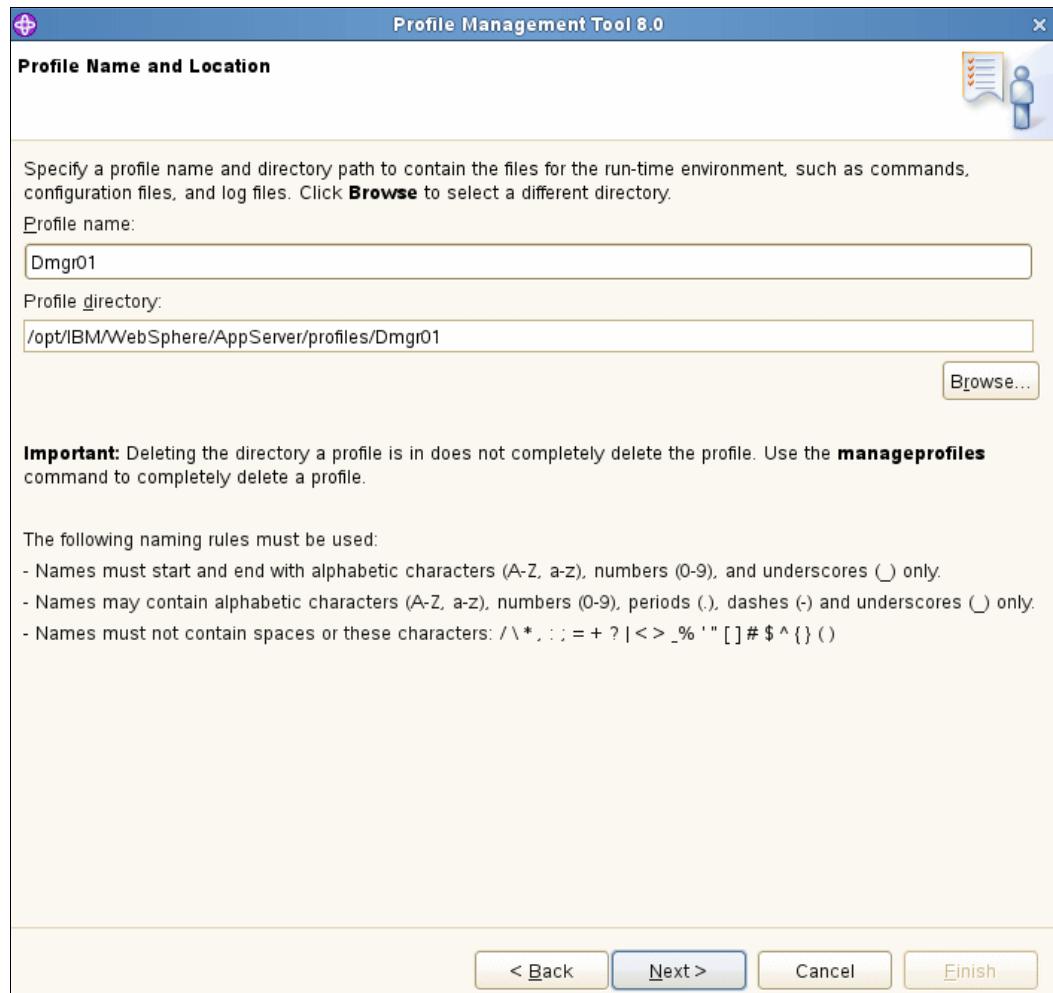


Figure 3-20 Creating a deployment manager profile - Enter name and location

- Enter the node, host, and cell names. The defaults are based on the host name of your system. The wizard recognizes if there are existing cells and nodes in the installation and takes this setup into account when creating the default names. Click **Next** (Figure 3-21).

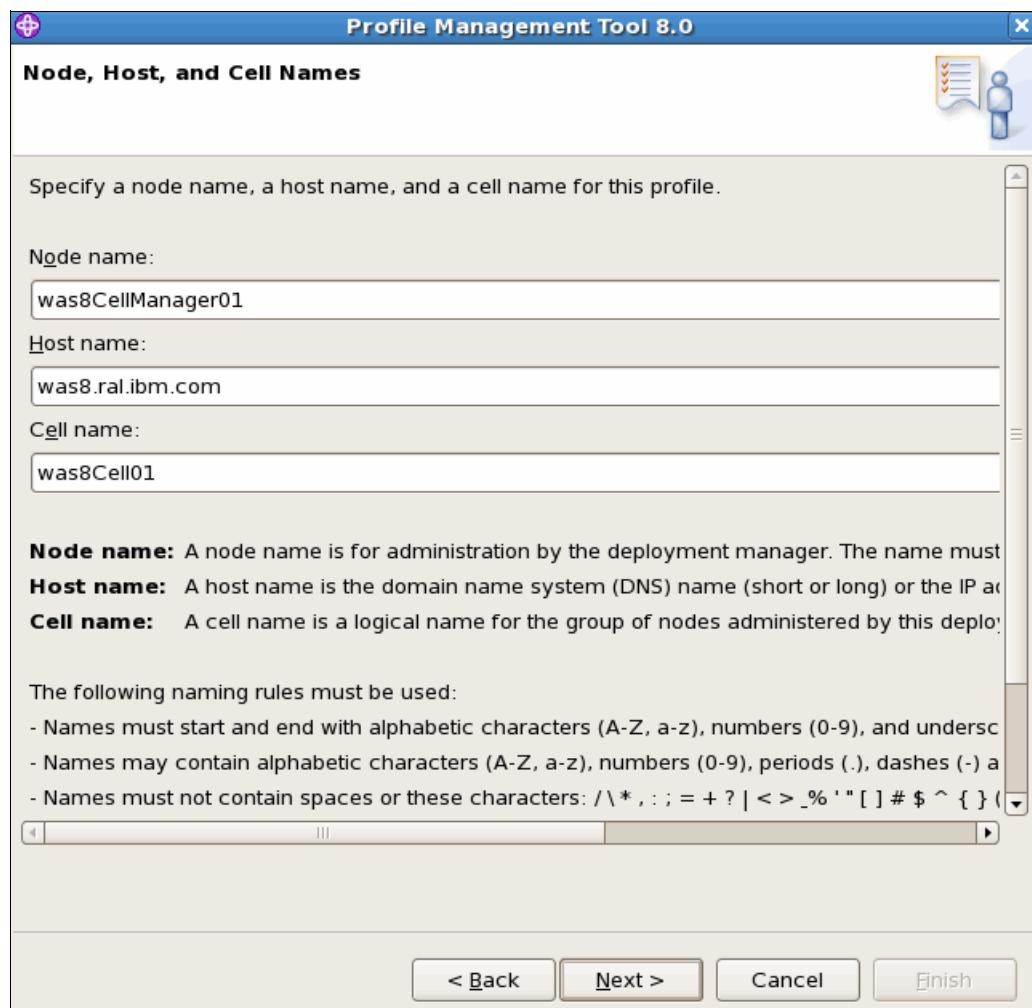


Figure 3-21 Creating a deployment manager profile - Enter cell, host, and node names

- Choose whether to enable administrative security. If you enable security here, you are asked for a user ID and password that will be added to a file-based user registry with the Administrator role. Click **Next**.
- Select to either create new default personal and root signing certificates or to import them. Click **Next**.
- Review and modify the certificate information as needed. Click **Next**.

12. The wizard presents a list of TCP/IP ports for use by the deployment manager. If you already have existing profiles on the system, they are taken into account when the wizard selects the port assignments. However, you should verify that these ports will be unique on the system (Figure 3-22).

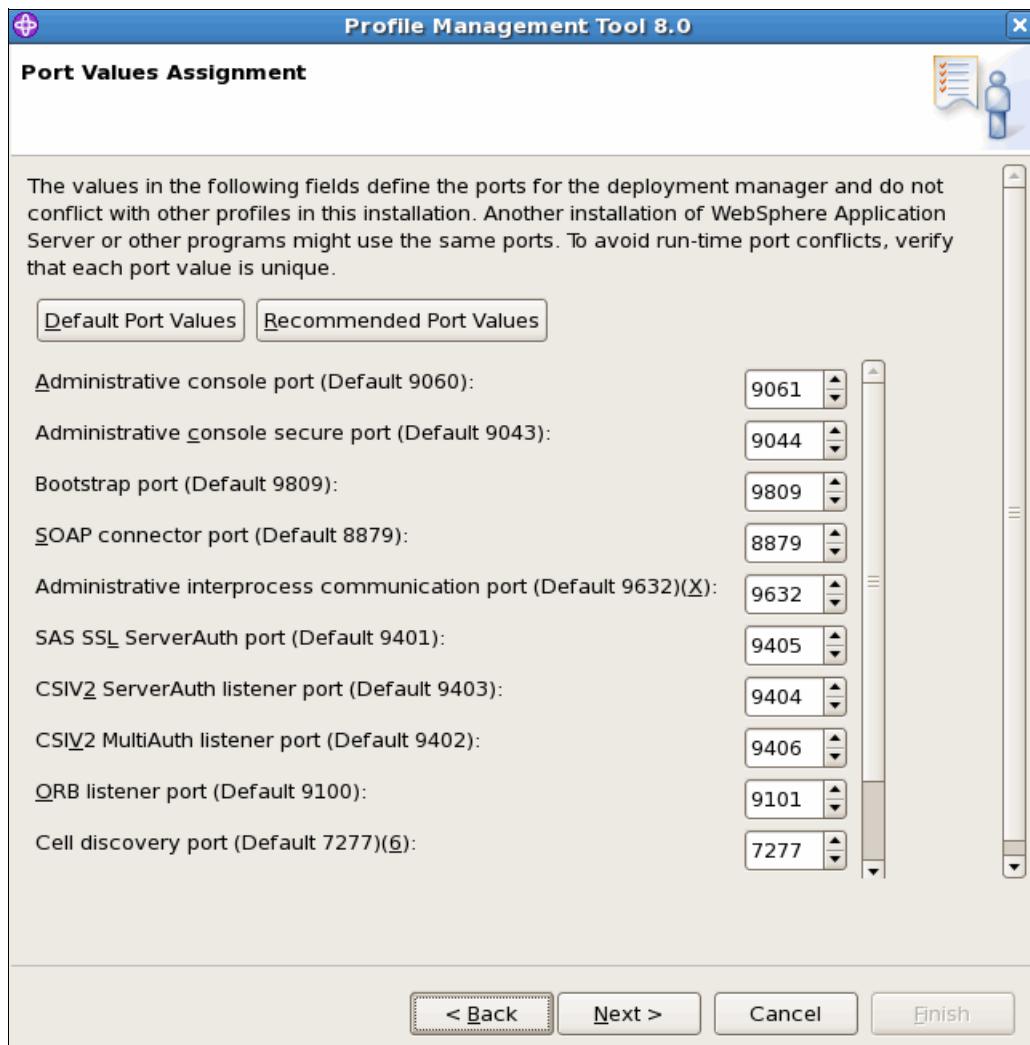


Figure 3-22 Creating a deployment manager profile - Select ports

Note: You might want to note the following ports for later use:

- ▶ SOAP connector port: If you use the `addNode` command to federate a node to this deployment manager, you need to know this port number. This is also the port you connect to when using the `wsadmin` administration scripting interface.
- ▶ Administrative console port: You need to know this port to access the administrative console. When you turn on security, you need to know the Administrative console secure port.

13. If you would like to run the process as a Windows or Linux service, leave the check box selected and enter the values for the logon and startup type. Click **Next**.
14. Review the options that you have chosen. If you took the Typical path through the wizard, make sure that the default selections suit your needs. Click **Create** to create the profile.

15. The final window indicates the success or failure of the profile creation. If you have errors, check the log at:

install_root/logs/manageprofiles/profile_name_create.log

You can also find logs for individual actions stored in:

profile_root/logs

16. Click **Finish** to close the wizard and start the First Steps application.

17. Verify the installation. You can do this directly from the First Steps menu. The IVT process starts the deployment manager and checks the log file for warnings or errors on start.

18. Open the administrative console by selecting the option in the First Steps window, or by accessing its URL from a web browser:

`http://<dmgr_host>:<admin_console_port>/ibm/console`

Here is a sample URL in the address bar:

`http://localhost:9060/ibm/console/`

19. Log in and display the configuration from the console. You should be able to see the following items from the administrative console:

- a. Cell information: Select **System administration** → **Cell**.
- b. Deployment manager: Select **System administration** → **Deployment manager**.
- c. Deployment manager node: Select **System administration** → **Nodes**.
- d. The default node group: Select **System administration** → **Node groups**.

Note that at the completion of this process you do not have:

- a. A node agent:

Node agents reside on nodes with managed application servers. You do not see node agents appear until you federate a node to the cell.

- b. Application servers

Working with deployment managers: For information about starting, stopping, and viewing deployment managers, see Chapter 7, “Administration of WebSphere processes” on page 265.

3.3.5 Creating a cell profile

Table 3-2 shows a summary of the options you have during a cell profile creation. Using this option actually creates two distinct profiles: a deployment manager profile and an application server profile. The application server profile is federated to the cell. The options you see are a reflection of the options you would see if you were creating the individual profiles versus a cell. The Profile Management Tool windows give you basically the same options that you would see if you created a deployment manager and then an application server.

Table 3-2 Cell profile options

Typical	Advanced
The administrative console and default application are deployed by default.	You have the option to deploy the administrative console (recommended), the default application, and the sample applications (if installed).

Typical	Advanced
The profile name for the deployment manager is Dmgrxx by default, where xx is 01 for the first deployment manager profile and increments for each one created. The profile is stored in <i>install_root</i> /profiles/Dmgrxx.	You can specify the profile name and its location.
The profile name for the federated application server and node is AppSrvxx by default, where xx is 01 for the first application server profile and increments for each one created. The profile is stored in <i>install_root</i> /profiles/AppSrvxx.	You can specify the profile name and its location.
Neither profile is made the default profile.	You can choose to make the deployment manager profile the default profile.
<p>The cell name is <host>Cellxx. The node name for the deployment manager is <host>CellManagerxx. The node name for the application server is <host>Nodexx. The host name is prefilled with your system's DNS host name.</p>	You can specify the cell name, the host name, and the profile names for both profiles.
You can enable administrative security (yes or no). If you select yes, you are asked to specify a user name and password that will be given administrative authority.	
TCP/IP ports default to a set of ports not used by any profiles in this WebSphere installation instance.	You can use the recommended ports for each profile (unique to the installation), use the basic defaults, or select port numbers manually.
(Windows) The deployment manager will be run as a service.	(Windows) You can choose whether the deployment manager will run as a service.
Does not create a web server definition.	Allows you to define an external web server to the configuration.

3.3.6 Creating a custom profile

A custom profile defines an empty node on a system. The purpose of this profile is to define a node on a system to be federated to a cell for management through a deployment manager.

As you create the profile, you have the option to federate the node to a cell during the wizard, or to simply create the profile for later federation. Before you can federate the custom profile to a cell, you need to have a running deployment manager.

Note: With other profiles, you have the option of registering the processes as Windows services. This does not appear as an option when you create a custom profile.

This section takes you through the steps of creating a custom profile using the WebSphere Customization Toolbox Profile Management Tool. It shows the steps in the Advanced path through the profile creation.

To create the profile, complete the following steps:

1. Start the WebSphere Customization Toolbox. You will see the Profile Management Tool window.
2. Click **Create**.

3. Click **Custom profile**. Click **Next**.
4. Select whether to take the typical settings or to go through the advanced windows. The options you see next depend on the path you take.
 - If Typical is selected, then you only see one more option (to enable security).
 - If Advanced is selected, continue with the following steps.
5. Enter a unique name for the profile or accept the default. The profile name becomes the directory name for the profile files.
Select the box if you want this directory to be the default profile for receiving commands. Click **Next** (Figure 3-23).

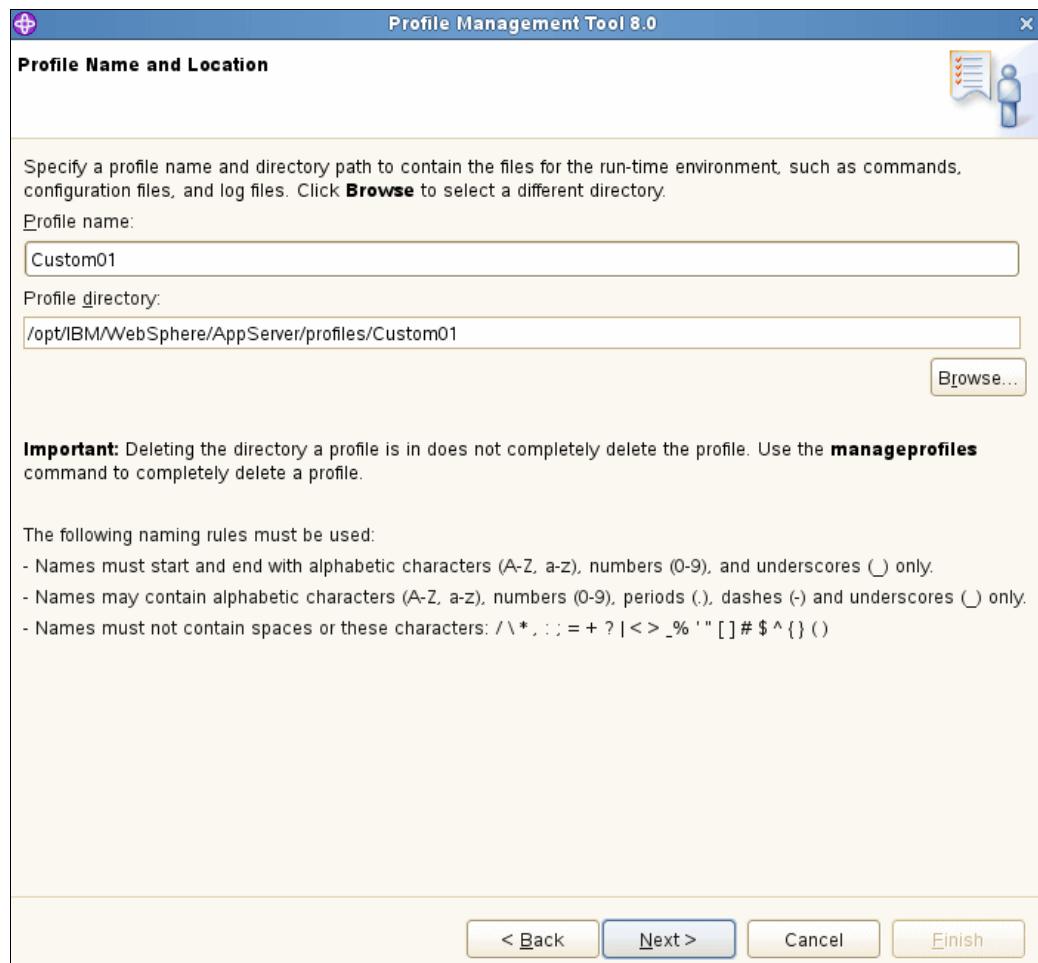


Figure 3-23 Creating a custom profile - Enter name and location

6. Enter the new node name and the system host name. The node name defaults to the host name of your system. The wizard recognizes if there are existing nodes in the installation and takes this situation into account when creating the default node name. Click **Next** (Figure 3-24).

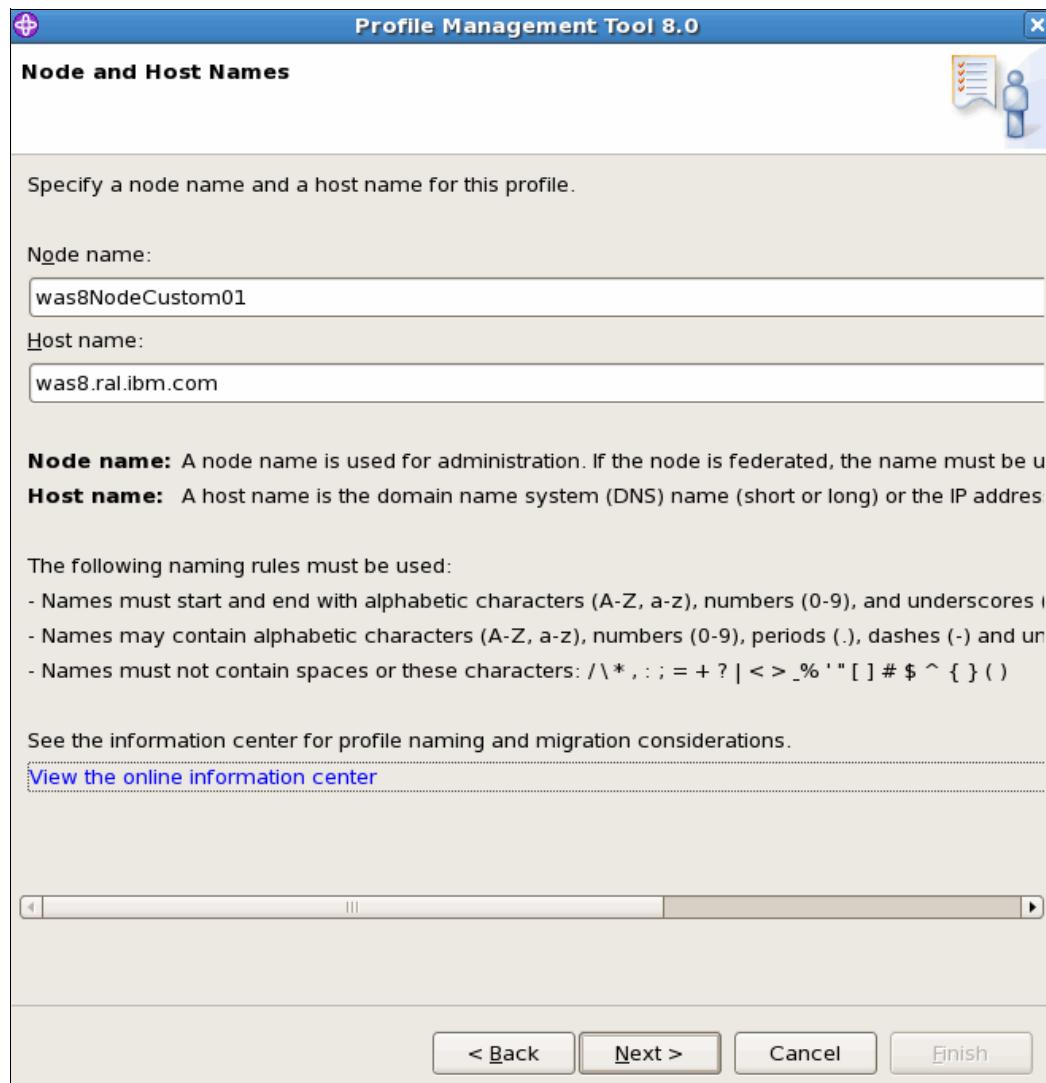


Figure 3-24 Creating a custom profile - Enter host and node names

7. If you would like to federate the new node defined by the profile to a cell as part of the wizard process, leave the **Federate this node later** check box clear and enter the host name and SOAP connector port for the deployment manager. Enter the user ID and password of the administrator ID for the deployment manager. Click **Next**.

When you click **Next**, a connection is attempted to the deployment manager. If you have entered any of these values incorrectly, you will be able to correct them (Figure 3-25).

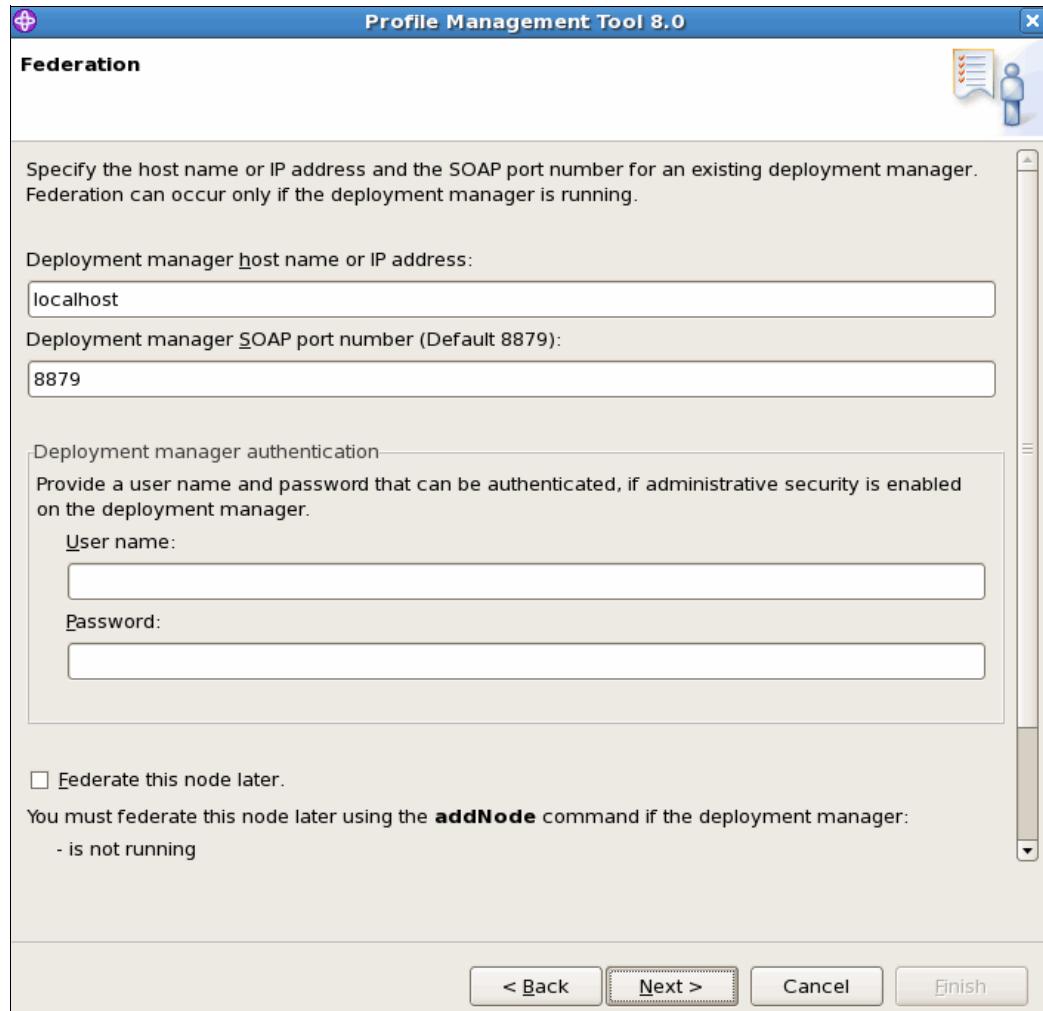


Figure 3-25 Creating a custom profile - Federate later

8. Review the options you have chosen. If you took the Typical path through the wizard, make sure that the default selections suit your needs.

Click **Create** to create the profile.

9. The final window indicates the success or failure of the Custom profile creation.

If you have errors, check the log at:

install_root/logs/manageprofiles/profile_name_create.log

Note that you will have to click **Finish** on the window to unlock the log.

You can also find logs for individual actions stored in:

profile_root/logs

Note: Custom profiles do not create a server process, so you cannot verify, stop, or start the profile. The only reason to launch the First Steps menu is if you want to link to the Information Center or launch the migration wizard.

10. The federation process creates a node agent for the new node, federates it to the cell, and starts the node agent. If you federated the custom profile, open the deployment manager administrative console and view the node and node agent:
- Select **System Administration** → **Nodes**. You should see the new node.
 - Select **System Administration** → **Node agents**. You should see the new node agent.
 - Select **System Administration** → **Cells**. Click the **Topology** tab and expand the view. From here, you can see a tree diagram of the cell (Figure 3-26).

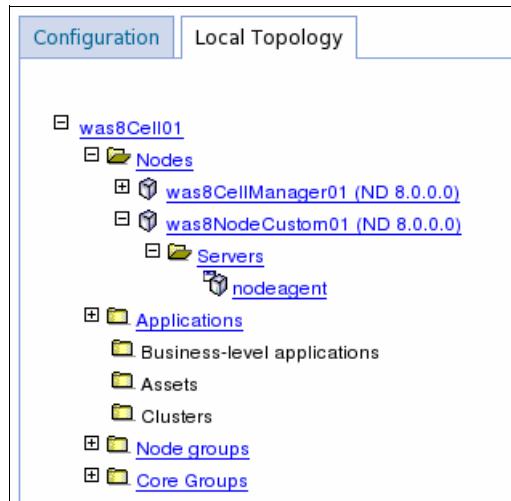


Figure 3-26 Topology view of a cell

If you have not federated the node, proceed to 3.3.7, “Federating nodes to a cell” on page 112. Otherwise, you can continue by defining an application server on the new node (see 7.4.1, “Creating an application server” on page 272).

3.3.7 Federating nodes to a cell

A custom profile defines a node that can be added to a cell. The **addNode** command is used to federate a node in a custom profile to a cell.

A stand-alone application server can also be federated to a cell with the **addNode** command, or from the deployment manager administrative console. The administrative console invokes the **addNode** command on the target system.

When you federate a node, the node name from the federated node is used as the new node name and must be unique in the cell. If the name of the node that you are federating already exists, the **addNode** operation will fail.

The **addnode** command

The **addNode** command is run from the *install_root/bin* or *profile_root/bin* directory of the installation for the profile that will be federated (that is, from the custom profile or application server profile installation).

Addnode command syntax

The syntax of the **addNode** command is shown in Example 3-1.

Example 3-1 The addNode command syntax

```
Usage: addNode dmgr_host [dmgr_port] [-conntype <type>] [-includeapps]
      [-includebuses] [-startingport <portnumber>] [-portprops <qualified-filename>]
      [-nodeagentshortname <name>] [-nodegroupname <name>]
      [-registerservice] [-serviceusername <name>] [-servicepassword <password>]
      [-coregroupname <name>] [-noagent] [-statusport <port>] [-quiet] [-nowait]
      [-logfile <filename>]
      [-replacelog] [-trace] [-username <username>] [-password <pwd>]
      [-localusername <localusername>] [-localpassword <localpassword>]
      [-profileName <profile>] [-excludesecuritydomains] [-asExistingNode]
      [-help]
```

Where:

- ▶ **dmgr_host, -username, -password**

This command connects to the deployment manager, so you have to specify the deployment manager host name and a user ID and password with administrative privileges on the deployment manager.

- ▶ **dmgr_port, -conntype**

The default is to connect to the deployment manager using SOAP and port 8879. If your deployment manager was defined with this port, you do not need to specify anything. If not, you can specify the correct port, or you can use RMI as the connection type.

For SOAP connections, the port defined as the SOAP_CONNECTOR_PORT number on the deployment manager must be specified. If you choose to use an RMI connection instead, the ORB_LISTENER_ADDRESS port must be specified. You can see these settings in the port list of the deployment manager in the administrative console.

Tip: Port numbers are also stored in the <profile_root>/properties/portdef.props file.

- ▶ **-startingport, -portprops <filename>**

The new node agent is assigned a range of ports automatically. If you want to specify the ports for the node rather than taking the default, you can specify a starting port using the **-startingport** parameter. The numbers are incremented from this number.

For example, if you specify 3333, the BOOTSTRAP_ADDRESS port will be 3333, CSIV2_SSL_MUTUALAUTH_LISTENER_ADDRESS will be 3334, and so on.

As an alternative, you can provide specific ports by supplying a file with the port properties.

- ▶ **-includeapps, -includebuses**

If you are federating an application server, you can keep any applications that are deployed to the server and you can keep any service integration bus definitions that have been created. The default is that these are not included during federation and are lost.

For more information about the **addNode** syntax and options, see the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.nd.multiplatform.doc/info/ae/ae/rxml_addnode.html

The **addNode** command performs the following actions:

1. Connects to the deployment manager process. This action is necessary for the file transfers performed to and from the deployment manager to add the node to the cell.
2. Attempts to stop all running application servers on the node.
3. Backs up the current stand-alone node configuration to the *profile_root/config/backup/base/* directory.
4. Copies the stand-alone node configuration to a new cell structure that matches the deployment manager structure at the cell level.
5. Creates a new local config directory and definition (*server.xml*) for the node agent.
6. Creates entries (directories and files) in the master repository for the new node's managed servers, node agent, and application servers.
7. Uses the FileTransfer service to copy files from the new node to the master repository.
8. Uploads applications to the cell only if the **-includeapps** option is specified.
9. Performs the first file synchronization for the new node. This action synchronizes data from the cell to the new node.
10. Corrects the node's **setupCmdLine** and **wsadmin** scripts to reflect the new cell environment settings.
11. Launches the node agent (unless **-noagent** is specified).

Federating a custom node to a cell

Note: You only have to do this action if you created a custom profile and chose *not* to federate it at the time. This action requires that you have a deployment manager profile and that the deployment manager is up and running.

To federate the node to the cell, complete the following steps:

1. Start the deployment manager.
2. Open a command window on the system where you created the custom profile for the new node. Switch to the *profile_root/bin* directory or *install_root/bin* directory.
3. Run the **addNode** command.

Example 3-2 shows an example of using the **addNode** command on a Windows system to add Node01 to the deployment manager using 8879 as the SOAP connector address. If administrative security is enabled, use the **-username** and **-password** arguments on the command line to provide the user ID and password. If you do not provide the arguments, you are prompted for them when using the **addNode** command.

Example 3-2 addNode command

```
/opt/IBM/WebSphere/AppServer/profiles/Custom01/bin>
/opt/IBM/WebSphere/AppServer/profiles/Custom01/bin>addNode.sh localhost 8879
ADMU0116I: Tool information is being logged in file
/opt/IBM/WebSphere/AppServer/profiles/Custom01/logs/addNode.log
ADMU0128I: Starting tool with the Custom01 profile
CWPKI0308I: Adding signer alias "CN=was8.ral.ibm.com, OU=Root Certificate, " to local
keystore "ClientDefaultTrustStore" with the following SHA digest:
AF:60:11:60:15:5B:B3:54:0C:46:84:1A:B5:DC:C6:A9:B6:DC:0F:0E
CWPKI0309I: All signers from remote keystore already exist in local keystore.
ADMU0001I: Begin federation of node was8NodeCustom01 with Deployment Manager
at localhost:8879.
```

ADMU0009I: Successfully connected to Deployment Manager Server: localhost:8879
ADMU0507I: No servers found in configuration under:
`/opt/IBM/WebSphere/AppServer/profiles/Custom01/config/cells/was8Node02Cell`
`/nodes/was8NodeCustom01/servers`
ADMU2010I: Stopping all server processes for node was8NodeCustom01
ADMU0024I: Deleting the old backup directory.
ADMU0015I: Backing up the original cell repository.
ADMU0012I: Creating Node Agent configuration for node: was8NodeCustom01
ADMU0014I: Adding node was8NodeCustom01 configuration to cell: was8Cell01
ADMU0016I: Synchronizing configuration between node and cell.
ADMU0018I: Launching Node Agent process for node: was8NodeCustom01
ADMU0020I: Reading configuration for Node Agent process: nodeagent
ADMU0022I: Node Agent launched. Waiting for initialization status.
ADMU0030I: Node Agent initialization completed successfully. Process id is:
5368

ADMU0300I: The node was8NodeCustom01 was successfully added to the was8Cell01 cell.

ADMU0306I: Note:
ADMU0302I: Any cell-level documents from the standalone was8Cell01 configuration have not been migrated to the new cell.
ADMU0307I: You might want to:
ADMU0303I: Update the configuration on the was8Cell01 Deployment Manager with values from the old cell-level documents.

ADMU0306I: Note:
ADMU0304I: Because -includeapps was not specified, applications installed on the standalone node were not installed on the new cell.
ADMU0307I: You might want to:
ADMU0305I: Install applications onto the was8Cell01 cell using wsadmin \$AdminApp or the Administrative Console.

ADMU0003I: Node wea01NodeCustom01 has been successfully federated.

/opt/IBM/WebSphere/AppServer/profiles/Custom01/bin>

-
4. Open the deployment manager administrative console and view the node and node agent:

- Select **System Administration → Nodes**. You should see the new node.
- Select **System Administration → Node agents**. You should see the new node agent and its status.

The node is started as a result of the federation process. If it does not appear to be started in the console, you can check the status from a command window on the node system:

```
cd profile_root\bin
serverStatus -all
```

If you find that it is not started, start it with this command:

```
cd profile_root\bin
startNode
```

For more information about managing nodes, see 7.5, “Working with nodes in a distributed environment” on page 301.

Creating application servers on the new node: The custom profile does not automatically give you an application server. You can complete the steps in 7.4.1, “Creating an application server” on page 272 to create a new server after the custom profile has been federated to a cell.

Federating an application server profile to a cell

If you are using the administrative console to federate an application server, keep in mind the following considerations:

- ▶ Both the deployment manager and the application server must be running.
- ▶ You need to be logged into the console with an ID that has administrator privileges.
- ▶ The command connects to the application server. This action requires you to specify the application server host name and a user ID that can connect to the server. In turn, the node has to connect to the deployment manager. Specify a user ID and password for this connection.
- ▶ You need to specify the host name, JMX connection type, and port number to use to connect to the application server. The JMX connection type can be SOAP or RMI. The default is a SOAP connection using port 8880.

To federate an application server profile to a cell (Figure 3-27 on page 117), complete the following steps:

1. Ensure that the application server and deployment manager are running.
2. Open the deployment manager administrative console.
3. Click **System Administration → Nodes → Add Node**.
4. Click **Managed node** and click **Next**.
5. Enter the host name and SOAP connector port of the application server profile.

If you want to keep the sample applications and any other applications you have installed, select the **Include applications** check box.

Enter the administrator user ID and passwords for both the application server and the deployment manager.

Node connection

- * Host: was8.ral.ibm.com
- * JMX connector type: SOAP
- * JMX connector port: 8880
- Application server user name: wasadmin
- Application server password:
- * Deployment manager user name: wasadmin
- * Deployment manager password:
- Config URL: file://\${USER_INSTALL_ROOT}/properties/sas.client.props

Options

- Include applications
- Include buses

Starting port

- Use default
- Specify

Port number:

Figure 3-27 Adding a stand-alone application profile to a cell

Click **OK**.

6. If the node is a Windows node, you have the opportunity to register the new node agent as a Windows service. Make your selection and click **OK**.

The federation process stops the application server. It creates a new node agent for the node, and adds the node to the cell. The federation process then starts the node agent, but not the server.

You can now display the new node, node agent, and application server from the console. You can also start the server from the console.

At the completion of the process:

- The profile directory for the application server still exists and is used for the new node.
- The old cell name for the application server has been replaced in the profile directory with the cell name of the deployment manager:
profile_root/config/cells/dmgr_cell
- A new entry in the deployment manager profile directory has been added for the new node:
dmgr_profile_root/config/cells/dmgr_cell/nodes/federated_node

- ▶ An entry for each node in the cell is added to the application server profile configuration. Each node entry contains the `serverindex.xml` file for the node:
`profile_root/config/cells/dmgr_cell/nodes/federated_node`
In turn, an entry for the new node is added to the `nodes` directory for each node in the cell with a `serverindex.xml` entry for the new node.

Example 3-3 shows an example of using the `addNode` command to add an application server profile to a cell. The command specifies the deployment manager host (T60) and the SOAP connector port (8882). Applications currently installed on the application server will still be installed on the server after federation.

Example 3-3 The addNode command usage examples

```
C:\WebSphereV8\AppServer\bin>addNode t60 8882 -profileName node40b -includeapps -username admin -password adminpwd
ADMU0116I: Tool information is being logged in file
          C:\WebSphereV8\AppServer\profiles\node40b\logs\addNode.log
ADMU0128I: Starting tool with the node40b profile
CWPKI0308I: Adding signer alias "default_2" to local keystore
              "ClientDefaultTrustStore" with the following SHA digest:
              9D:99:04:63:97:8C:C0:76:19:46:5A:C4:C0:35:20:FE:DE:21:FD:29
ADMU0001I: Begin federation of node node40b with Deployment Manager at
          t60:8882.
ADMU0009I: Successfully connected to Deployment Manager Server: t60:8882
ADMU0505I: Servers found in configuration:
ADMU0506I: Server name: server40b1
ADMU2010I: Stopping all server processes for node node40b
ADMU0512I: Server server40b1 cannot be reached. It appears to be stopped.
ADMU0024I: Deleting the old backup directory.
ADMU0015I: Backing up the original cell repository.
ADMU0012I: Creating Node Agent configuration for node: node40b
ADMU0120I: isclite.ear will not be uploaded since it already exists in the target
repository.
ADMU0120I: DefaultApplication.ear will not be uploaded since it already exists in the
target repository.

ADMU0016I: Synchronizing configuration between node and cell.
ADMU0018I: Launching Node Agent process for node: node40b
ADMU0020I: Reading configuration for Node Agent process: nodeagent
ADMU0022I: Node Agent launched. Waiting for initialization status.
ADMU0030I: Node Agent initialization completed successfully. Process id is: 5512
ADMU0505I: Servers found in configuration:
ADMU0506I: Server name: nodeagent
ADMU0506I: Server name: server40b1

ADMU0308I: The node node40b and associated applications were successfully added to the
Cell140 cell.

ADMU0306I: Note:
ADMU0302I: Any cell-level documents from the standalone Cell140 configuration have not been
migrated to the new cell.
ADMU0307I: You might want to:
ADMU0303I: Update the configuration on the Cell140 Deployment Manager with values from the
old cell-level documents.

ADMU0003I: Node node40b has been successfully federated.
```

3.3.8 Creating an administrative agent profile

This section takes you through the steps of creating the administrative agent profile using the WebSphere Customization Toolbox Profile Management Tool. It shows the steps in the Advanced path through the profile creation.

To create the profile, complete the following steps:

1. Start the WebSphere Customization Toolbox. You see the **Profile Management Tool** window.
2. Click **Create**.
3. Click **Management**. Click **Next**.
4. Click **Administrative agent** and click **Next** (Figure 3-28).

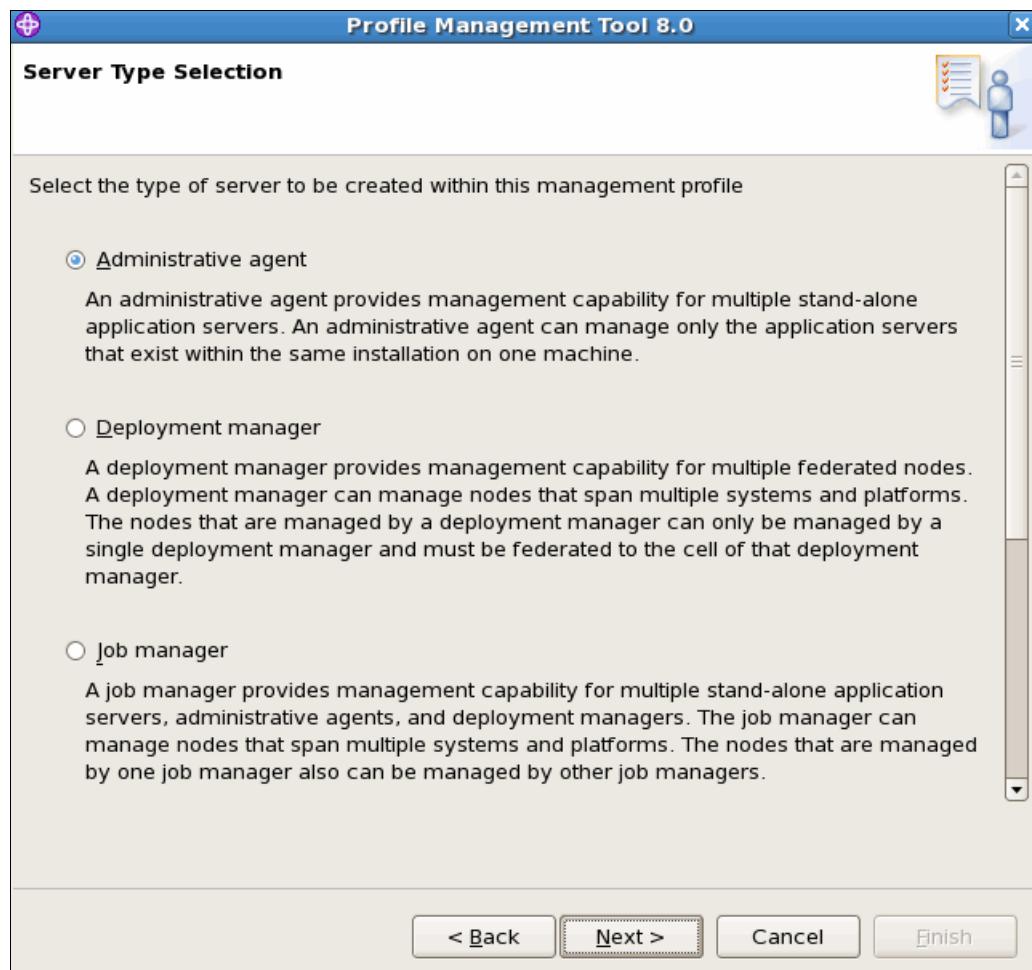


Figure 3-28 Administrative agent option

5. Select the typical or advanced path. Click **Next**.
 - If Typical is selected, then you only see one more option (to enable security).
 - If Advanced is selected, you see the next step.

6. Select the option to install the administrative console (Figure 3-29). Click **Next**.

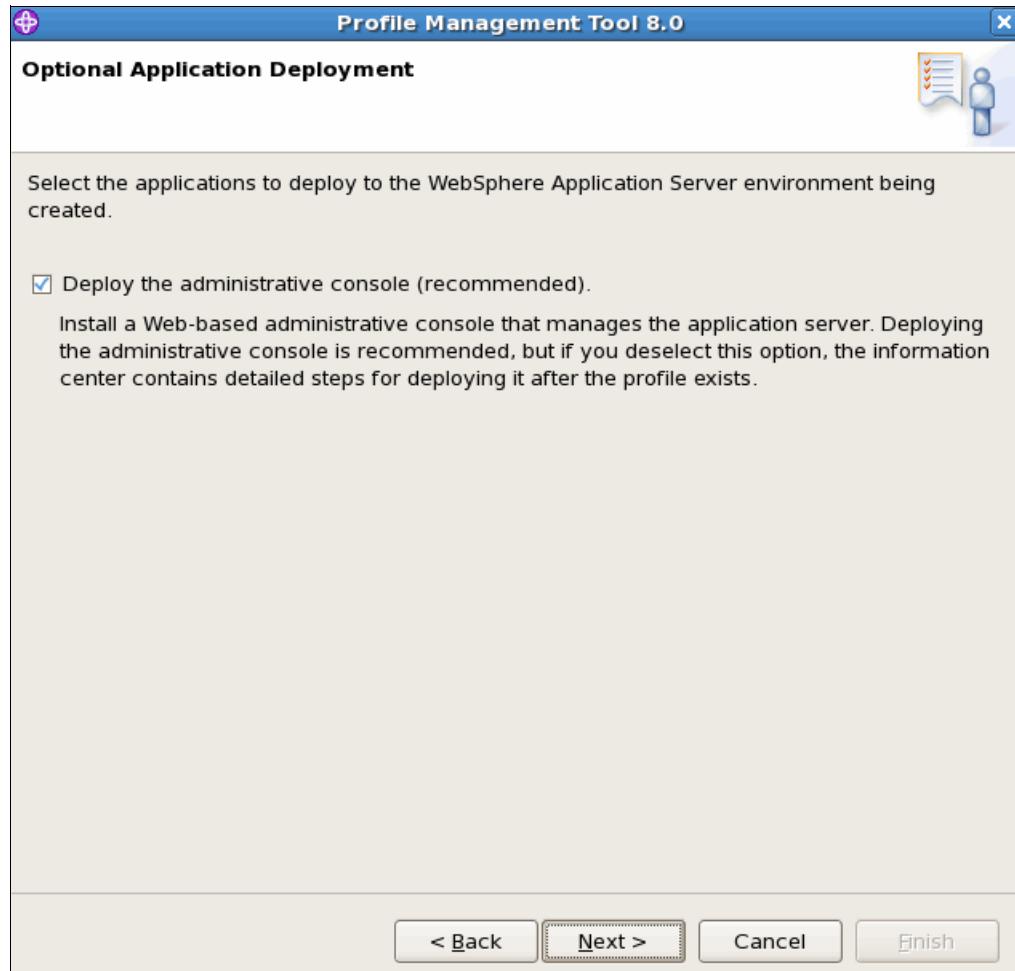


Figure 3-29 Deploy administrative console

7. Enter a unique name for the profile or accept the default. The profile name becomes the directory name for the profile. (Figure 3-30). Click **Next**.

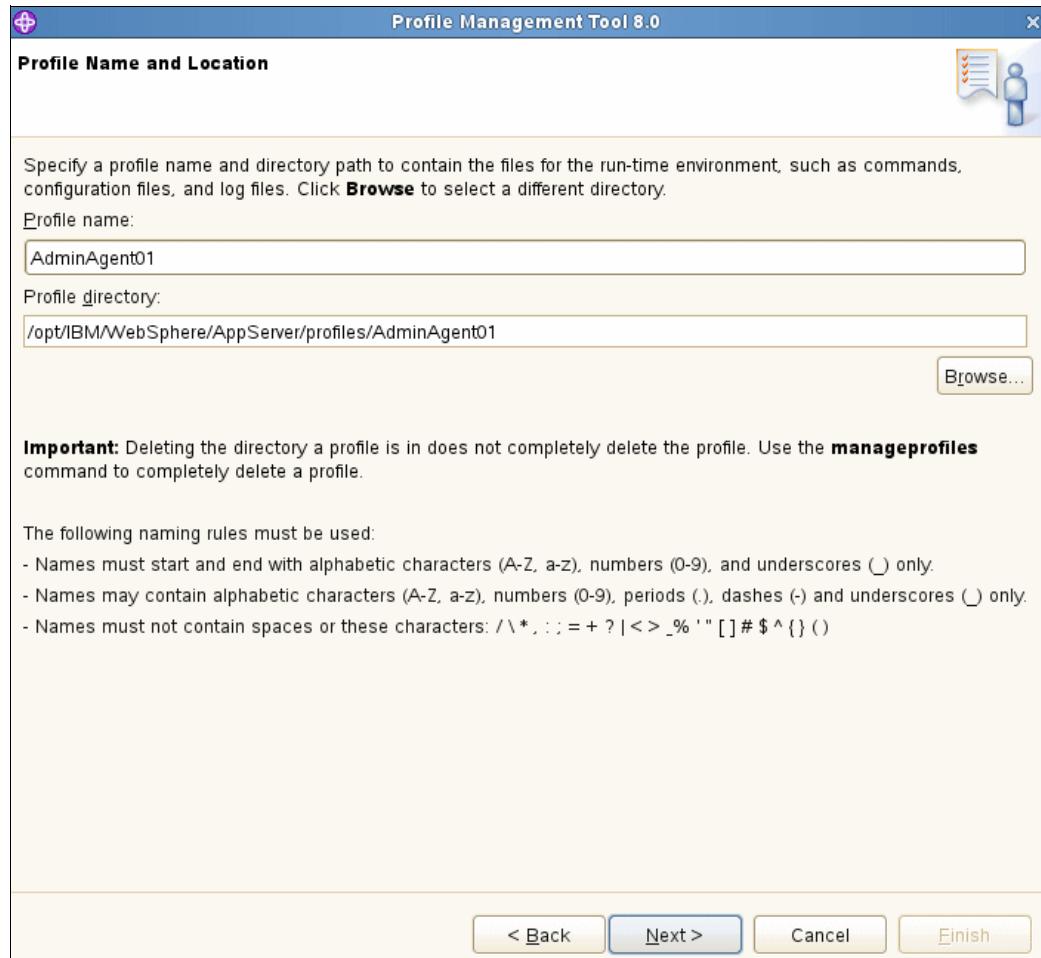


Figure 3-30 Enter the name and location of the profile

Note: The first profile created is the default profile. All commands are executed against the default profile. After creating the first profile, this window will change slightly. It now shows an option for making this new profile the default profile. Select the check box if you want this directory to be the default profile for receiving commands

8. Enter the new node name and the system host name. The node name defaults to the host name of your system. The wizard recognizes if there are existing nodes in the installation and takes this setting into account when creating the default node name (Figure 3-31). Click **Next**.

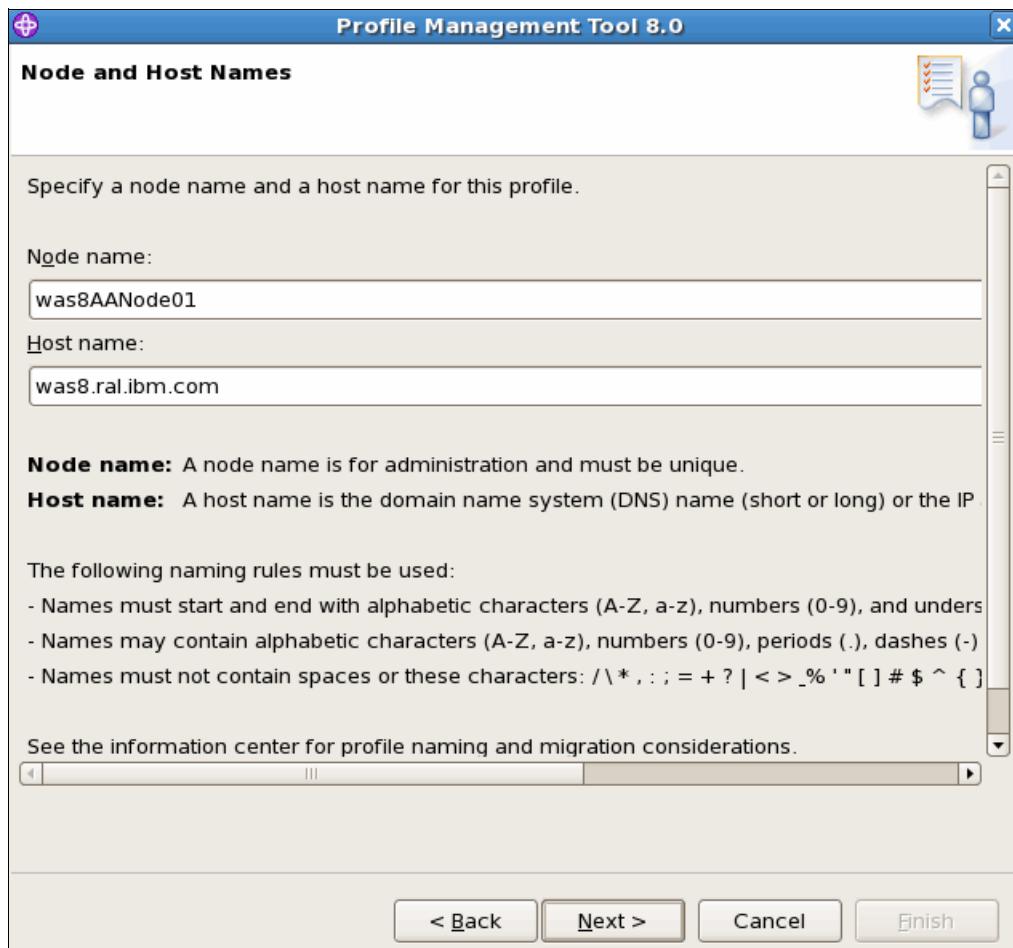


Figure 3-31 Enter host and node names

9. Choose whether to enable administrative security. If you enable security here, you are asked for a user ID and password that will be added to a file-based user registry with the Administrative role. Click **Next**.
10. Elect to either create new default personal and root signing certificates or to import them. Click **Next**.
11. Review and modify the certificate information as needed. Click **Next**.

12. The wizard presents a list of TCP/IP ports for use by the application server. If you already have existing profiles on the system (within this installation), this setting will be taken into account when the wizard selects the port assignments, but you should verify that these ports will be unique on the system and alter them if required (Figure 3-32). Click **Next**.

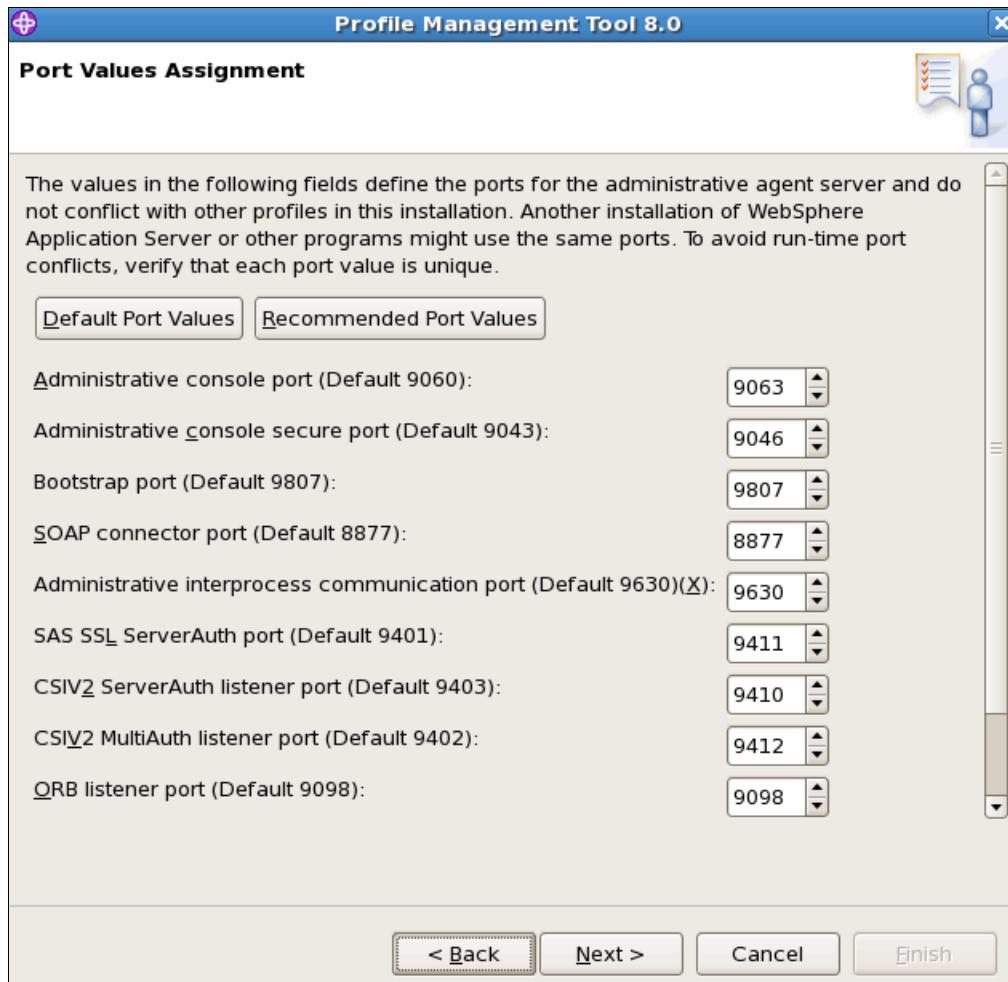


Figure 3-32 Select ports

Note the administrative console and SOAP connector ports for future use.

13. On Windows and Linux systems, select whether to run the administrative agent process as a service. Click **Next**.
14. Review the options you have chosen and click **Create** to create the profile.
15. After the wizard has finished, you are presented with the window indicating the success or failure of the process.
If you have errors, check the log at:
`install_root/logs/manageprofiles/profile_name_create.log`
You can also find logs for individual actions stored in:
`profile_root/logs`
16. Click **Finish** to close the wizard and start the First Steps application.
17. Run the Installation verification test to ensure the installation was successful. The test will start the administrative agent.

18. Open the administrative agent administrative console from the First Steps menu, or using the following URL:

`http://admin_host:port/ibm/console/`

Where port is the administrative console port selected during profile creation.

19. View the administrative agent by clicking **System Administration → Administrative agent** (Figure 3-33).

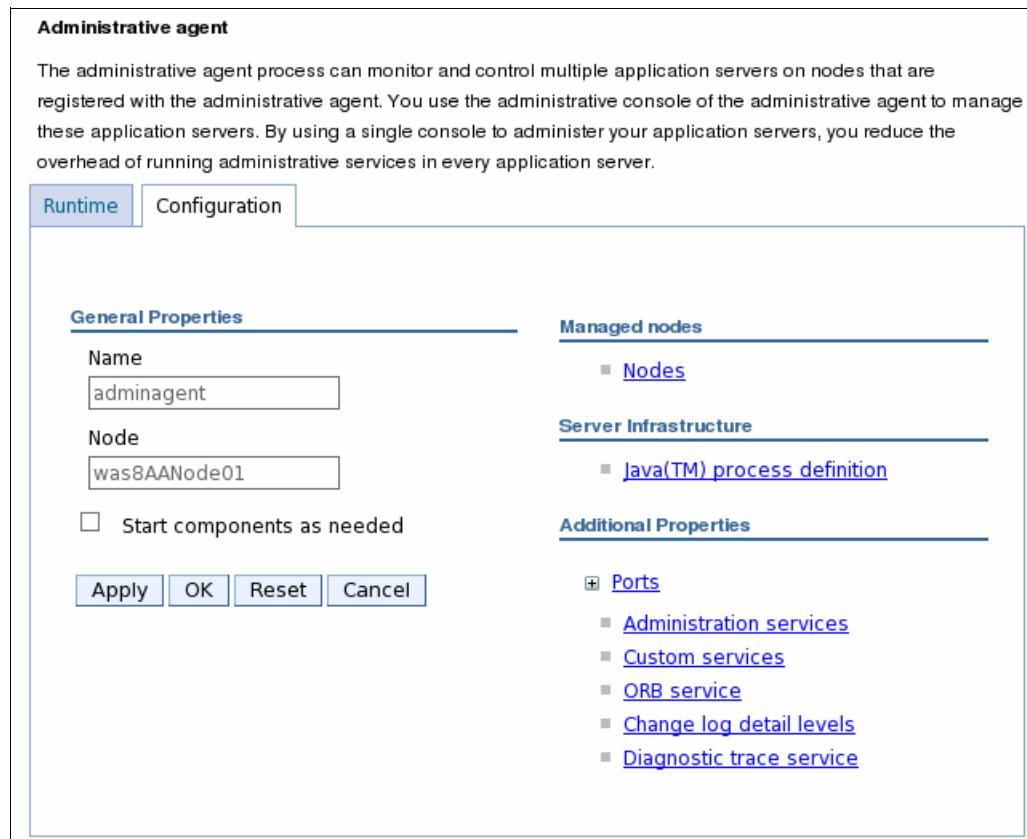


Figure 3-33 Administrative agent details in the administration console

Nodes that are registered to the administrative agent can be viewed by clicking the **Nodes** link under Managed nodes. The list is initially empty. To register stand-alone application server nodes to the administrative agent, see 3.3.10, “Registering nodes to an administrative agent” on page 125.

3.3.9 Creating a job manager profile

This section takes you through the steps of creating the job manager using the WebSphere Customization Toolbox Profile Management Tool. It shows the steps in the Advanced path through the profile creation. The steps for creating a job manager profile are exactly the same as those for the administrative agent profile.

To create the profile, complete the following steps:

1. Start the WebSphere Customization Toolbox. You see the Profile Management Tool window.
2. Click **Create**.
3. Select **Management**. Click **Next**.
4. Select **Job manager** and click **Next** (Figure 3-34).

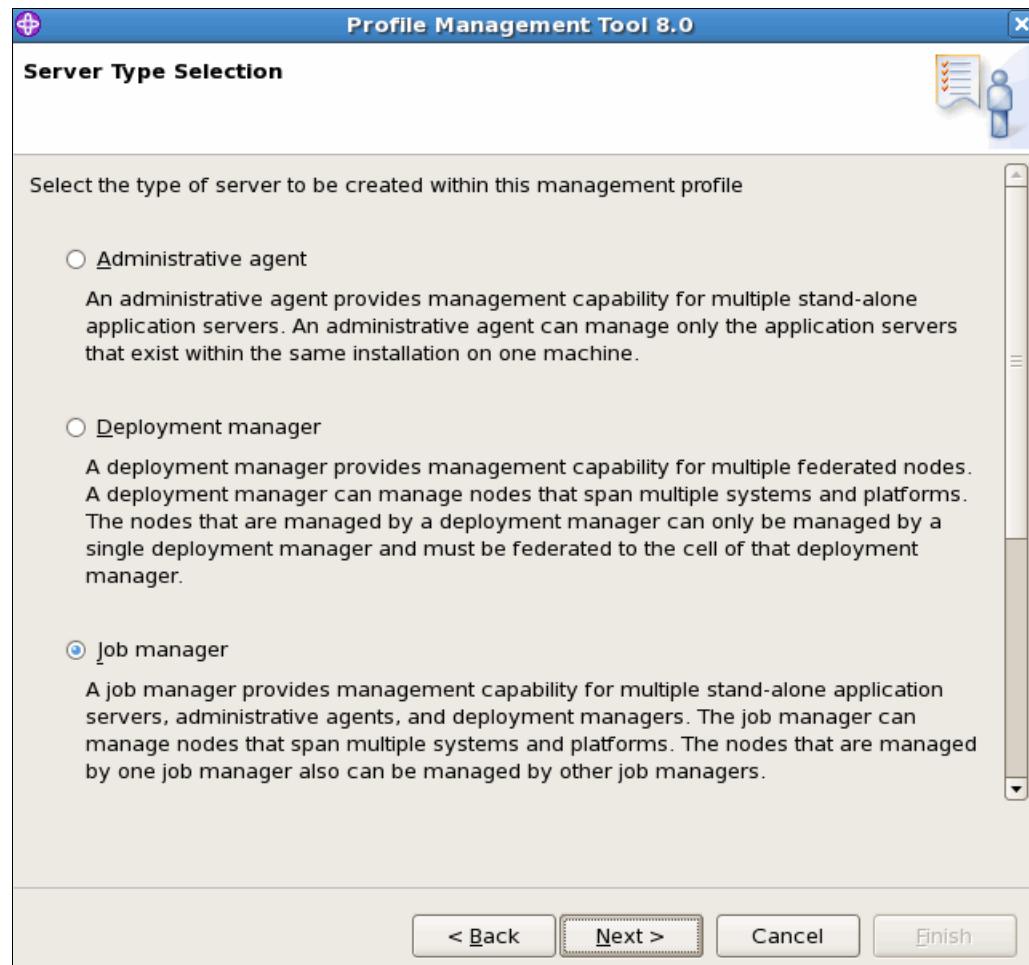


Figure 3-34 Job manager type selection

All further steps are exactly as described in the administrative agent profile.

You can open the job manager console from the First Steps menu, or by using the following URL:

`http://job_manager_host:port/ibm/console/`

Where port is the administrative console port selected during profile creation.

3.3.10 Registering nodes to an administrative agent

An administrative agent provides a single interface to unfederated application server nodes (stand-alone application server profiles).

Notes for use:

- ▶ The administrative agent and application servers must be on the same machine or sysplex.
- ▶ The administrative agent must be started before running **registerNode**.

You can only run the command on an unfederated stand-alone application server. When you run the command, the node for the stand-alone server is converted into a node that the administrative agent manages.

The **registerNode** command is used to register a node with an administrative agent. The syntax of the command is:

```
registerNode [options]
```

The options can be displayed using the **-help** parameter (Example 3-4).

Example 3-4 registerNode options

```
/opt/IBM/WebSphere/AppServer/bin]registerNode.sh -help
Usage: registerNode -profilePath <path to the base profile to be registered>
      [-host <adminagent host>] [-connType <SOAP | RMI | JSR160RMI | IPC>]
      [-port <adminagent JMX port>] [-name <managed node name>]
      [-openConnectors <SOAP,IPC,...>] [-username <adminagent user name>]
      [-password <adminagent password>] [-nodeusername <base node user name>]
      [-nodepassword <base node password>] [-profileName <adminagent profile name>]
      [-portsFile <jmx ports filename>] [-trace] [-help]
```

Note: In WebSphere Application Server V8, the RMI connector type option is deprecated. You should switch to the JSR160RMI connector type.

For details about the **registerNode** command, see the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=/com.ibm.websphere.nd.doc/info/ae/ae/ragt_registerNode.html

You can enter the command directly. For example, to register the application server defined by the SASrv40 profile to the administrative agent adminAgnt40, you would enter the following command:

```
registerNode.bat -profileName adminAgnt40 -profilePath
"/opt/IBM/WebSphere/AppServer/profiles/SASrv40" -host t60 -connType SOAP -port
8877 -username wasadmin -password wasadmin08 -nodeusername wasadmin -nodepassword
wasadmin08
```

Alternatively, you could create an execution file with the **registerNode** command. For example:

1. Create a file with contents similar to Example 3-5. The name of the file in this example is `registerAppSrv01WithAdminAgent.bat`. Modify the set statements to match your environment.

Example 3-5 Sample registerNode .bat file contents

```
echo on
set WAS_HOME=C:\WebSphere\AppServer
set appAdminUser=wasadmin
```

```

set appAdminPassword=wasadmin08
set adminAgentAdminUser=wasadmin
set adminAgentAdminPassword=wasadmin08
set adminAgentProfileName=AdminAgent01
set adminAgentHostName=localhost
set adminAgentSoapPort=8877
set baseProfileName=AppSrv03

cd %WAS_HOME%\profiles\%adminAgentProfileName%\bin

registerNode.bat -profileName %adminAgentProfileName% -profilePath
"%WAS_HOME%\profiles\%baseProfileName%" -host %adminAgentHostName% -conntype
SOAP -port %adminAgentSoapPort% -username %adminAgentAdminUser% -password
%adminAgentAdminPassword% -nodeusername %appAdminUser% -nodepassword
%appAdminPassword%

```

2. Copy this file to the *adminAgnt_profile_root/bin* directory and run the file. You see results similar to Example 3-6. Check the final message to make sure that the node was registered.

Example 3-6 Sample execution to register a node to the Administration profile

```

echo on

set WAS_HOME=C:\WebSphere\AppServer
set appAdminUser=wasadmin
set appAdminPassword=wasadmin08
set adminAgentAdminUser=wasadmin
set adminAgentAdminPassword=wasadmin08
set adminAgentProfileName=AdminAgent01
set adminAgentHostName=localhost
set adminAgentSoapPort=8877
set baseProfileName=AppSrv01

registerNode.bat -profilePath "C:\WebSphere\AppServer\profiles\%baseProfileName%" -host
localhost -conntype SOAP -port 8877 -username %appAdminUser% -password %appAdminPassword%
-nodeusername %adminAgentAdminUser% -nodepassword %adminAgentAdminPassword%
ADMU0116I: Tool information is being logged in file
          C:\WebSphere\AppServer\profiles\AdminAgent01\logs\registerNode.log
ADMU0128I: Starting tool with the AdminAgent01 profile
ADMU8053I: Successfully connected to AdminAgent Server: localhost:8877
ADMU8002I: Exchanging signers between adminagent and node with path
          C:\WebSphere\AppServer\profiles\%baseProfileName%
ADMU8007I: Exchanged signers successfully.
ADMU0505I: Servers found in configuration:
ADMU0506I: Server name: server1
ADMU2010I: Stopping all server processes for node was8Node03
ADMU0512I: Server server1 cannot be reached. It appears to be stopped.
ADMU8010I: Begin registration of Application Server with path
          C:\WebSphere\AppServer\profiles\%baseProfileName%
ADMU0024I: Deleting the old backup directory.
ADMU8004I: Backing up the original config directory of the node will be
registered.
ADMU8037I: Backing up the original wsadmin.properties file of the node will be
registered.
ADMU8036I: Registering the node with an AdminAgent.

```

ADMU8042I: Node has been successfully registered.
ADMU8040I: The administrative agent is initializing the administrative subsystem for the registered node.
ADMU8014I: The administrative subsystem for registered node has been successfully initialized.
ADMU8041I: The administrative agent is starting the administrative subsystem for the registered node.
ADMU8015I: The administrative subsystem for registered node has been successfully started.
ADMU0505I: Servers found in configuration:
ADMU0506I: Server name: server1
ADMU8012I: Application Server with path
C:\WebSphere\AppServer\profiles\AppSrv01 has been successfully registered.

3. The next time you log in to the administrative agent console, you have the option to select a node to administer. In this case, you can select between the administrative agent (the top choice) and the new node you registered.

Select the administrative agent to view the new configuration (Figure 3-35).

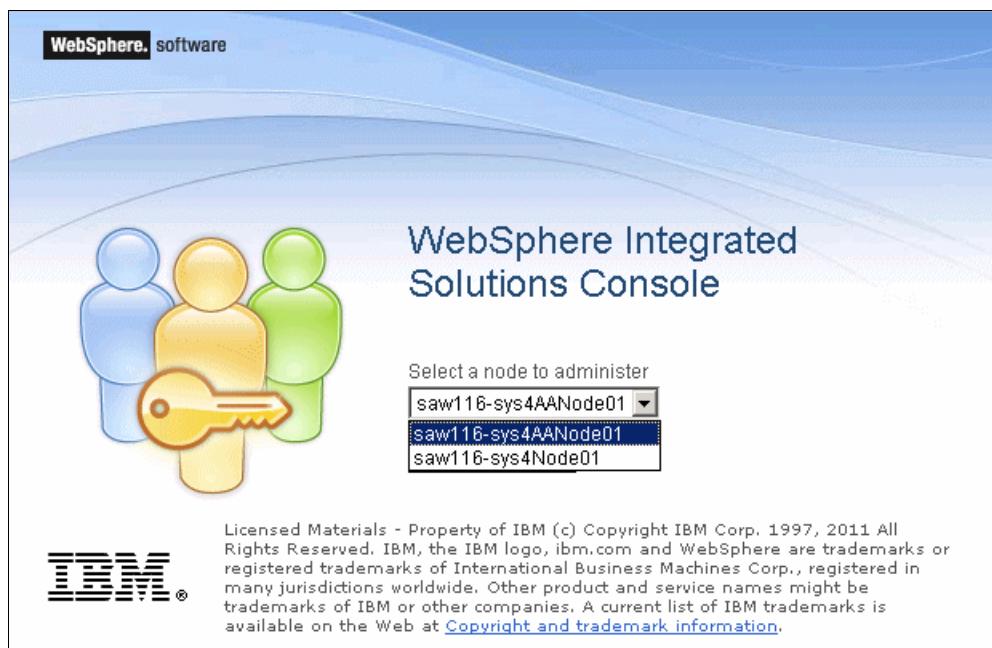


Figure 3-35 Option to pick which local application server to administer

4. Click **System administration** → **Administrative agent** → **Nodes**. This action shows that this node is now registered with the administrative agent (Figure 3-36).

The screenshot shows a user interface titled "Administrative agent > Nodes". A descriptive text at the top explains that it lists managed nodes registered to the administrative agent, with options to register or unregister from a job manager. Below this is a toolbar with icons for preferences, registration, and unregistration. A search bar allows filtering by "Name" or "Unique ID". A table displays a single registered node: "saw116-sys4Node01" (Name) and "AppSrv01-BASE-39c265c9-f61f-4246-821f-6ae46aad4fe3" (Unique ID). A note below the table states "Total 1".

Figure 3-36 Node registered with the administrative agent

Note: You can register an application server node with the administrative agent or federate the node with a deployment manager, but not both.

3.3.11 Deregistering a node from the administrative agent

To deregister a node from the administrative agent, run **deregisterNode** from the *adminAgnt_profile_root/bin* directory (see Example 3-7).

Example 3-7 deregisterNode command

```
deregisterNode.bat -connType SOAP -port 8877 -profilePath
C:\WebSphere\AppServer\profiles\AppSrv01 -username wasadmin -password wasadmin08
```

3.3.12 Registering an administrative agent node with a job manager

This section describes the steps to register nodes within the administrative agent with a job manager:

1. Log on to the administrative agent node (Figure 3-37).

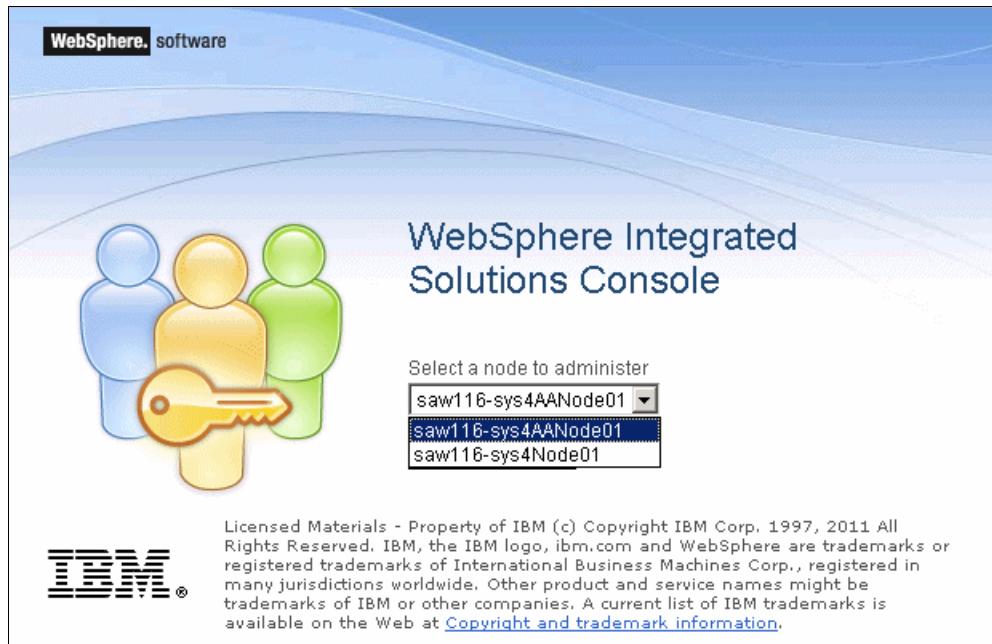


Figure 3-37 Option to pick which local application server to administer

2. Click **System administration** → **Administrative agent** (Figure 3-38).

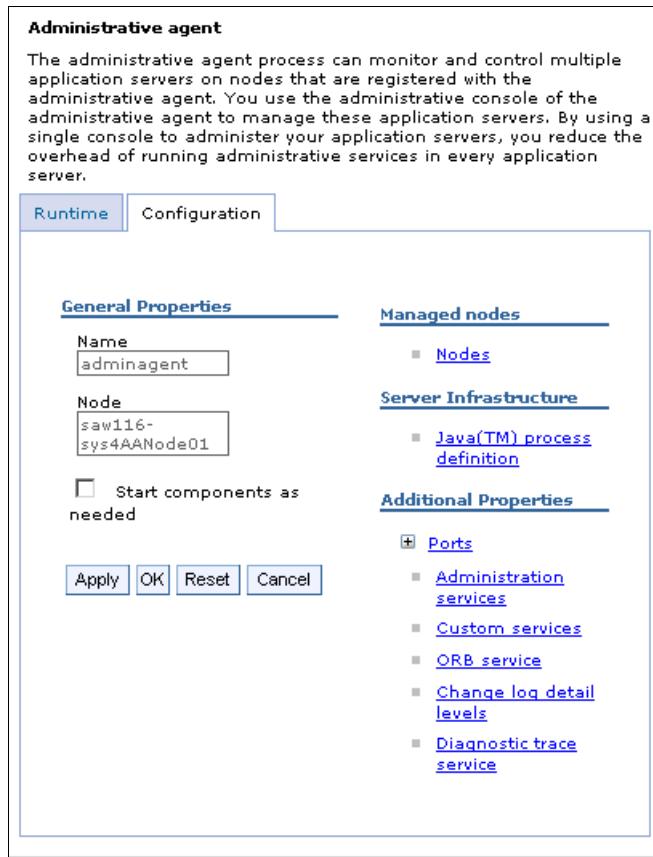


Figure 3-38 Administrative agent details

3. Click **Nodes** and select the node that you want to register with the job manager (Figure 3-39).



Figure 3-39 Select which node will be registered with the job manager

Note: You can register some or all of the nodes with the job manager, or you can register some nodes with one job manager and other nodes with another job manager. You can also register all of the nodes with multiple different job managers.

Click **Register with Job Manager**.

4. Enter the information required to connect to the job manager, including the host name, administrative host port, and user ID and password.

If the node name you are registering is already in use by the job manager, you can enter an alias for the node (Figure 3-40).

Administrative agent > Nodes > Register with Job Manager

Register a managed node with job manager. If you are using the administrative agent administrative console, you register a node that is already registered to the administrative agent. If you are using the deployment manager administrative console, you register the deployment manager. Specify an alias if the managed node name is in use by another node. Use the administrative host port of the job manager, which defaults to 9943 when security is enabled.

General Properties

* Managed node name

Alias

Host name

Port

User name

Password

Confirm password

Figure 3-40 Detailed options for registering a node with a job manager

Click **OK** to register the node.

5. Log in to the job manager and click **Jobs** → **Targets** to see the newly registered node (Figure 3-41).

The screenshot shows the 'Targets' panel in a web-based administrative interface. At the top, there's a search bar with fields for 'Target name' (set to '*'), 'Job type' (set to '='), and 'Unique identifier' (set to '='). Below the search bar are 'Advanced find options' and 'Resources' filters. The 'Resources' section includes a 'With' dropdown, several dropdown menus for filtering, and a '+' button. A small icon of a host server is also present. Under 'Maximum results', the value '50' is entered. At the bottom of the search area are 'Find' and 'Reset' buttons, and a message indicating 'Retrieved 1 of 1 results.' Below this is a toolbar with buttons for 'New Host...', 'Display Resources', and 'Delete Host'. The main content area displays a table with one row. The table has columns for 'Select' (checkbox), 'Target name' (dropdown set to 'Target name'), and 'Version' (dropdown set to 'Version'). The single row contains a checkbox, the target name 'saw116-sys4Node01', and the version 'ND 8.0.0.0'. At the bottom of the table, it says 'Total 1'.

Figure 3-41 Nodes which the job manager is able to administer

For the equivalent command-line steps for this process, see the Information Center at the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.nd.doc/info/ae/ae/tagt_adminagent_setup.html

This section describes the steps to register a deployment manager node with a job manager:

1. Log in to the deployment manager administrative console and click **System administration** → **Deployment manager**.
2. Under Additional Properties, click **Job managers**.

- Click the **Register with Job Manager** button (Figure 3-42).

The screenshot shows the 'Deployment manager > Job managers' interface. At the top, there is a brief description about job managers and their registration. Below this is a toolbar with a 'Preferences' icon and two buttons: 'Register with Job Manager' (highlighted with a blue border) and 'Unregister from a Job Manager'. Underneath the toolbar are several icons for file operations like copy, move, and delete. A search bar with dropdown menus for 'Select' (set to 'UUID') and 'URL' is present. Below the search bar, a list box displays 'None' and 'Total 0'.

Figure 3-42 Registering a deployment manager with a job manager

- Enter the information required to connect to the job manager, including the host name, administrative host port, and user ID and password.

If the node name you are registering is already in use by the job manager, you can enter an alias for the node.

Click **OK** (Figure 3-43).

The screenshot shows the 'Deployment manager > Job managers > Register with Job Manager' dialog. It contains a descriptive text about registering managed nodes with a job manager. Below this is a 'General Properties' section with the following fields:

- Managed node name:** was8CellManager01 (highlighted with a yellow background)
- Alias:** (empty input field)
- Host name:** saw116-sys4
- Port:** 9943
- User name:** wasadmin
- Password:** (redacted)
- Confirm password:** (redacted)

At the bottom are three buttons: 'OK' (highlighted with a blue border), 'Reset', and 'Cancel'.

Figure 3-43 Job manager details

This action registers the deployment manager with the job manager.

- To view the newly registered deployment manager, log in to the job manager console and click **Jobs** → **Targets**. This action lists the nodes and deployment managers that are registered with the job manager.

3.4 Managing profiles

Each profile you create is registered in a profile registry:

install_root/properties/profileRegistry.xml

You have already seen how profiles are created with the Profile Management Tool. At the heart of this wizard is the **manageprofiles** command. This command enables you to maintain activities for profiles. For example, you can call this command to create profiles natively or silently, list profiles, delete profiles, validate the profile registry, and other functions.

3.4.1 Using the **manageprofiles** command

The **manageprofiles** command can be found in the *install_root/bin* directory. The syntax is **manageprofiles(.sh) -mode -arguments**. The modes listed in Table 3-3 are available.

Table 3-3 *manageprofiles* modes

Mode	Use
-create	Creates a new profile.
-delete	Deletes a profile.
-augment	Augments the given profile using the given profile template.
-unaugment	Unaugments the profile.
-unaugmentAll	Unaugments all the profiles.
-deleteAll	Deletes all registered profiles.
-listProfiles	Lists the profiles in the profile registry.
-listAugments	Lists the registered augments on a profile that is in the profile registry.
-getName	Returns the name of the profile at the path specified.
-getPath	Returns the path of the profile name specified.
-validateRegistry	Validates the profile registry and returns a list of profiles that are not valid.
-validateAndUpdateRegistry	Validates the profile registry and lists the non-valid profiles that it purges.
-getDefaultName	Returns the name of the default profile.
-setDefaultName	Sets the default profile.
-backupProfile	Back ups the given profile into a compressed file.
-restoreProfile	Restores the given profile from a compressed file.
-response	Manage profiles from a response file.
-help	Shows help.

3.4.2 Getting help

Run **manageprofiles -mode -help** for detailed help about each mode. Example 3-8 shows the help information for the **-create** mode.

Example 3-8 Getting help for the manageprofiles command

```
C:\WebSphere\AppServer\bin>manageprofiles -create -help
```

Function:

Creates a new profile

Syntax:

```
manageprofiles -create -<argument> <argument parameter> ...
```

Arguments:

The following command line arguments are required for this mode:

-templatePath <argument parameter>: The fully qualified pathname of the profile template that is

located on the file system.

-profileName <argument parameter>: The name of the profile.

-profilePath <argument parameter>: The intended location of the profile in the file system.

The following command line arguments are optional, and have no default values:

-isDefault <argument parameter>: Make this profile the default target of commands that do not use

their profile parameter.

-omitAction <argument parameter>: Omit optional features.

Note: Command-line arguments are case sensitive.

Note: If argument accepts a parameter containing spaces, the parameter must be enclosed in "double quotes".

Note: The default profile template is "default" and may be overridden by the **-templatePath** switch.

Note: Each profile template will have its own set of required and optional arguments.

3.4.3 Getting a list of profiles

Run **manageprofiles -listProfiles** to see a list of the profiles in the registry. Example 3-9 shows a sample output of **-listProfiles**.

Example 3-9 Listing profiles

```
C:\WebSphere\AppServer\bin>manageprofiles -listProfiles
```

```
[AppSrv01, Dmgr01, Custom01, AppSrv02, AdminAgent01, JobMgr01, AppSrv03]
```

Depending on the operation used, there may be other parameters that are required. These parameters are documented in the Information Center at the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.migration.base.doc/info/aes/ae/rxml_manageprofiles.html

3.4.4 Creating a profile with the manageprofiles command

You can use the `manageprofiles` command to create profiles.

Profile templates

Profiles are created based on templates supplied with the product. These templates are located in `install_root/profileTemplates`. Each template consists of a set of files that provide the initial settings for the profile and a list of actions to perform after the profile is created. When you create a profile using `manageprofiles`, you need to specify one of the following templates:

- ▶ Default (for application server profiles)
- ▶ Management (for deployment manager, job manager, and administrative agent profiles)
- ▶ Managed (for custom profiles)
- ▶ Cell (for cell profiles)

For example, the command used to create a deployment manager with node name `TestDmgr01` under profile name `TestDmgr01` is shown in Example 3-10.

Example 3-10 Creating a profile with the manageprofiles command

```
manageprofiles.bat -create -templatePath  
c:\WebSphere\AppServer\profileTemplates\management -serverType DEPLOYMENT_MANAGER  
-profileName TestDmgr01 -profilePath  
c:\WebSphere\AppServer\profiles\TestDmgr01 -enableAdminSecurity true  
-adminUserName wasadmin -adminPassword wasadmin11 -cellName TestCell01 -nodeName  
TestDmgr01
```

Log files that result when you run the `manageprofiles` command are located in:

`install_root/logs/manageprofile/profilename_action.log`

For example:

`C:/WebSphere/AppServer/logs/manageprofiles/TestDmgr01_create.log`

Additional log files are created in:

`install_root/logs/manageprofile/profile_name/`

For example:

`C:/WebSphere/AppServer/logs/manageprofiles/TestDmgr01`

Important: Do not manually modify the files that are located in the `install_root/profileTemplates` directory.

Options for specifying ports

During profile creation using the `manageprofiles` command, you can accept the default port values, or you can specify your port settings. If you want to specify ports, you can do so in any of the following ways:

- ▶ Specify the use of a port file that contains the port values.
- ▶ Specify the use of a starting port value.
- ▶ Specify the use of the default port values.

During profile creation, the **manageprofiles** command uses an automatically generated set of recommended ports. You can modify the port values using the following parameters on the **manageprofiles** command:

- ▶ **-defaultPorts**: Assigns default or base port values to the profile.

Note: Do not use the **-defaultPorts** parameter if you are using the managed profile template or using the **-startingPort** or **-portsFile** parameters.

- ▶ **-startingPort**: Specifies the starting port number for generating and assigning all ports for the profile.
- ▶ **-portsFile**: Specifies a path to a file that defines port settings for the profile.

You can use the **-validatePorts** parameter, which specifies the ports that should be validated to ensure they are not reserved or in use. This parameter helps you identify ports that are not being used.

Port file that contains the port values

You can supply a file containing the port values that you want to use for any profile by using the **-portsFile** option. See Example 3-11 for an example ports file named *portdef.props*.

Example 3-11 Example contents of portdef.props file

```
WC_defaulthost=39080 WC_adminhost=39060 WC_defaulthost_secure=39443
WC_adminhost_secure=39043 BOOTSTRAP_ADDRESS=32809 SOAP_CONNECTOR_ADDRESS=38880
IPC_CONNECTOR_ADDRESS=39633 SAS_SSL_SERVERAUTH_LISTENER_ADDRESS=39401
CSIV2_SSL_SERVERAUTH_LISTENER_ADDRESS=39403
CSIV2_SSL_MUTUALAUTH_LISTENER_ADDRESS=39402 ORB_LISTENER_ADDRESS=39100
DCS_UNICAST_ADDRESS=39353 SIB_ENDPOINT_ADDRESS=37276
SIB_ENDPOINT_SECURE_ADDRESS=37286 SIB_MQ_ENDPOINT_ADDRESS=35558
SIB_MQ_ENDPOINT_SECURE_ADDRESS=35578 SIP_DEFAULTHOST=35060
SIP_DEFAULTHOST_SECURE=35061
```

Incrementing port numbers from a starting point

The **manageprofiles** command can assign port numbers based on a starting port value. You can provide the starting port value from the command line using the **-startingPort** parameter. The command assigns port numbers sequentially from the starting port number value. However, if a port value in the sequence conflicts with an existing port assignment, the next available port value is used.

The order of port assignments is arbitrary. Predicting assignments is not possible.

Use the **-startingPort** option for the **manageprofiles** command. For example, the command used to create an application server profile with the node name *profile01Node* under profile name *profile01* and creating ports staring with 21000 is shown in Example 3-12.

Example 3-12 Creating a profile with the manageprofiles command

```
C:\WebSphere\AppServer\bin>manageprofiles.bat -create -templatePath
c:\WebSphere\AppServer\profileTemplates\default -profileName profile01
-profilePath
c:\WebSphere\AppServer\profiles\profile01 -startingPort 21000 -enableAdminSecurity
true -adminUserName wasadmin -adminPassword wasadmin11 -cellName TestCell01
-nodeName profile01Node
```

Default ports

Assigns the default or base port values to the profile. Use the **-defaultPorts** option with the **manageprofile** command.

Changing port settings after profile creation

Use the **updatePorts** tool to change port settings. For more information, read the article “Updating ports in an existing profile” at the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.nd.doc/info/ae/ae/tins_updatePorts.html

3.4.5 Deleting profiles

To delete a profile, complete the following steps (according to your situation):

- ▶ If you are removing a custom profile or application server profile that has been federated to a cell:
 - a. Stop the application servers on the node.
 - b. Remove the node from the cell using the administrative console or the **removeNode** command. Removing a node does not delete it, but restores it to its pre-federated configuration that was saved as part of the federation process.
 - c. Delete the profile using **manageprofiles -delete -profileName profile_name**.
 - d. Use the **manageprofiles -validateAndUpdateRegistry** command to clean the profile registry.
 - e. Delete the *profile_root* directory.
- ▶ If you are removing an application server profile that has not been federated to a cell:
 - a. Stop the application server.
 - b. Delete the profile using **manageprofiles -delete -profileName profile_name**.
 - c. Use the **manageprofiles -validateAndUpdateRegistry** command to clean the profile registry.
 - d. Delete the *profile_root* directory.
- ▶ If you are removing a deployment manager profile:
 - a. Remove any nodes federated to the cell using the administrative console or the **removeNode** command. Removing a node does not delete it, but restores it to its pre-federated configuration that was saved as part of the federation process.
 - b. Stop the deployment manager.
 - c. Delete the profile using **manageprofiles -delete -profileName profile_name**.
 - d. Use the **manageprofiles -validateAndUpdateRegistry** command to clean the profile registry.
 - e. Delete the *profile_root* directory.

If you have errors while deleting the profile, check the following log:

install_root/logs/manageprofile/profilename_delete.log

For example, in Example 3-13, you can see the use of the `manageprofiles` command to delete the profile named Node06.

Example 3-13 Deleting a profile using manageprofiles

```
C:\WebSphere\ND\profiles\dmgr01\bin>manageprofiles -delete -profileName Node06
INSTCONFSUCCESS: Success: The profile no longer exists.
```

As you can see, all seems to have gone well. But, as an additional step to ensure the registry was properly updated, you can list the profiles to ensure that the profile is gone from the registry, and validate the registry. See Example 3-14.

Example 3-14 Verifying the delete profile results

```
C:\WebSphere\ND\profiles\dmgr01\bin>manageprofiles -listProfiles
[Dmgr01, AppSrv01, AppSrv02, SamplesServer, WebServer2Node, DmgrSecure]

C:\WebSphere\ND\profiles\dmgr01\bin> manageprofiles -validateAndUpdateRegistry
[]
```

Note: If there are problems during the delete, you can manually delete the profile. For information about this, see the topic *Deleting a profile* in the Information Center at the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.installation.base.doc/info/aes/ae/tpro_removeprofile.html

See Chapter 27, “System recovery” on page 995 for information about some of the most common system management tasks dealing with configuration files, such as:

- ▶ Backing up a profile
- ▶ Restoring a profile
- ▶ Exporting and importing profiles



Installing WebSphere Application Server on z/OS systems

In this chapter, we explain how to install WebSphere Application Server on z/OS systems and provide an overview of IBM Installation Manager.

This chapter includes the following topics:

- ▶ IBM Installation Manager overview
- ▶ IBM Installation Manager installation
- ▶ Working with Installation Manager
- ▶ Using Installation Manager
- ▶ Installing WebSphere Application Server
- ▶ WebSphere Customization Toolbox

4.1 IBM Installation Manager overview

IBM Installation Manager is the Eclipse-based tool that allows you to manage all the aspects of installing WebSphere software, such as installation, update, rollback, and so on. It was originally introduced to support installation of Rational products, but is available for all platforms and provides the following benefits:

- ▶ Consistency across all platforms using the same methodology.
- ▶ Life cycle management of any Installation Manager product.
- ▶ The same “look and feel”, and it will run as command-line UNIX System Services application.
- ▶ Common packaging.
- ▶ More efficiency for delivering new fixes. A ++APAR will not be needed to install an interim fix.

We use the following concepts and terminology in this chapter:

- ▶ *Package*: A software product that can be installed by Installation Manager. Each package has a name, a version, and an identifier:
 - Package name: com.ibm.websphere.zOS.v80
 - Package version: 8.0.0.20110503_0249
 - Package identifier: com.ibm.websphere.zOS.v80_8.0.0.20110503_02
- ▶ The packages are installed at a defined location in a UNIX System Services file system. Installation Manager allows you to control where products will be installed and at which level.
- ▶ *Package group*: When more than one product is installed at the same location, the package group names are set automatically by Installation Manager.
- ▶ *Repository*: A place where the packages to be installed can be found. There is metadata in the repository that describes the software version and how it should be installed. It looks like a list of files organized in a tree structure, and it can reside on a local directory or on a reachable server.

Installation Manager maintains the software, and installs, uninstalls, or modifies it using a software repository. It uses three software locations:

- ▶ *Binary location*: Where Installation Manager is installed.
- ▶ *Agent data location*: The directory where Installation Manager stores data associated to an application, which includes state and operations history. It is also known as appDataLocation.
- ▶ *Object Cache location*: Used by Installation Manager to reduce the time when an operation is performed, which avoids spending time going online to use objects.

Figure 4-1 shows the Installation Manager components.

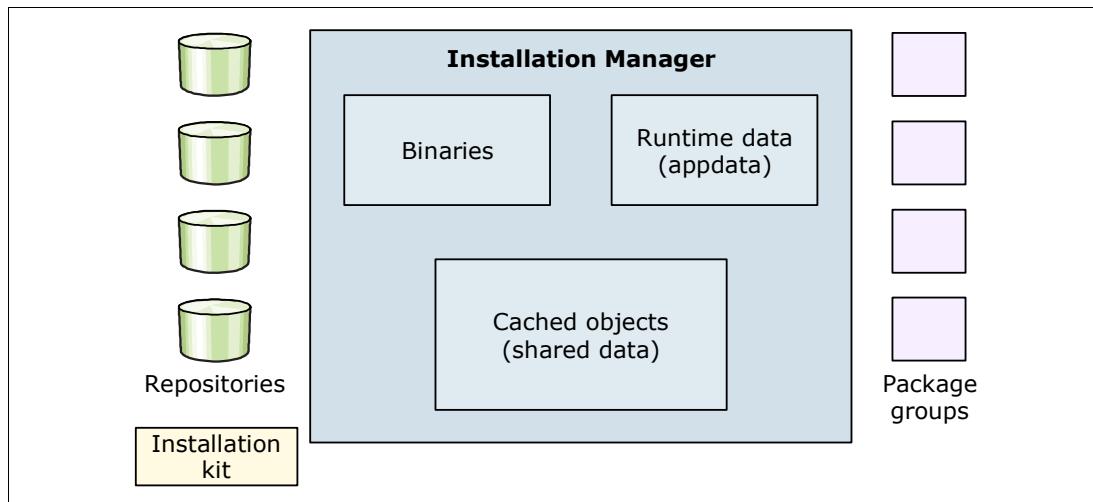


Figure 4-1 Installation Manager structure

4.2 IBM Installation Manager installation

You need to run Installation Manager only on those systems on which you install or update product code. You normally need only one Installation Manager on a system because one Installation Manager can track any number of product installations.

You install Installation Manager through SMP/E, and then use z/OS Profile Management Tool (zPMT) to configure your environment, as shown in Figure 4-2.

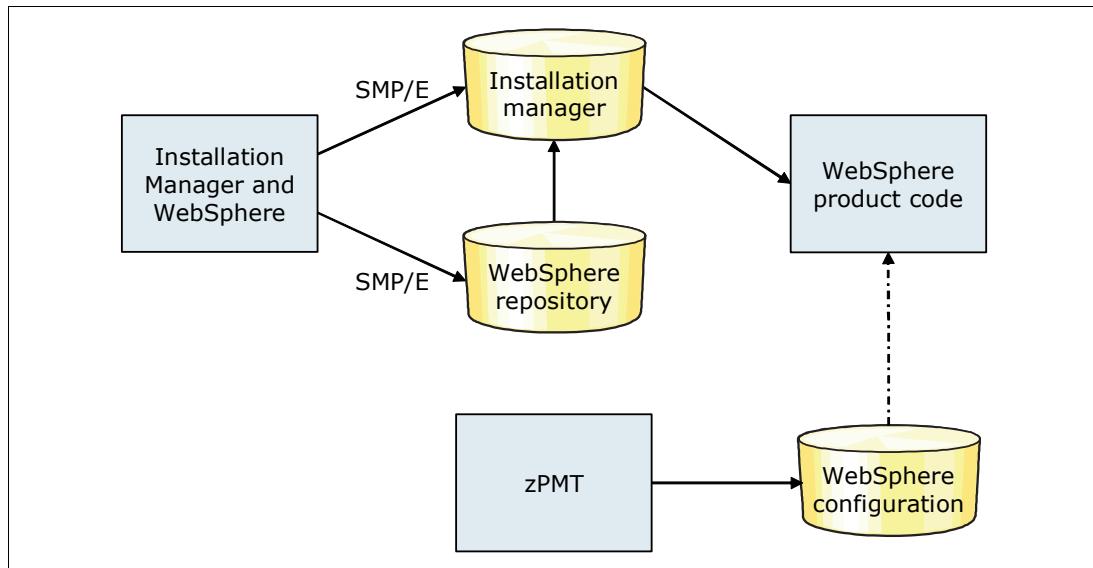


Figure 4-2 New installation process on z/OS

4.2.1 Checking prerequisites

Before you install WebSphere Application Server V8 for z/OS, be sure to check the hardware and software prerequisites at the following website:

<http://www-01.ibm.com/support/docview.wss?uid=swg27021536>

4.2.2 Obtaining an Installation Manager installation kit

Installation Manager has function modification identifier (FMID) HGIN140. You can obtain the installation kit in one of the following ways:

- ▶ Through ServerPac or IBM SystemPac® in a ready to use format
- ▶ Custom-Built Product Delivery Offering (CBPDO)

In this scenario, you must use the following jobs to install CB PDO:

- GINRECEV to receive the deliverables
 - GINALLOC to allocate target and distribution data sets
 - GINDEF to create SMP/E DDDEFs
 - GINISMKD to mount the installation file system and create directories
 - GINAPPLY to apply the installation kit FMID
 - GINACCEPT to accept the installation kit FMID
- ▶ Download the kit from the Internet, transfer the compressed file to the z/OS system, and extract it.

Notice that if you download Installation Manager from the Internet that SMP/E cannot maintain and track its level. WebSphere Application Server V8 ships with Installation Manager V1.4.3, but Installation Manager should be at the latest level. You can order Installation Manager through Shopz or find the most current version at the following website:

http://publib.boulder.ibm.com/infocenter/install/v1r4/index.jsp?topic=/com.ibm.cic.agent.ui.doc/topics/r_links.html

4.2.3 Installing Installation Manager on the system

Before you create Installation Manager, you must install APAR OA34228 on each system that will host Installation Manager.

After you install this APAR, you need to define the mode in which Installation Manager will run on the system:

- ▶ Admin mode: Installation Manager can be invoked by any superuser, and there can be only one admin mode Installation Manager on a system.
- ▶ User mode: Installation Manager can be invoked by a non-superuser, and there can be only one user mode Installation Manager per user.
- ▶ Group mode: Installation Manager can be invoked by any user who is connected to the group that owns Installation Manager, and there is no limit for the number of group mode installations.

Installation Manager includes the following sets of files:

- ▶ Binaries: A set of executable files.
- ▶ Runtime: A set of files that describe the installed products.

- ▶ Shared data: A set of files that store artifacts from a repository when installing packages. To install WebSphere Application Server, the shared data should have at least 30,000 tracks of free space. For tips about using shared data, see “Shared data” on page 146.

Important: All sets of files must be writable by an Installation Manager user or group. The permissions for admin and user mode are 755 and for group mode are 775.

The user who will run Installation Manager must have the following permissions on IBM Resource Access Control Facility (RACF®):

- ▶ Read access to FACILITY profile BPX.FILEATTR.APF
- ▶ Read access to FACILITY profile BPX.FILEATTR.PROGCTL
- ▶ Read access to FACILITY profile BPX.FILEATTR.SHARELIB
- ▶ Read access to UNIXPRIV profile SUPERUSER.FILESYS.CHOWN
- ▶ Read access to UNIXPRIV profile SUPERUSER.FILESYS.CHANGEPERMS

Table 4-1 lists the file systems that are needed to hold Installation Manager information.

Table 4-1 Installation Manager default locations

Information type	Default location
Binaries	/InstallationManager/bin
Application data	/InstallationManager/appdata

Complete the following steps to install Installation Manager:

1. Go to the directory where Installation Manager is installed and run the following command:
`./set-ext-attr.sh`
2. Choose a user ID to be the owner of this Installation Manager. If you need to create a user ID, use the GIN2ADMN job.
3. Create and mount two file systems to configure Installation Manager. You can use the GIN2CFS sample job.
4. To install an Installation Manager issue one of the following commands:

installc	Install Installation Manager in admin mode.
userinstc	Install Installation Manager in user mode.
groupinstc	Install Installation Manager in group mode.

Important: The user who runs the command will be the Installation Manager owner.

Example 4-1 demonstrates how to install Installation Manager in admin mode. Use the following parameters:

-acceptLicense	Accepts the software license agreement.
-installationDirectory	Specifies the directory where Installation Manager binaries are installed.
-dataLocation	Specifies the directory where runtime data is installed.

Example 4-1 Install Installation Manager in admin mode

/usr/lpp/InstallationManager/V1R4/installc	-acceptLicense
-installationDirectory	/InstallationManager/bin -dataLocation
/InstallationManager/appdata	

You can use the GIN2INST sample job to install the Installation Manager.

Verification: To verify that Installation Manager was installed, run the following command:
`/InstallationManager/bin/eclipse/tools/imcl -version`

Shared data

The data stored in the shared data directory is used only for a package rollback if a product repository is not available. The amount of shared data can become large if you have several products installed. You can delete the shared data contents just after using Installation Manager if you have access to the product repositories. As an alternative, you can ask Installation Manager to discard the cache objects by adding the preference value shown in Example 4-2 to the `imcl install` command.

Example 4-2 The preferences parameter to discard cached objects

```
-preferences com.ibm.cic.common.core.preferences.preserveDownloadedArtifacts=false
```

This value is used in the sample jobs in library SBBOJCL.

Note: The shared resources directory is set at the first time a package is installed. If all cached objects are removed and the directory becomes empty, Installation Manager can unset the shared resources location.

4.3 Working with Installation Manager

When you work with Installation Manager, you can influence the way that Installation Manager works by setting its preferences. You also need one or more repositories. In this section, we explain how to work with Installation Manager.

4.3.1 Installation Manager preferences

You can set the Installation Manager preferences using one of the following methods:

- ▶ When using command line, specify the `-preferences` parameter followed by the parameter name and its value, as shown in Example 4-2.
- ▶ When using a response file, specify the preference in XML format, as shown in Example 4-3.

Example 4-3 Setting a preference in a response file

```
<preference name='com.ibm.cic.common.core.preferences.eclipseCache'  
value='/InstallationManager/sharedResources' />
```

4.3.2 Repository overview

The Installation Manager uses a repository to identify the packages or updates to install. A repository is a location that stores data for installing, modifying, rolling back, updating, or uninstalling packages. Each installed package has an embedded location for its default update repository. You can add, edit, or remove repositories to be used by the Installation Manager.

Based on the configured repositories, the Installation Manager determines and lists all the available packages to install, including products, fix packs, interim fixes, and so on. It checks prerequisites and interdependencies, and installs the selected packages.

The Installation Manager repository contains one or multiple product offerings that have both metadata and actual payload for the offerings. The offering metadata describes the following aspects of the offering:

- ▶ Name, version, supported platforms, and so on
- ▶ Required and optional features
- ▶ Relationships and dependency between offerings and features of offerings

Normally, an Installation Manager repository contains the full content that is required to install on various platforms, operating systems, and so on.

Repository topologies can be generalized into the following categories:

1. A public repository that is accessible to the general public at an IBM-hosted site, such as IBM Passport Advantage
2. A local repository that is used by a single user and not shared with others
3. An enterprise repository that is located behind the firewall and is accessed by multiple machines within the enterprise

4.3.3 Updating Installation Manager

To update Installation Manager, you need to update the installation kit. You can perform the update as described in 4.2.2, “Obtaining an Installation Manager installation kit” on page 144.

You need to refresh the Installation Manager installation using the same commands that you used to create the Installation Manager (`installc`, `userinstc`, or `groupinstc`). For details about these commands, refer to 4.2.3, “Installing Installation Manager on the system” on page 144.

4.3.4 Installing the WebSphere Application Server initial repository

WebSphere Application Server V8 has FMID HBBO800 and can be obtained in one of the following ways:

- ▶ ServerPac or SystemPac: Follow the instructions in the *Installing Your Order* guide to create repository file system <high level qualifier (HLQ)>.SBB0IMR and <HLQ>.SBB0JCL library.
- ▶ Custom-Built Product Delivery Offering (CB PDO): You must run the jobs to install WebSphere Application Server repository following the instructions that are available at Program Directory to create a repository file system <high level qualifier (HLQ)>.SBB0IMR and <HLQ>.SBB0JCL library.

Run the following jobs to create the WebSphere Application Server repository:

- ▶ BBORECEV to receive the deliverables
- ▶ BBOISMKD to allocate the system paths
- ▶ BBODDDEF to define de SMP/E DDDEFs
- ▶ BBOAPPLY to apply product repository
- ▶ BBOACCEP to accept product repository

4.4 Using Installation Manager

Before you begin using Installation Manager, consider in which mode it is going to be used and what actions are going to be performed in the software that is going to be maintained. The following options are available:

- ▶ Command-line mode: Use the Installation Manager command line (`imcl`) to manage installations from the `/eclipse/tools` subdirectory.
- ▶ Silent mode: Use this mode to install software easily to multiple systems. Silent mode uses the `imcl` command in conjunction with response files

You can use all the commands that we demonstrate in this chapter in z/OS jobs using BPXBATCH.

4.4.1 Key features of Installation Manager

Installation Manager is used to perform several maintenance operations, such as update, modify, or roll back packages. These operations differ slightly from how they are performed on a distributed platform.

Installing packages

A package can be installed by Installation Manager. When the specific version of a package is not specified in the installation command, Installation Manager checks the repository that is used and picks the highest level that is available in the repository. For example, the WebSphere package name for z/OS is `com.ibm.websphere.zOS.v80`. Thus, if there is more than one WebSphere level at a repository and the installation should be done at a level that is not the highest, you need to specify the complete package name, for example, `com.ibm.websphere.zOS.v80_8.0.0.20110503_0249`.

Updating packages

You can update a package as soon as the updates are available in an Installation Manager repository. You can apply the following types of updates:

- ▶ A *fix pack* is a new product level. Each fix pack repository is a delta on top of the previous fix pack level. Fix packs have the same package name but a different version level. A fix pack can be delivered as a SMP/E program temporary fix (PTF) to the initial product repository.
- ▶ An *interim fix* is also known as a patch. Interim fixes uses the package name, and they are not available as SMP/E PTF.

You can also obtain fix packs and interim fixes by downloading them from the IBM Fix Central website:

<http://www.ibm.com/support/fixcentral/>

When you need to verify the fixes that are available for maintenance, use the **imcl listAvailableFixes** command, as shown in Example 4-4, with the following parameters:

<package name> The package to be installed with the version
-repositories The repository location

Example 4-4 Listing available fixes in a repository

```
/InstallationManager/bin/eclipse/tools $ imcl listAvailableFixes
com.ibm.websphere.zOS.v80_8.0.0.20110503_0249 -repositories
/usr/lpp/InstallationManagerRepository/HBB0800
```

Modifying packages

A package can have features, languages, and functions added or removed by Installation Manager. To modify a package, run **imcl install**. To add sample applications to WebSphere Application Server, as shown in Example 4-5, run **imcl install** with the following parameters:

<package name>, <feature name> The package and feature name to be installed
-installationDirectory The directory where the package is installed
-repositories The repository location

Example 4-5 Adding sample applications to WebSphere Application Server

```
/InstallationManager/bin/eclipse/tools $ imcl install
com.ibm.websphere.zOS.v80,samples -installationDirectory /opt/zWebSphere/V8R0
-repositories /usr/lpp/InstallationManagerRepository/HBB0800
```

To uninstall the sample applications from WebSphere Application Server, as shown in Example 4-6, run **imcl install** with the following parameters:

<package name> The package name to be installed
-installationDirectory The directory where the package is installed

Example 4-6 Adding a language pack to WebSphere Application Server

```
/InstallationManager/bin/eclipse/tools $ imcl uninstall
com.ibm.websphere.zOS.v80,samples -installationDirectory /opt/zWebSphere/V8R0
```

To add a language pack to WebSphere Application Server, as shown in Example 4-7, run **imcl install** with the following parameters:

<package name> The package name to be installed
-properties cic.selector.nl The language pack to be installed
-installationDirectory The directory where the package is installed
-repositories The repository location

Example 4-7 Adding a language pack to WebSphere Application Server

```
/InstallationManager/bin/eclipse/tools $ imcl install com.ibm.websphere.zOS.v80
-installationDirectory /opt/zWebSphere/V8R0 -properties cic.selector.nl=de
-repositories /usr/lpp/InstallationManagerRepository/HBB0800 -acceptLicense
```

Rolling back packages

To roll back a package, run **imcl install** and specify the previous product level. For example, if you are running WebSphere at Fix Pack 14 and want to roll it back to Fix Pack 13, issue a installation command that specifies Fix Pack 13.

If there were interim fixes installed at the previous level, you need to reinstall these interim fixes. You can use only one command or you can just install the interim fixes after rolling back the fix pack.

Caution: Take care when interim fixes are involved in building the product's previous level. If interim fixes are available only online, you need to specify all the repositories (local and online) in the installation command and separate their names by commas without blank spaces in between them.

Creating a keyring

When you access certain URLs, for example, to download fixes from IBM Fix Central, credentials are required for authentication. To use these credentials, you need a keyring file. Because the **imcl** command does not have this capability, you need to use the **imutilsc** command. You can use this command, as shown in Example 4-8, with the following parameters:

saveCredential	Instructs imutilsc to save credentials
-url	The URL that is accessed
-userName	The user name that authenticates to the URL
-userPassword	The password for the user
-keyring	The file where information is stored

Example 4-8 Creating a keyring

```
/InstallationManager/bin/eclipse/tools $ imutilsc saveCredential -url  
http://www.mycorporation.com/repository -userName jsmith -userPassword secret  
-keyring /u/jsmith/corporate.keyring
```

Listing installed products

You can list information about the installed products by running **imcl listInstalledPackages**, as shown in Example 4-9.

Example 4-9 Listing the installed packages

```
/InstallationManager/bin/eclipse/tools $ imcl listInstalledPackages  
com.ibm.cic.agent_1.4.3000.20110303_1846  
com.ibm.websphere.zOS.v80_8.0.0.20110503_0249  
com.ibm.websphere.PLG.zOS.v80_8.0.0.20110503_0306
```

At certain times, you need to check the features that are installed with a package. Use the **imcl listInstalledPackages -feature** command, as shown in Example 4-10.

Example 4-10 Listing the features installed by packages

```
/InstallationManager/bin/eclipse/tools$ imcl listInstalledPackages -features  
com.ibm.cic.agent_1.4.3000.20110303_1846 :  
com.ibm.websphere.PLG.zOS.v80_8.0.0.20110503_0306 :  
com.ibm.websphere.zOS.v80_8.0.0.20110503_0249 :  
ejbdeploy,embeddablecontainer,thinclient
```

You can use the **versionInfo.sh** command to show the same information as the **-feature** option.

4.4.2 Uninstalling Installation Manager

Before you uninstall Installation Manager, you must uninstall all the packages that were previously installed by Installation Manager. To uninstall Installation Manager, complete the following steps:

1. Log in as the Installation Manager owner's user ID.
2. Uninstall all software packages.
3. To uninstall Installation Manager, run **uninstallc**, which can be found in the agent data directory under the **uninstall** subdirectory, as shown in Example 4-11.

Example 4-11 Uninstall Installation Manager

```
/InstallationManager/appdata/uninstall $ ./uninstallc
```

You can use the GIN2UNIN sample job to uninstall Installation Manager.

4.5 Installing WebSphere Application Server

You can install WebSphere Application Server V8 by using the command line or the supplied jobs. We describe how to install WebSphere Application Server using the command line.

4.5.1 Installing using the command line

Complete the following steps to install WebSphere Application Server using the command line:

1. An empty file system is needed to hold the WebSphere Application Server installation. You can create an empty file system manually or you can use the **zCreateFileSystem.sh** script. This file system needs at least 35,000 tracks (3390) or 1,800 MB. Example 4-12 shows a sample execution where the following parameters were used:

-name	Data set name
-type	Type of file system
-megabytes	Primary and secondary allocation
-volume	Volume where the data set will reside
-mountpoint	Directory where the file system will be mounted
-owner	Directory owner
-group	Group owner

Example 4-12 Creating an empty file system

```
/InstallationManager/bin/eclipse/tools $ zCreateFileSystem.sh -name
OMVS.WAS800.SBBOHFS -type ZFS -megabytes 1800 200 -volume TARHF1 -mountpoint
/opt/zWebSphere/V8R0 -owner STC -group TS0
CWLC9023I Defining file system OMVS.WAS800.SBBOHFS.
IOEZ00248I VSAM linear dataset OMVS.WAS800.SBBOHFS successfully created.
CWLC9024I File system OMVS.WAS800.SBBOHFS successfully defined.
CWLC9022I Formatting ZFS file system OMVS.WAS800.SBBOHFS.
IOEZ00077I HFS-compatibility aggregate OMVS.WAS800.SBBOHFS has been
successfully created
CWLC9012I Creating mount point directory /opt/zWebSphere/V8R0.
```

```
CWLCS9013I Mount point directory /opt/zWebSphere/V8R0 successfully created.  
CWLCS9006I Mounting data set OMVS.WAS800.SBBOHFS at mount point  
/opt/zWebSphere/V8R0.  
CWLCS9007I OMVS.WAS800.SBBOHFS successfully mounted at mount point  
/opt/zWebSphere/V8R0.  
CWLCS9017I Setting owner and group for directory /opt/zWebSphere/V8R0.  
CWLCS9018I Owner and group successfully set for directory /opt/zWebSphere/V8R0.  
CWLCS9019I Setting permissions for directory /opt/zWebSphere/V8R0.  
CWLCS9020I Permissions successfully set for directory /opt/zWebSphere/V8R0.
```

You can also use the BBO1CFS job instead of using the command line.

2. From the /InstallationManager/bin/eclipse/tools directory, run **imcl listAvailablePackages** to check if the repository has the needed packages, as shown in Example 4-13, using the **repositories** parameter to set the repository location.

Example 4-13 Listing available packages on a repository

```
/InstallationManager/bin/eclipse/tools $ imcl listAvailablePackages  
-repositories /usr/lpp/InstallationManagerRepository/HBB0800  
com.ibm.websphere.IHS.zOS.v80_8.0.0.20110503_0303  
com.ibm.websphere.NDDMZ.zOS.v80_8.0.0.20110503_0249  
com.ibm.websphere.PLG.zOS.v80_8.0.0.20110503_0306  
com.ibm.websphere.zOS.v80_8.0.0.20110503_0249
```

License note: You can find the license for a package in the `1afies` subdirectory of the Installation Manager repository. Be sure to check it before installing products.

3. To install WebSphere Application Server, run **imcl install**, as shown in Example 4-14, using the following parameters:

<package name>	The package to be installed
-installationDirectory	The directory where the package is installed
-repositories	The repository location
-sharedResourcesDirectory	The directory where the artifacts from repository are stored

Parameter note: This parameter can be omitted in subsequent commands after the shared resources directory has been set for the first time.

-acceptLicense To accept the software license agreement

Example 4-14 WebSphere Application Server installation

```
/InstallationManager/bin/eclipse/tools $ imcl install com.ibm.websphere.zOS.v80  
-installationDirectory /opt/zWebSphere/V8R0 -repositories  
/usr/lpp/InstallationManagerRepository/HBB0800 -sharedResourcesDirectory  
/InstallationManager/sharedResources -acceptLicense
```

You can also use the BBO1INST job instead of using the command line.

4. After installing WebSphere Application Server, mount and remount the product file system in read-only mode for use by WebSphere Application Server nodes and servers.

4.5.2 Installing additional packages

WebSphere Application Server includes the following additional packages:

- ▶ DMZ secure proxy
- ▶ IBM HTTP server
- ▶ Web server plug-ins

You can install these packages using the following jobs:

- ▶ BBO2CFS to allocate a file system for the DMZ secure proxy
- ▶ BBO2INST to install the DMZ secure proxy
- ▶ BBO3CFS to allocate a file system for the web server plug-ins
- ▶ BBO3INST to install the web server plug-ins
- ▶ BBO4CFS to allocate a file system for the IBM HTTP Server
- ▶ BBO4INST to install the IBM HTTP Server

4.5.3 Creating response files

You can create a response file that will contain all the installation commands needed to install software with Installation Manager. This method allows you to reuse the commands to perform the installation on several machines. Some products deliver a sample file to use during the installation process. If a sample file is not available, as with WebSphere Application Server, you can create a sample file using the samples that are available at the information center, or you can record a sample file during an Installation Manager operation.

Sample files

You can find the sample files at the following website:

http://publib.boulder.ibm.com/infocenter/install/v1r4/index.jsp?topic=/com.ibm.silentinstall12.doc/topics/c_sample_response_files.html

Choose one of the scenarios and adapt the sample file to your installation. The commands that you can use in response files are available at the following website:

http://publib.boulder.ibm.com/infocenter/install/v1r4/index.jsp?topic=/com.ibm.silentinstall12.doc/topics/r_silent_inst_commands12.html

Recording a response file

During an Installation Manager operation, a response file can be recorded, allowing you to reuse it later. Example 4-15 shows how to generate a response file during the web server plug-in package installation. It uses the following parameters:

<command>	Specifies the action that should be taken
<package name>	Specifies the package that is handled by the <command> action
-repositories	Specifies the repository to use
-record	Specifies the file name that will be generated
-acceptLicense	Specifies that you agree to the license agreement
-installationDirectory	Specifies in which directory the package is installed

Example 4-15 Record a response file when installing a package

```
/InstallationManager/bin/eclipse/tools $ imcl install  
com.ibm.websphere.PLG.zOS.v80 -repositories
```

```
/usr/lpp/InstallationManagerRepository/HBB0800 -record /tmp/plg_inst.xml  
-acceptLicense -installationDirectory /opt/zWebSphere_Plugins/V8R0
```

The resulting file has all the instructions needed to install the web server plug-in package, as shown in Example 4-16.

Example 4-16 Response file for a web server plug-in installation

```
<?xml version="1.0" encoding="UTF-8"?>  
<agent-input>  
<server>  
<repository location='/usr/lpp/InstallationManagerRepository/HBB0800' />  
</server>  
<profile id='Web Server Plug-ins for IBM WebSphere Application Server V8.0'  
installLocation='/opt/zWebSphere_Plugins/V8R0'>  
<data key='eclipseLocation' value='/opt/zWebSphere_Plugins/V8R0' />  
<data key='user.import.profile' value='false' />  
</profile>  
<install modify='false'>  
<offering id='com.ibm.websphere.PLG.zOS.v80' version='8.0.0.20110503_0306'  
profile='Web Server Plug-ins for IBM WebSphere Application Server V8.0'  
features='core.feature' installFixes='none' />  
</install>  
<preference name='com.ibm.cic.common.core.preferences.eclipseCache'  
value='/InstallationManager/sharedResources' />  
<preference name='com.ibm.cic.common.core.preferences.connectTimeout' value='30' />  
<preference name='com.ibm.cic.common.core.preferences.readTimeout' value='45' />  
<preference name='com.ibm.cic.common.core.preferences.downloadAutoRetryCount'  
value='0' />  
<preference name='offering.service.repositories.areUsed' value='true' />  
<preference name='com.ibm.cic.common.core.preferences.ssl.nonsecureMode'  
value='false' />  
<preference  
name='com.ibm.cic.common.core.preferences.http.disablePreemptiveAuthentication'  
value='false' />  
<preference name='http.ntlm.auth.kind' value='NTLM' />  
<preference name='http.ntlm.auth.enableIntegrated.win32' value='true' />  
<preference name='com.ibm.cic.common.core.preferences.preserveDownloadedArtifacts'  
value='true' />  
<preference name='com.ibm.cic.common.core.preferences.keepFetchedFiles'  
value='false' />  
<preference name='PassportAdvantageEnabled' value='false' />  
<preference name='com.ibm.cic.common.core.preferences.searchForUpdates'  
value='false' />  
<preference name='com.ibm.cic.agent.ui.displayInternalVersion' value='false' />  
</agent-input>
```

4.5.4 Installing silently

You can use silent installation to perform software deployment to multiple systems by completing the following steps:

1. Install Installation Manager.
2. Generate a response file.
3. Manage your packages silently.

After Installation Manager is installed and a response file is recorded, you can use the response file to make new installations, as shown in Example 4-17. Run `imcl` with the following parameters:

<code>input</code>	Specifies which response file should be used
<code>-log</code>	Specifies the log file to be generated
<code>-acceptLicense</code>	Specifies that you agree to the license agreement

Example 4-17 Package installation in silent mode

```
/InstallationManager/bin/eclipse/tools $ imcl input /tmp/teste_plg.xml -log  
/tmp/silent.log -acceptLicense
```

4.5.5 The post-installer

For z/OS systems, sometimes WebSphere Application Server requires that service applied changes are made to configuration files. The post-installer can update the configuration files automatically or manually. You can use the post-installer to complete the following tasks:

- ▶ Run configuration actions by Installation Manager at installation or uninstallation time.
- ▶ Automatically detect the application or removal of fix packs, and run any necessary configuration actions.
- ▶ Automatically detect the addition or removal of an offering, optional feature, or interim fix, and then create or remove any associated symbolic links.

Installation Manager considers the post-installation step as a nonfatal step. Thus, if the post-installer returns FAIL or PARTIAL SUCCESS return code, Installation Manager will displays the following message:

The packages are installed with warnings

If you receive this message, consult the information found at this website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=%2Fcom.ibm.websphere.installation.zseries.doc%2Finfo%2Fzseries%2Fae%2Ftins_postinstallapp.html

4.5.6 Service information

Until WebSphere Application Server V7, software installers looked for actions that should be performed when installing software maintenance on SMP/E using the ACTION HOLD information provided in sysouts. For WebSphere Application Server V8, SMP/E is not used to install maintenance. To check the ACTION HOLD information, you need to check the preventive service planning (PSP) bucket, which is a source of service planning for IBM mainframe products. PSP buckets are available at the following website:

<http://www14.software.ibm.com/webapp/set2/psearch/search?domain=psp>

To find the proper information, you need to complete the upgrade name and subset name with the values for WebSphere Application Server and Installation Manager listed in Table 4-2.

Table 4-2 PSP information for WebSphere Application Server and Installation Manager

Product	Upgrade name	Subset name
WebSphere Application Server	WASAS800	HBBO800
Installation Manager	IIMZOSV1	HGIN140

4.5.7 Uninstalling packages

You can uninstall packages using the command line or using supplied jobs. We describe how to uninstall WebSphere Application Server using the command line.

Complete the following steps to install WebSphere Application Server:

1. From the /InstallationManager/bin/eclipse/tools directory, run **imcl listInstalledPackages** to verify that the package is installed, as shown in Example 4-18.

Example 4-18 Listing installed packages

```
/InstallationManager/bin/eclipse/tools $ imcl listInstalledPackages
com.ibm.cic.agent_1.4.3000.20110303_1846
com.ibm.websphere.zOS.v80_8.0.0.20110503_0249
com.ibm.websphere.PLG.zOS.v80_8.0.0.20110503_0306
```

2. To uninstall WebSphere Application Server, run **imcl uninstall**, as shown in Example 4-19, using the following parameters:

<package name>	The package to be uninstalled
-installationDirectory	The directory where the package is installed

Example 4-19 WebSphere Application Server uninstallation

```
/InstallationManager/bin/eclipse/tools $ imcl uninstall
com.ibm.websphere.zOS.v80 -installationDirectory /opt/zWebSphere/V8R0
```

You can also use the BBO1UNIN job instead of using the command line.

The uninstallation jobs for the other WebSphere components are:

- ▶ BBO2UNIN to uninstall the DMZ secure proxy
- ▶ BBO3UNIN to uninstall the web server plug-ins
- ▶ BBO4UNIN to uninstall the IBM HTTP Server

4.5.8 Preparing the base z/OS operating system

After installing WebSphere Application Server, you need to prepare the z/OS system. Because of extensive use of underlying z/OS services for security, reliability, and performance, consider the following resources:

- ▶ Sysplex
- ▶ Job entry subsystem (JES)
- ▶ Resource recovery services (RRS)
- ▶ Resource access control facility (RACF)
- ▶ TCP/IP

Be sure to check the WebSphere Application Server Information Center at the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=%2Fcom.ibm.websphere.installation.zseries.doc%2Finfo%2Fzseries%2Fae%2Ftins_preparez.html

4.6 WebSphere Customization Toolbox

The WebSphere Customization Toolbox for WebSphere Application Server V8.0 includes tools for managing, configuring, and migrating various parts of your WebSphere Application Server environment. It is an Eclipse framework application that existed in V7 that had a vastly smaller function for configuring only z/OS servers. The WebSphere Customization Toolbox is a container framework that holds various tools that are used for configuring your application server environment.

For details about the WebSphere Customization Toolbox and how to install it, refer to 2.6, “WebSphere Customization Toolbox” on page 70.



Working with profiles on z/OS systems

The information in this chapter can help you build your initial WebSphere Application Server environment using the new WebSphere Customization Toolbox.

This chapter includes the following topics:

- ▶ Creating WebSphere environments
- ▶ Getting started with the Profile Management tool
- ▶ Creating a sample z/OS Network Deployment cell
- ▶ Creating a deployment manager definition
- ▶ Creating the base application server definition
- ▶ Federating an application server
- ▶ Creating a job manager profile
- ▶ Creating an administrative agent profile

5.1 Creating WebSphere environments

Configuring a WebSphere Application Server for z/OS environment consists of setting up the configuration directory for the environment and making changes to the z/OS target system that pertain to the particular application serving environment. Configuring these application serving environments after product installation requires a fair amount of planning and coordination.

For example, when defining multiple deployment managers or application servers on a single machine or LPAR, you need to ensure that the ports and names you select for each one are unique and that the z/OS environment variables, generated jobs, and so on, are set up properly. Spend time planning the installation and, if possible, practice first by configuring a stand-alone application server using the default options.

You use the WebSphere Application Server for z/OS Profile Management tool available with the WebSphere Customization Toolbox to configure WebSphere Application Server environments for the z/OS platform.

The Profile Management tool is a dialog-driven tool that runs in the WebSphere Customization Toolbox. It is an Eclipse plug-in that allows you to perform the initial setup of WebSphere Application Server for z/OS cells and nodes. It provides the same functionality as the former ISPF dialog boxes with additional features.

The WebSphere Customization Toolbox itself does not create the cells and nodes; however, it creates batch jobs, scripts, and data files that you can use to perform WebSphere Application Server for z/OS customization tasks. These jobs, scripts, and data files form a *customization definition* on your workstation, which is then uploaded to z/OS where you submit the jobs, as shown in Figure 5-1.

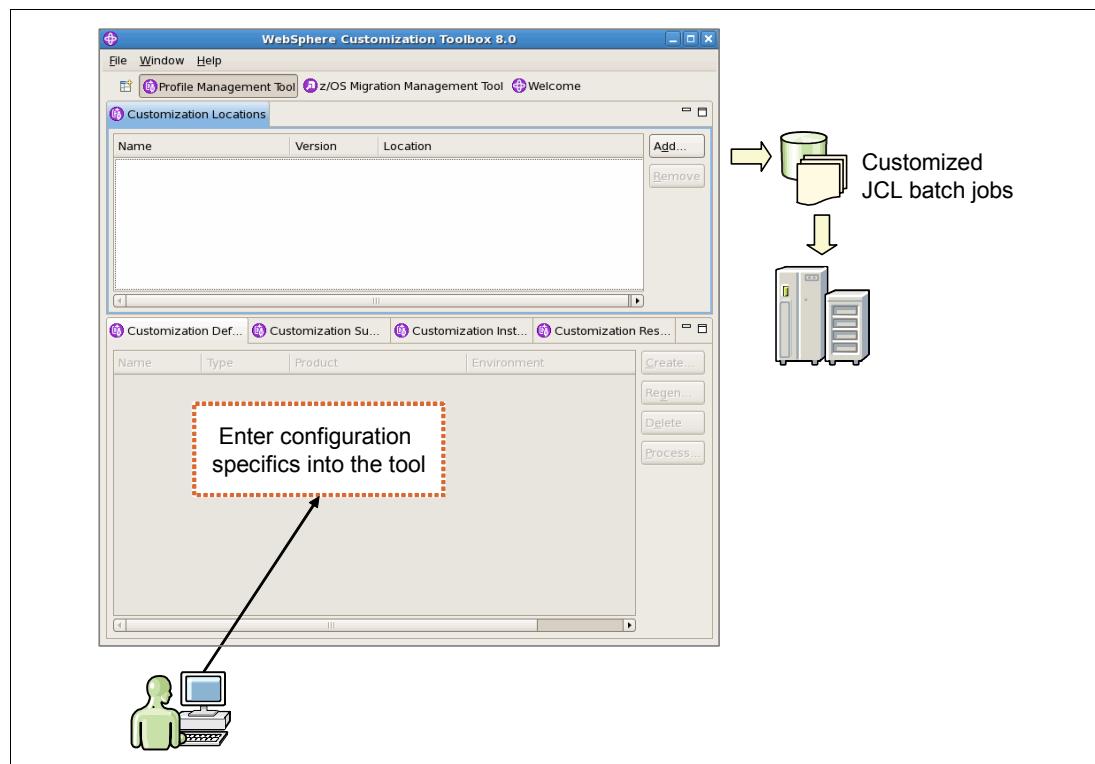


Figure 5-1 The WebSphere Customization Toolbox configuration flow

Review the documentation: The WebSphere Application Server Information Center contains planning topics for each WebSphere Application Server package that is tailored to each platform. This section gives you a high level look at the planning tasks that you need to perform.

If you are planning a WebSphere Application Server for z/OS environment, review the installation planning material for z/OS platforms that is available at the following website:

<http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.installation.zseries.doc/info/zseries/ae/welc6topinstalling.html>

You can use a spreadsheet to create and record configuration variables and plan many of the values that you need to specify to the Profile Management tool. You can download a template spreadsheet from the following website:

<http://www-03.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/PRS4686>

Using the Profile Management tool, you can create environments for the following products:

- ▶ WebSphere Application Server for z/OS
- ▶ WebSphere DMZ secure proxy server for z/OS

5.1.1 WebSphere Application Server for z/OS

The Profile Management tool provides support to generate jobs to create the following WebSphere Application Server for z/OS environments:

- ▶ Create a cell environment consisting of a deployment manager and a federated application server.
- ▶ Create a management environment, which can be either:
 - A deployment manager
 - A job manager
 - An administrative agent
- ▶ Create a stand-alone application server environment.
- ▶ Create a managed (custom) node and federate it into an existing cell.
- ▶ Federate an application server.

5.1.2 WebSphere DMZ secure proxy server for z/OS

The DMZ secure proxy server for an IBM WebSphere Application Server installation allows you to install your proxy server in the DMZ, while reducing the security risk that might occur if you choose to install an application server in the DMZ to host a proxy server. The risk is reduced by removing any functionality from the application server that is not required to host the proxy servers, but this reduction in security can also pose a security risk. Installing the secure proxy server in the DMZ rather than the secured zone presents new security challenges. However, the secure proxy server is equipped with capabilities to provide protection from these challenges.

The Profile Management tool for WebSphere Application Server V8.0 provides support for the following WebSphere DMZ secure proxy server for z/OS environments:

- ▶ Management
Generates the customization jobs to create an administrative agent for the secure proxy server.
- ▶ Secure proxy
Generates the customization jobs to create a secure proxy server.

5.2 Getting started with the Profile Management tool

This section explains how to prepare and start the Profile Management tool. These steps are common for any type of profile.

To start the Profile Management tool, complete the following steps:

1. Start the WebSphere Customization Toolbox.

On Windows systems, click **Start → All Programs → IBM WebSphere → WebSphere Customization Toolbox V8.0 → WebSphere Customization Toolbox**.

On Linux, click ***operating_system_menus_to_access_programs → IBM WebSphere → WebSphere Customization Toolbox V8.0 → WebSphere Customization Toolbox***.

2. Click the **Welcome** tab. Then, click **Profile Management Tool (z/OS only)**, as shown in Figure 5-2. Click **Launch Selected Tool**.

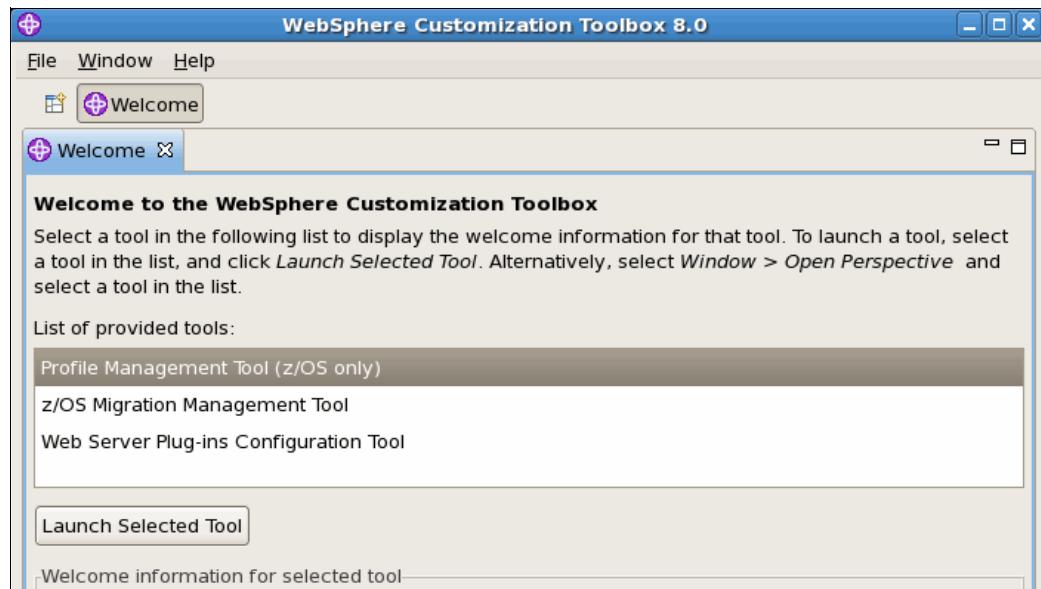


Figure 5-2 WebSphere Customization Toolbox welcome

3. On the Customization Locations tab, create a new location by clicking **Add**, as shown in Figure 5-3.

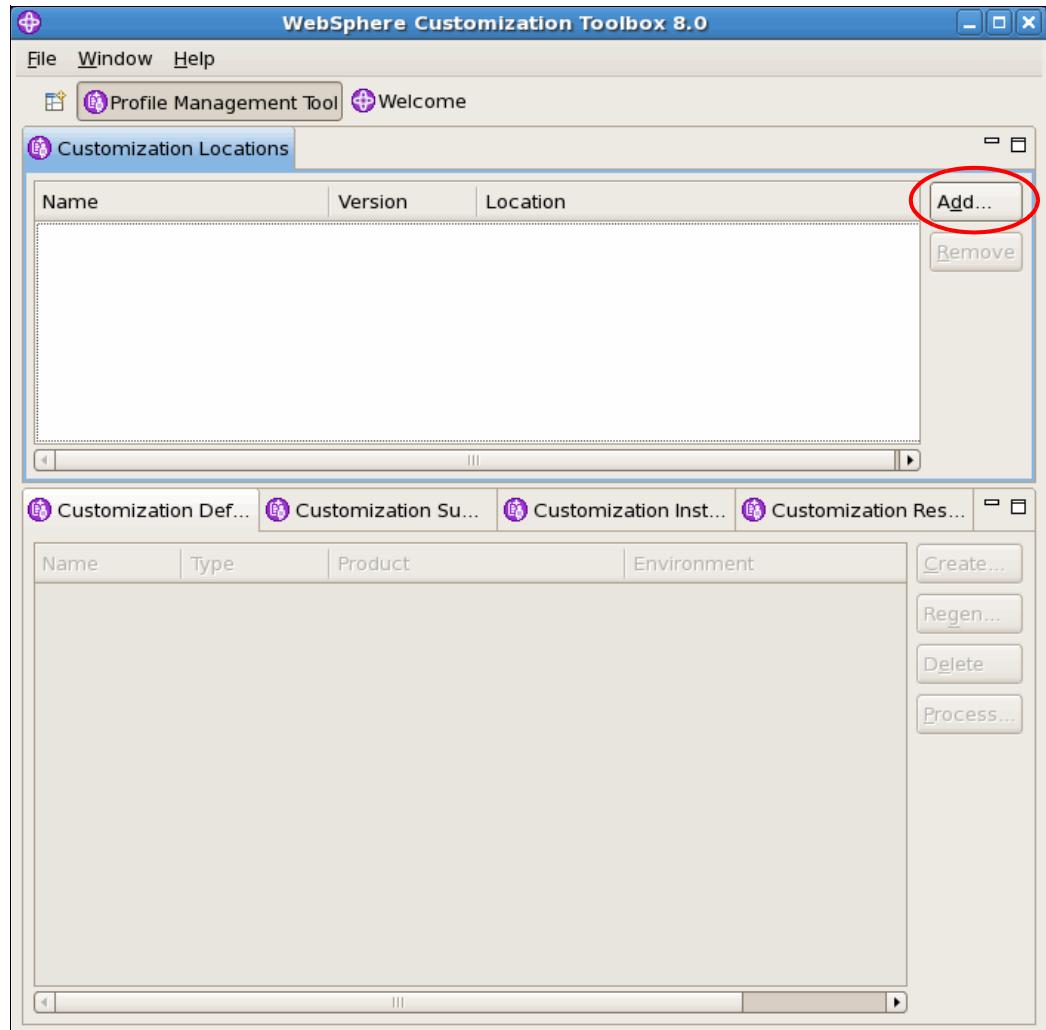


Figure 5-3 Add a new location

4. The definitions that contain the generated customization jobs are stored at the location that you define in the Add Customization dialog box. Enter the following information, as shown in Figure 5-4:
 - a. Select **Create a new customization location**, and enter a name for the new location.
 - b. Click **8.0** from the Version drop-down menu, and click the **Browse** button to define the corresponding folder for this location.
 - c. Click **Finish** to create this location.

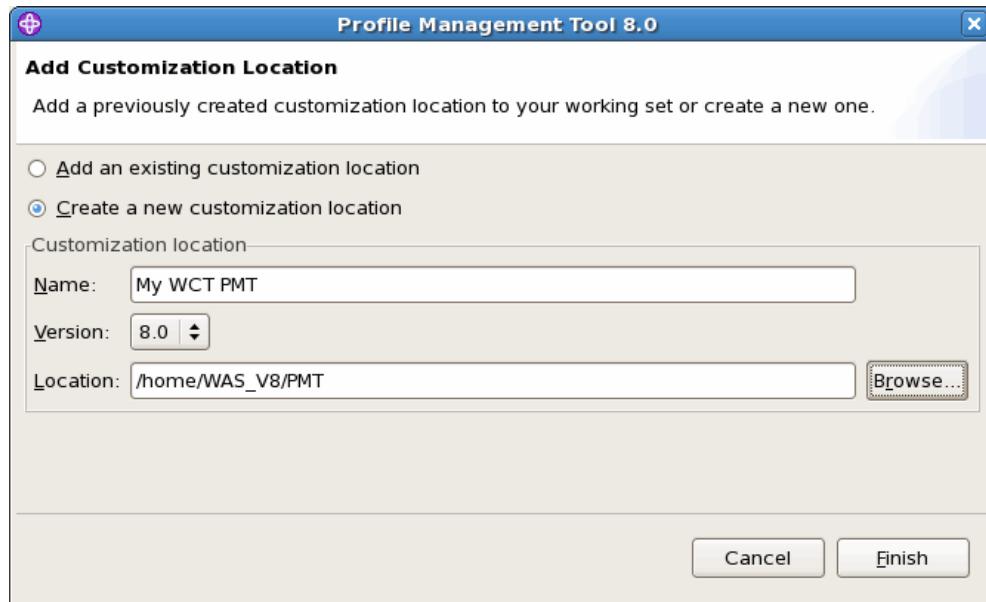


Figure 5-4 Create the customization location

Tip: You can create a single location to hold all the definitions. Depending on the size of your environment, using a single location for all definitions can become confusing. Consider creating one location per cell definitions as a method of organizing your definitions.

The Profile Management Tool main windows shows the new Customization Location, as shown in Figure 5-5.

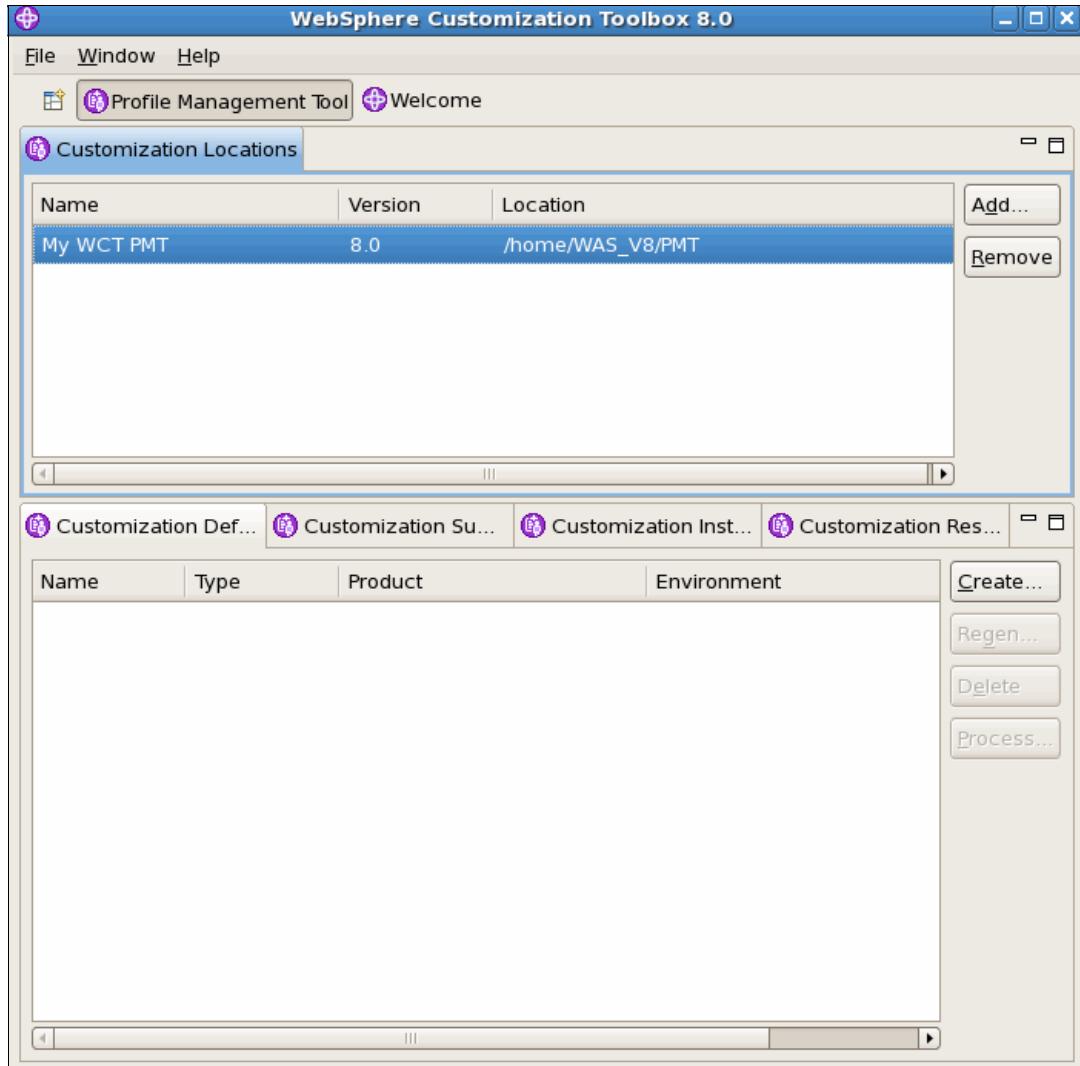


Figure 5-5 Profile Management Tool main window

Using response files: Each time you create a customization definition, it is stored in a directory in the selected customization location. A corresponding response file is generated in the same directory.

You can display this response file by switching to the Customization Response File tab shown in Figure 5-5, which is the last tab in the bottom portion of the window. You can use this response file when creating future profiles to populate the input fields with values contained in the response file.

Configuring additional users: If your daemon and control region adjunct processes must run using different user IDs from the associated control region process, click **Window** → **Preferences** → **Profile Management Tool** in the WebSphere Customization Toolbox. Then, select **Enable unique user IDs for daemon and adjunct** and click **Apply**.

With this setting, you get an additional window when you build a customization definition that allows you to specify additional user IDs for processes that are relevant to the profile (that is, for the controller adjunct and daemon processes).

5.3 Creating a sample z/OS Network Deployment cell

This section demonstrates how to use the Profile Management tool to create a z/OS Network Deployment cell (deployment manager and application server). This configuration is representative and inclusive of the windows and steps that you will encounter with the other environments.

Figure 5-6 shows the flow to generate the jobs for this sample configuration.

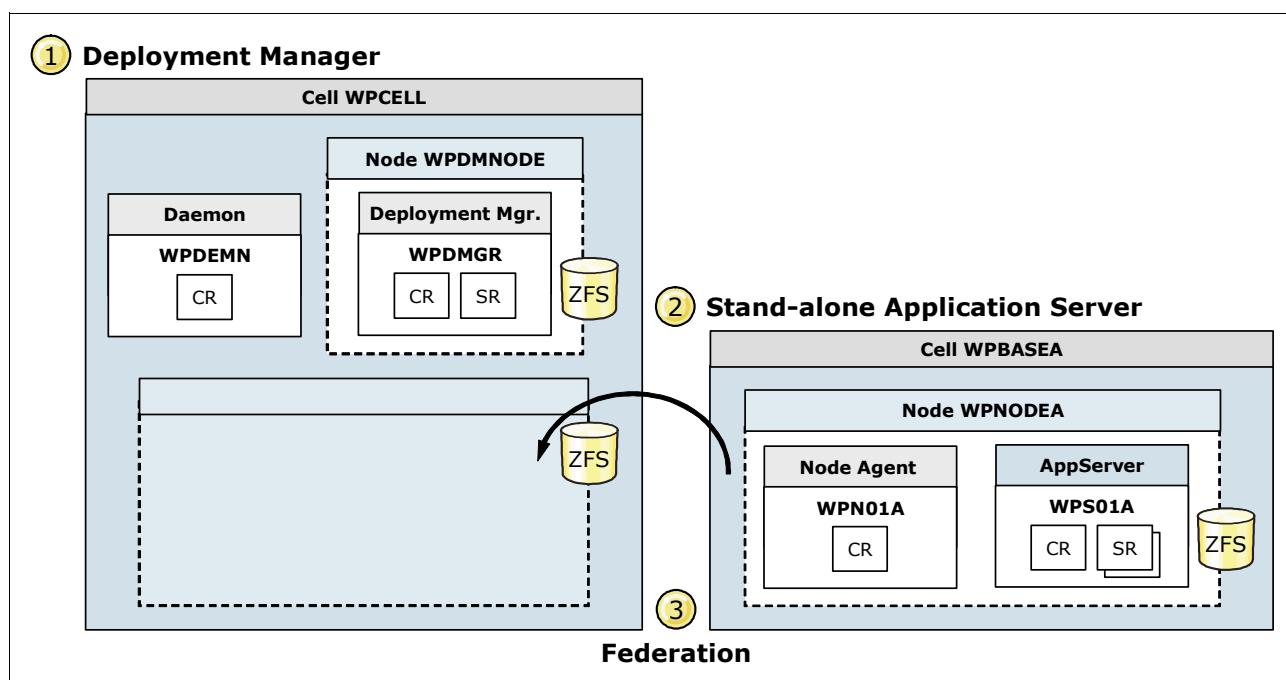


Figure 5-6 Configuration objectives

Building the sample environment involves the following steps:

1. Creating a deployment manager definition
2. Creating the base application server definition
3. Federating an application server

5.3.1 Creating a deployment manager definition

When building the sample environment, you first create the deployment manager definition.

Creating the customization definition

To begin, run the Profile Management tool to create the custom definition:

1. On the Profile Management Tool main window, click **Create**.
2. Click **Management** (as shown in Figure 5-7) and then click **Next**.

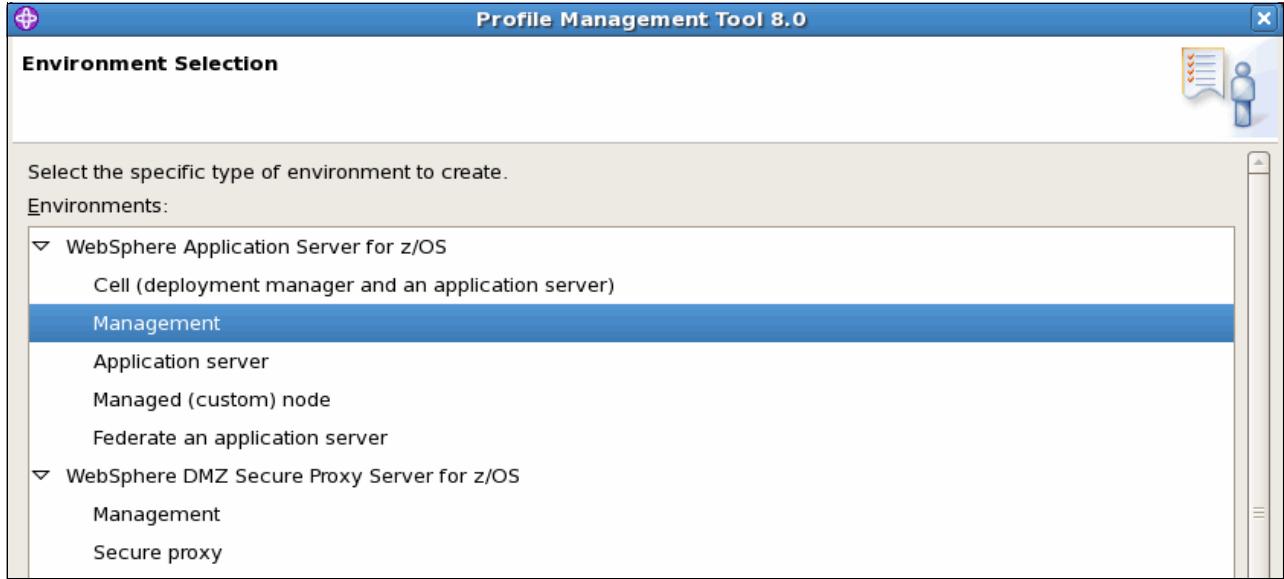


Figure 5-7 Environment Selection window

3. Select **Deployment manager** and then click **Next**.

4. In the Customization Definition Name window, shown in Figure 5-8, complete the following fields:

- Customization definition name

Specify the customization profile that you are about to create. This name is not transported to your host system.

- Response file path name

Specify a saved file with values from a previously created configuration. Using a previously created configuration populates the fields throughout the windows that follow with the values that are contained in the response file. This field is optional.

Because you are creating a profile for the first time, you might not have this file. A response file is written each time a z/OS customization definition is created, and its name is the customization definition name itself with the extension *.responseFile*. The file is created in the root directory for the customization definition. Normally, you specify a response file from a customization definition of the same type as the definition that you are about to define. However, you can use a response file from a similar customization type to pre-load most of the default values.

After you complete the fields, click **Next**.

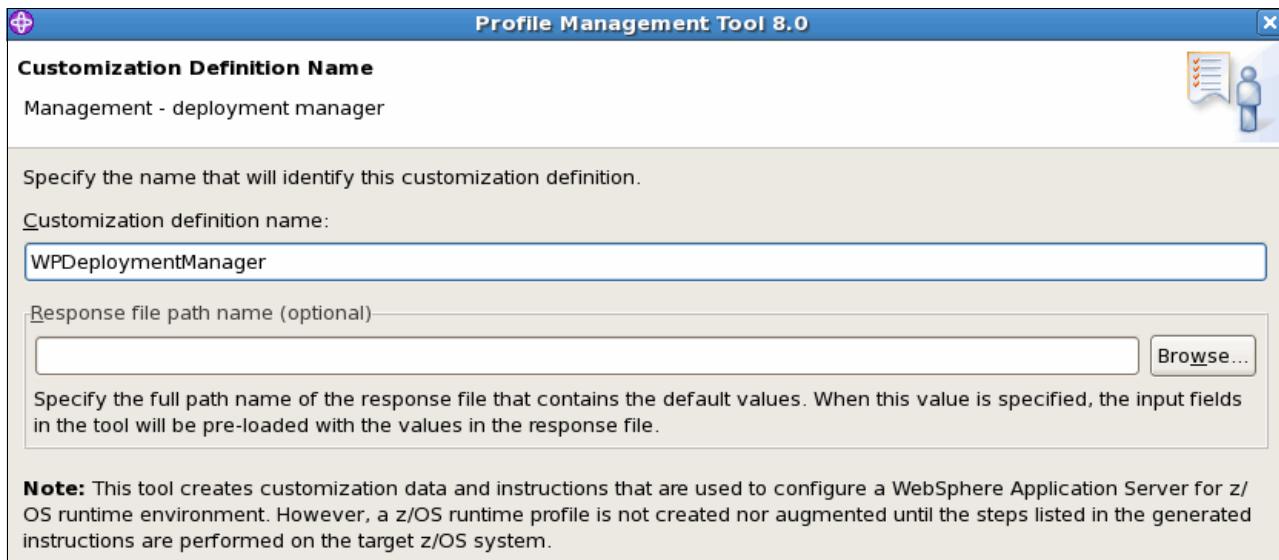


Figure 5-8 Customization Definition Name

5. In the Default values window (shown in Figure 5-9), specify defaults for GID and UID values, name and user ID defaults based on a two character prefix that identifies the cell, and specify a default range for ports that are assigned to the process. Click **Next**.

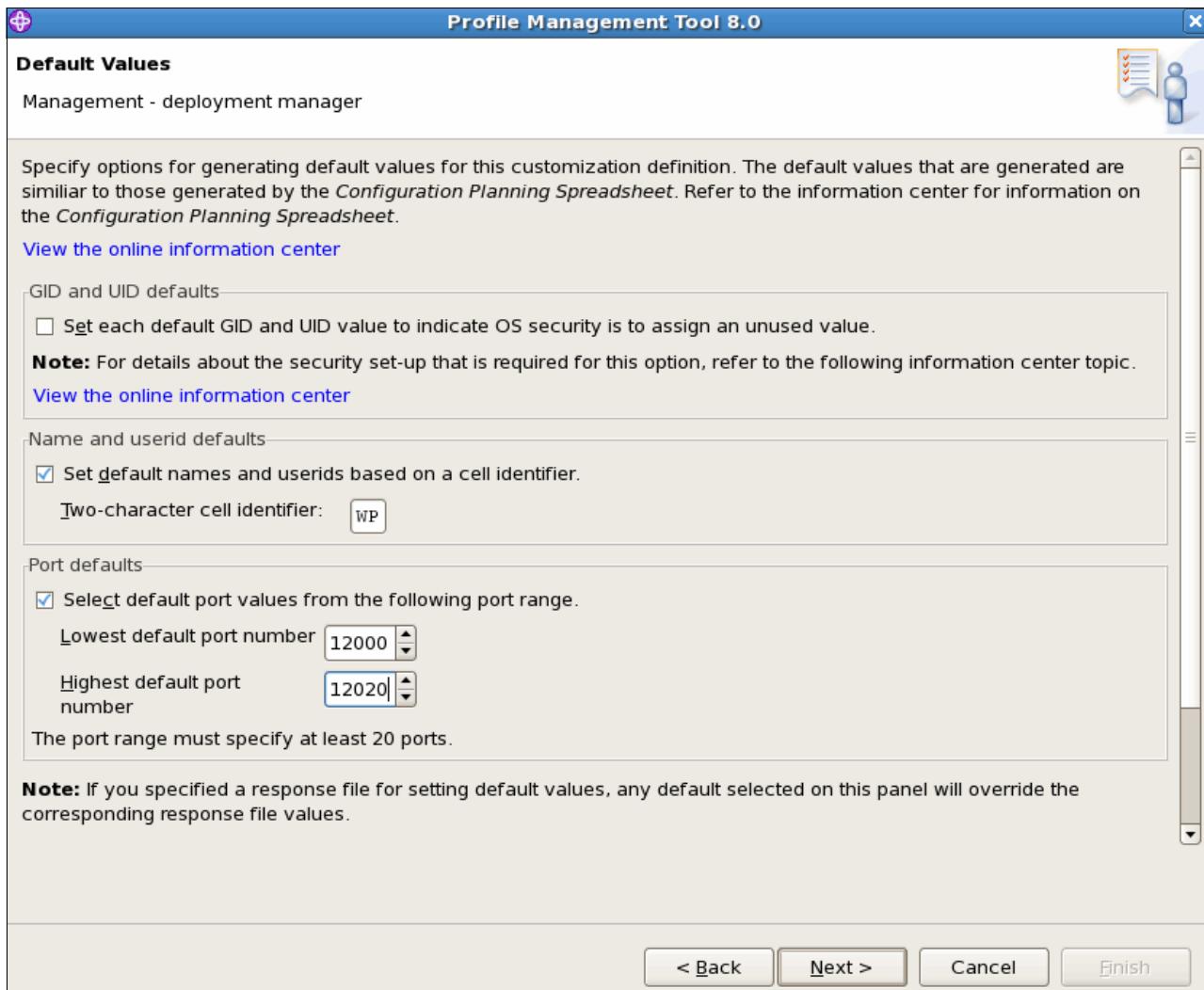


Figure 5-9 Specify default values

6. In the Target Data Sets window (shown in Figure 5-10), specify a high level qualifier for the target z/OS data sets that will contain the generated jobs and instructions.

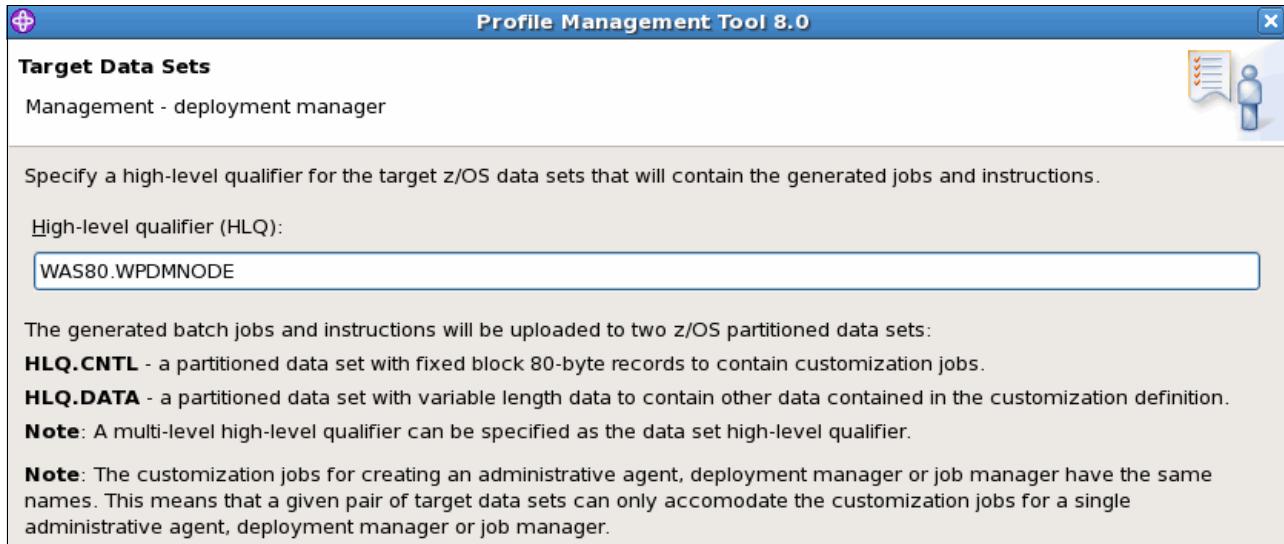


Figure 5-10 Target Data Sets

The high level qualifier can be composed of multiple qualifiers up to 39 characters. When a customization profile is uploaded on the target z/OS system, the generated jobs and files are written on a pair of data sets. The same data sets can be reused for a future installation; however, as a best practice, you should create a new pair of data sets for every new profile installation.

A good planning and naming convention is crucial when defining this type of information. As a best practice, try to set the high level qualifier according to the version and release of WebSphere Application Server for z/OS, the task you are performing, and the cell (and, in some cases, the node name) you are configuring.

In this case, the following data sets are created when the customization profile is uploaded to the target z/OS system:

- WAS80.WPDMNODE.CNTL
- WAS80.WPDMNODE.DATA

The CNTL data set is a partitioned data set with a fixed block 80-byte records that keeps the customization jobs. The DATA data set is a partitioned data set as well, but with variable length data to contain the other customization data.

Click **Next**.

Data set names: After you create the customization profile, you cannot change the data set names, because all jobs are based on these data set names.

7. The Configure Common Groups window (shown in Figure 5-11) contains the fields to configure common groups.

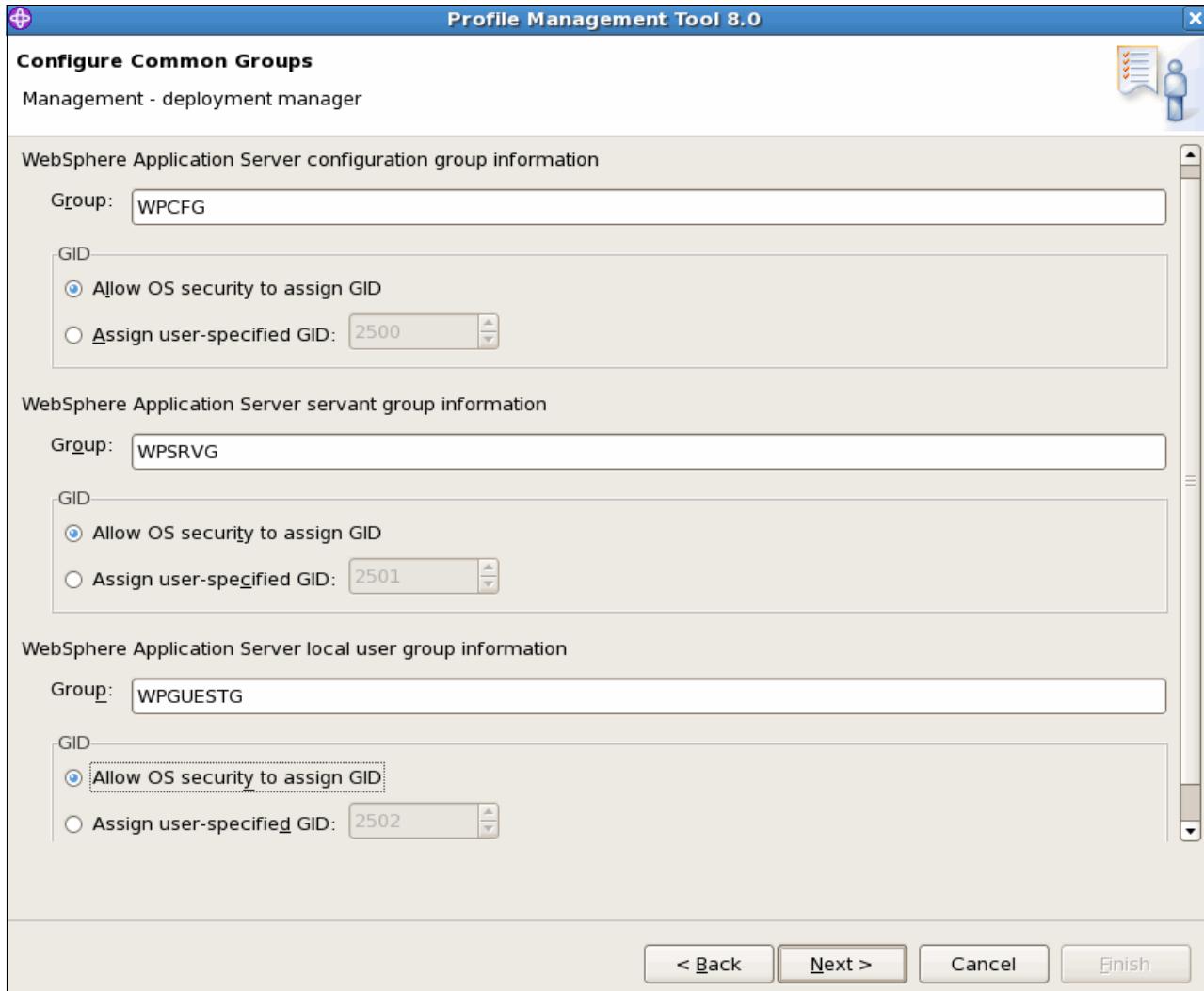


Figure 5-11 Configure Common Groups

Provide the following information for this window:

- WebSphere Application Server Configuration Group Information
Used to specify the group name for the WebSphere Application Server administrator user ID and all server user IDs.
- WebSphere Application Server Servant Group Information
Used to connect all servant user IDs to this group. You can use it to assign subsystem permissions, such as DB2 authorizations, to all servants in the security domain.
- WebSphere Application Server Local User Group Information
Specify the local client group. This group provides minimal access to the cell.

- WebSphere Application Server user ID home directory

Specify a new or existing z/OS file system directory in which home directories for WebSphere Application Server for z/OS user IDs will be created by the customization process. Note that this directory does not need to be shared among z/OS systems in a WebSphere Application Server cell.

Click **Next**.

8. The Configure Common Users window (shown in Figure 5-12) contains configuration information about the common users.

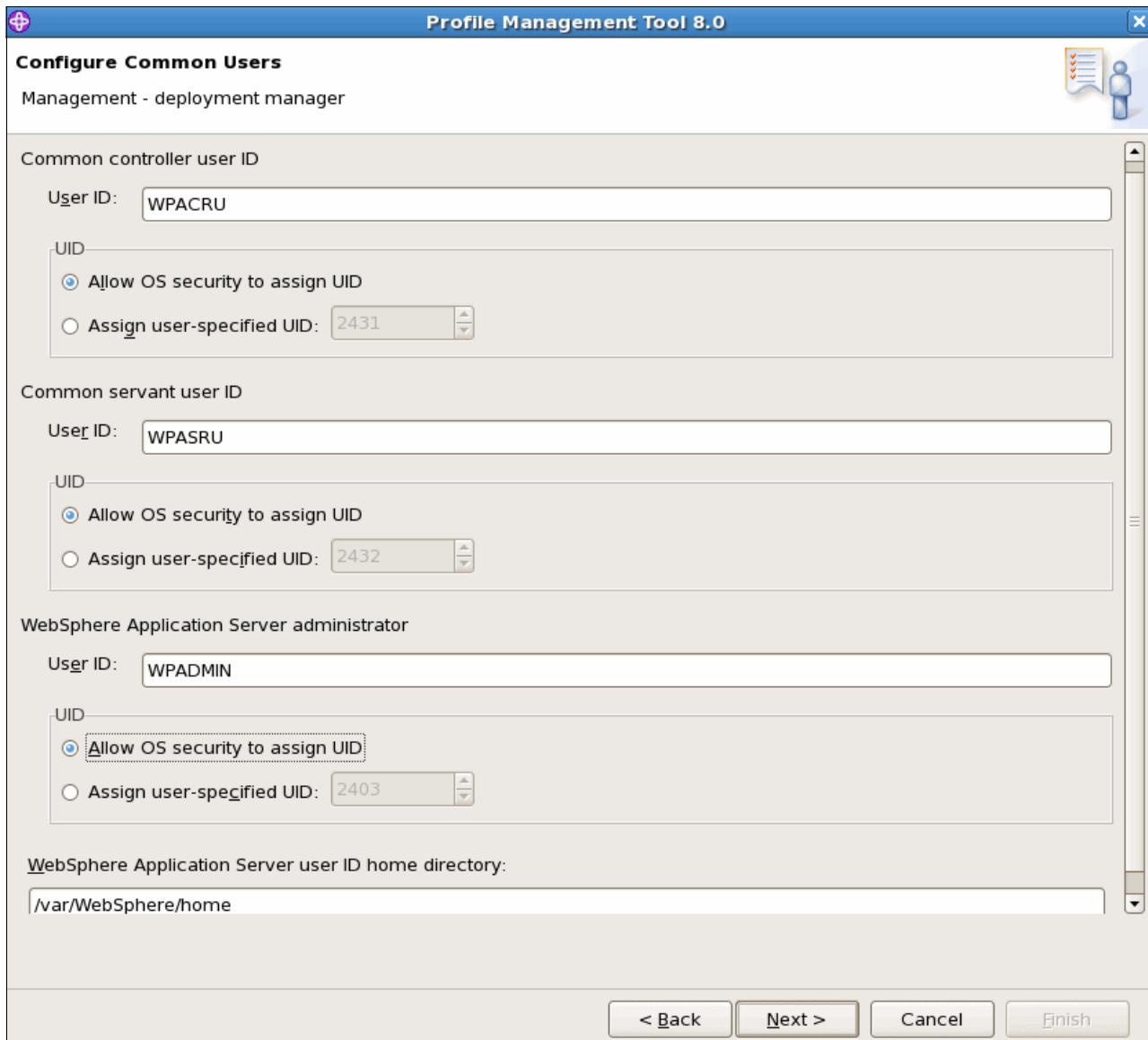


Figure 5-12 Configure Common Users

In this window, enter the following information:

- Common controller user ID

Specifies the user ID that is associated with all the control regions and the daemon. This user ID also owns all of the configuration file systems.

- Common servant user ID
Specifies the user ID that is associated with the servant regions.
 - WebSphere Application Server administrator user ID
Defines the initial WebSphere Application Server administrator. This ID must have the WebSphere Application Server configuration group as its default UNIX System Services group. The UNIX System Services UID number for the administrator user ID is specified here, and must be a unique numeric value between 1 and 2,147,483,647.
- Click **Next**.
9. The System and Data Set Names window (shown in Figure 5-13) requests system and data set names.



Figure 5-13 System and Data Set Names

In this panel, complete the following information:

- System name: The system name of the target z/OS system
- Sysplex name: The sysplex name of the target z/OS system

System and sysplex names: If you are not sure of the system and sysplex names for your target z/OS system, use the D SYMBOLS console command on the target z/OS system to display them.

- PROCLIB data set name: The PROCLIB data set where the WebSphere Application Server for z/OS catalogued procedures are to be added.

Click **Next**.

10. The Cell, Node and Server Names window (shown in Figure 5-14) requests the cell, node, and server names.

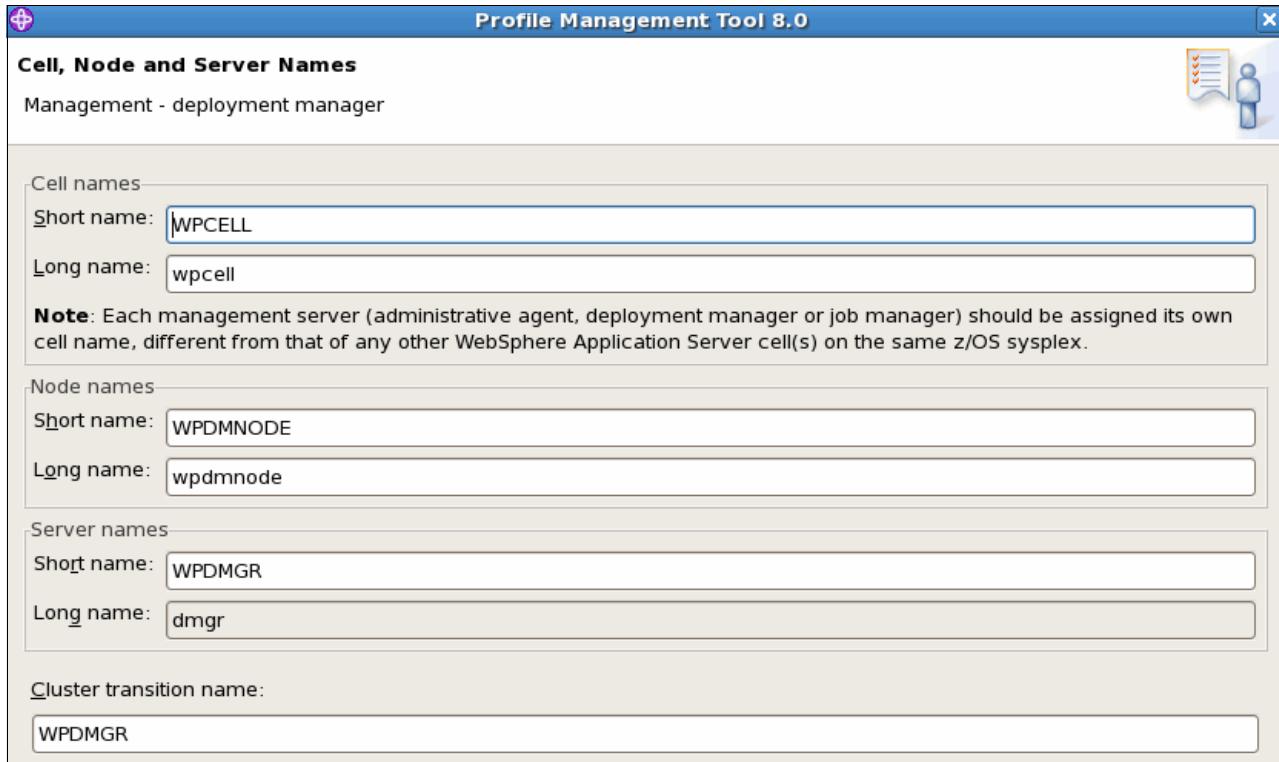


Figure 5-14 Cell, Node and Server Names

In this window, complete the following information:

- Cell short name
Identifies the cell to z/OS facilities, such as SAF.
- Cell long name
Defines the primary external identification of this WebSphere Application Server for this z/OS cell. This name identifies the cell as displayed through the administrative console.
- Deployment manager short name
Specifies the name that identifies the node to z/OS facilities, such as SAF.
- Deployment manager long name
Identifies the primary external identification of this WebSphere Application Server for the z/OS node. This name identifies the node as displayed through the administrative console.
- Deployment manager server short name
Identifies the server to z/OS facilities, such as SAF. The server short name is also used as the server JOBNAME.
- Deployment manager server long name
Specifies the name of the application server and the primary external identification of this WebSphere Application Server for the z/OS server. This name identifies the server as displayed through the administrative console.

- Deployment manager cluster transition name

Defines the WLM application environment (WLM APPLENV) name for the deployment manager. If this is a server that is converted into a clustered server, this name becomes the cluster short name. The cluster short name is the WLM APPLENV name for all servers that are of the same cluster.

Click **Next**.

11. The Configuration File System window (shown in Figure 5-15) requests configuration file system information for your z/OS system. The file system can be either HFS or zFS. It is used to hold WebSphere Application Server configuration information.

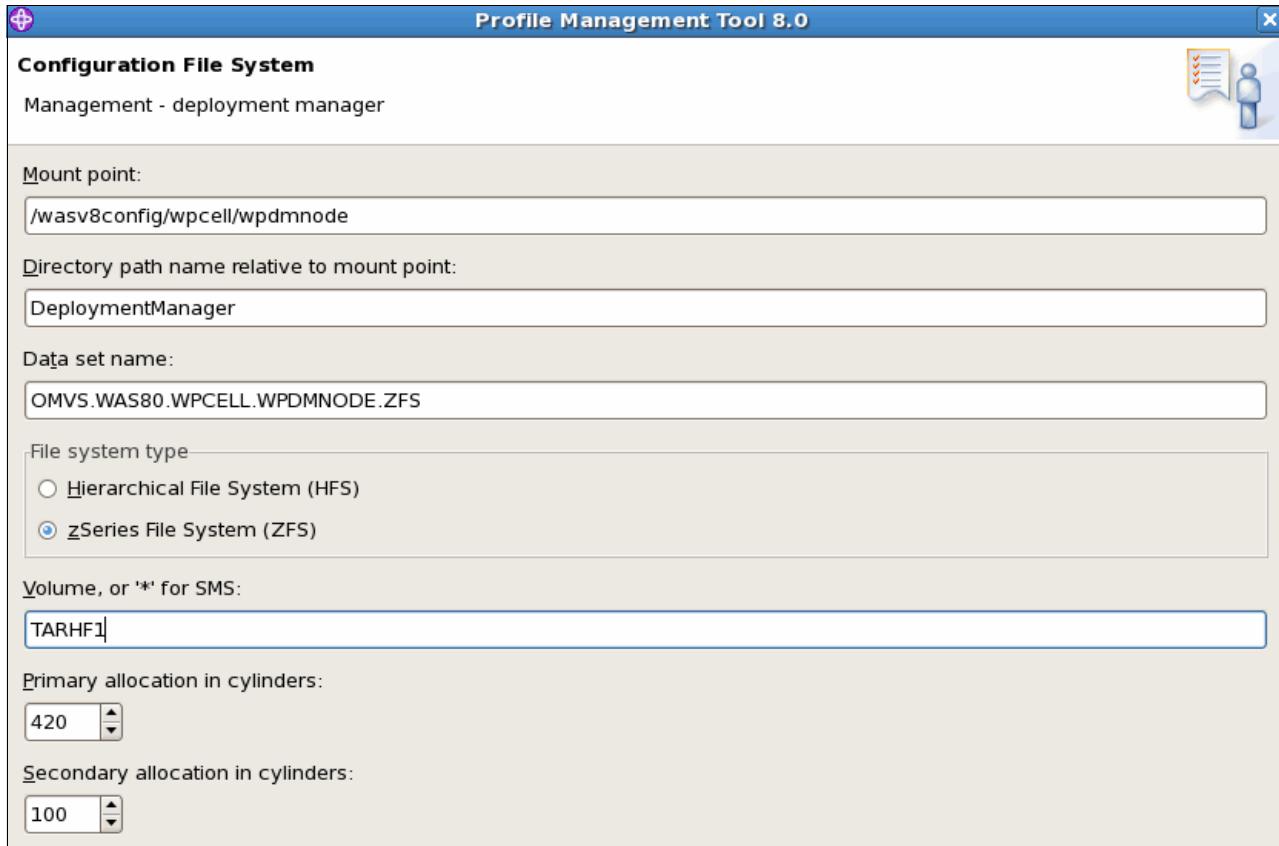


Figure 5-15 Configuration File System

In this window, complete the following information:

- Mount point

Specifies the read/write HFS directory where application data and environment files are written. The customization process creates this mount point if it does not already exist.

- The directory path name relative to the mount point

Defines the directory that is used for the deployment manager home directory.

- Data set name

Specifies the file system data set that you will create and mount at the specified mount point.

- File system type

Select the files system type to allocate and mount the configuration file system data set using as either HFS or zFS.

- Volume, or "*" for SMS

Specify either the DASD volume serial number to contain the above data set or "*" to let SMS select a volume. Using "*" requires that SMS automatic class selection (ACS) routines be in place to select the volume. If you do not have SMS set up to handle data set allocation automatically, list the volume explicitly.

- Primary allocation in cylinders

Set the initial size allocation for the configuration file system data set. In the application server, the total space needed for this data set increases with the size and number of the installed applications. The minimum suggested size is 250 cylinders (3390).

- Secondary allocation in cylinders

Specifies the size of each secondary extent. The minimum suggested size is 100 cylinders.

Click **Next**.

12. The WebSphere Application Server Product File System window (shown in Figure 5-16) defines the product file system directory and allows you to set up an intermediate symbolic link.

Best practice: Use intermediate symbolic links for flexibility when applying maintenance and running with different versions of the code.

After you complete this information, click **Next**.

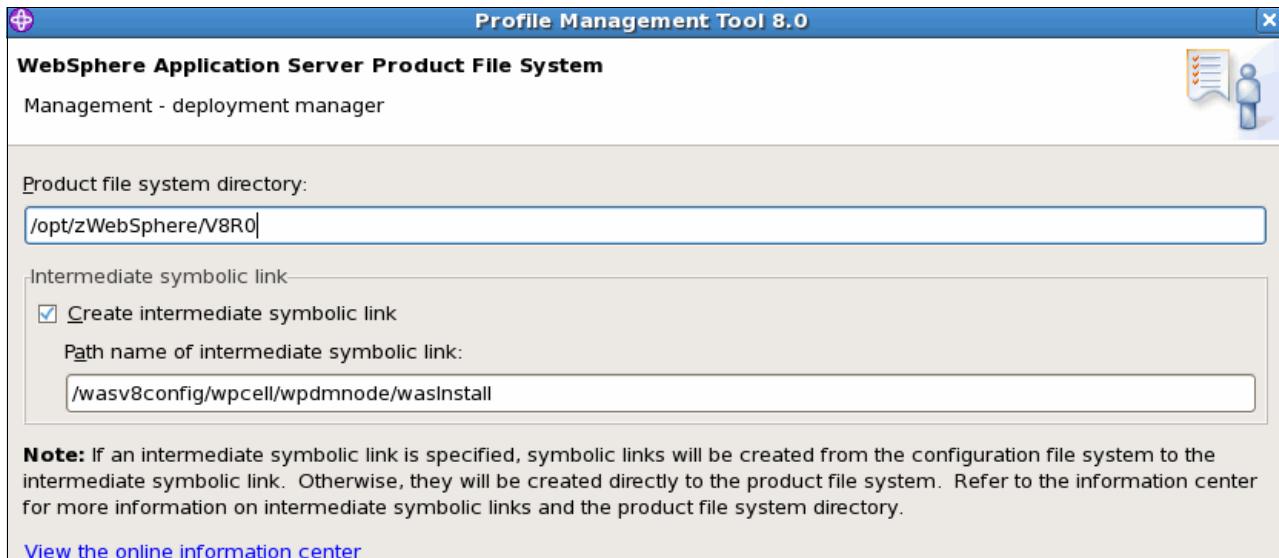


Figure 5-16 WebSphere Application Server Product File System

13. In the Process Definitions window (shown in Figure 5-17), enter the job names, procedure names, and user IDs to use for each process.

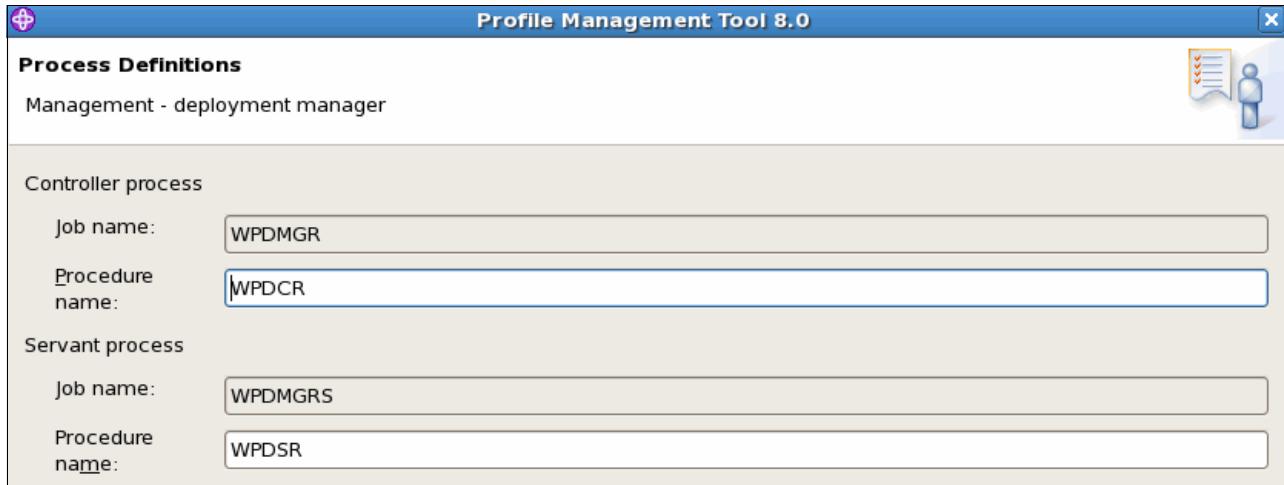


Figure 5-17 Process Definitions

Enter the following information:

– Deploy manager controller process

The job name is specified in the IBM MVS™ START command JOBNAME parameter that is associated with the control region. This job name is the same job name as the server short name, and it cannot be changed during customization. The procedure name is the member name in your procedure library to start the control region. The user ID is the user ID that is associated with the control region.

– Deploy manager servant process

Specify the job name used by WLM to start the servant regions. This job name is set to the server short name followed by the letter *S*, and it cannot be changed during customization. The procedure name is the member name in your procedure library to start the servant regions. The user ID is the user ID that is associated with the servant regions.

After you complete this information, click **Next**.

14. The Port Values Assignment window (Figure 5-18) requires you to specify the ports to use for each process. Planning is important to avoid port conflicts, so be sure that you have all the values you need to complete this window.

The screenshot shows the 'Profile Management Tool 8.0' window titled 'Port Values Assignment'. It is a configuration interface for a deployment manager. The window contains several input fields for specifying port numbers:

Setting	Value
Node host name or IP address:	wtsc58.itso.ibm.com
JMX SOAP connector port:	12002
Cell Discovery Address Port (6):	12012
ORB listener IP address:	*
ORB port:	12003
ORB SSL port:	12004
HTTP transport IP address (2):	*
Administrative console port:	12005
Administrative console secure port:	12006
Administrative interprocess communication port (X):	12009
High Availability Manager Communication Port (DCS):	12010
DataPower appliance manager secure inbound port:	12011

Figure 5-18 Port Values Assignment

After entering the required ports, click **Next**.

15. In the Location Service Daemon Definitions window (shown in Figure 5-19), enter the location service daemon settings. The location daemon service is the initial point of client contact in WebSphere Application Server for z/OS. The server contains the CORBA-based location service agent that places sessions in a cell. All RMI/IIOP IORs (for example, enterprise beans) establish connections to the location service daemon first, then forward them to the target application server.

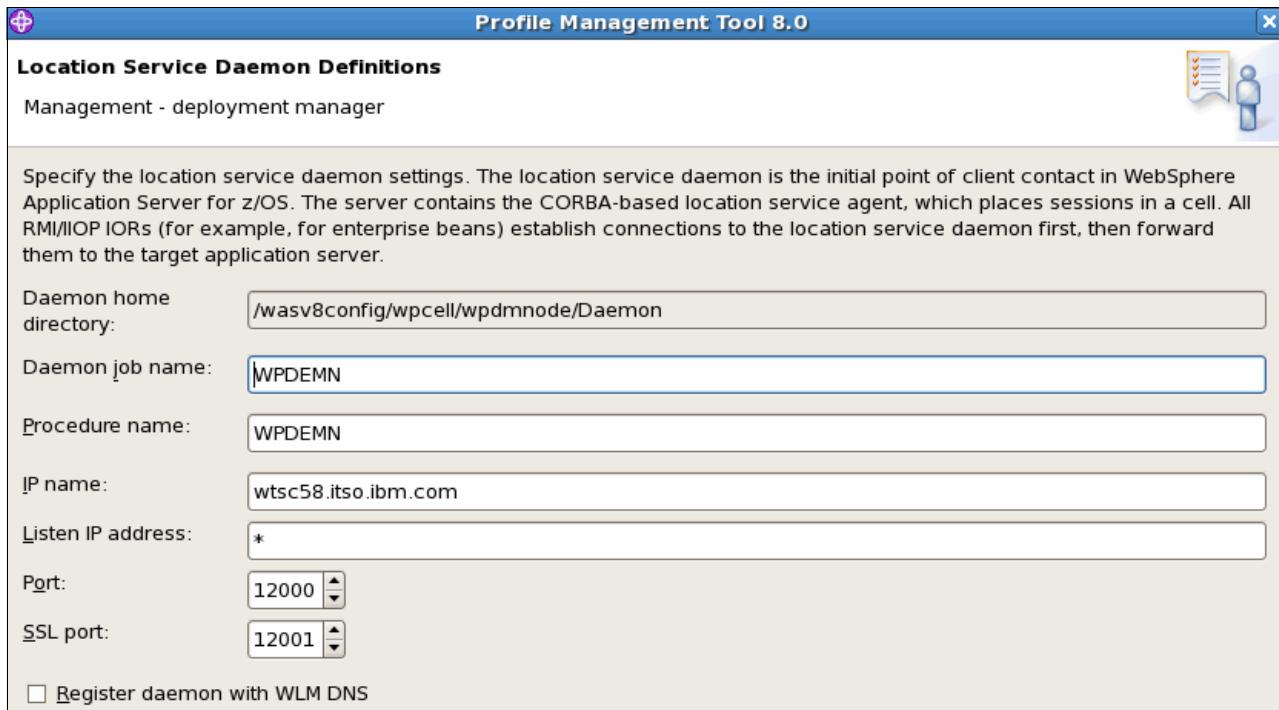


Figure 5-19 Location Service Daemon Definitions

In this window, enter the following information:

- Daemon home directory

The directory in which the location service daemon resides. This setting is set to the configuration file system mount point / daemon and cannot be changed.

- Daemon job name

Specifies the job name of the location service daemon, which is specified in the JOBNAME parameter of the MVS start command used to start the location service daemon. When configuring a new cell, be sure to choose a new daemon job name value. A server automatically starts the location service daemon if it is not already running.

- Procedure name

The member name in your procedure library to start the location service daemon.

- IP name

The fully qualified IP name, registered with the Domain Name Server (DNS), that the location service daemon uses. The default is your node host name. In a sysplex, you should consider using a virtual IP address (VIPA) for the location service daemon IP name. Select the IP name for the location service daemon carefully. You can choose any name that you want, but, after being chosen, it is difficult to change, even in the middle of customization.

- Listen IP

The address at which the daemon listens. Select either * or a dotted IP address for this value.

- Port

Specify the port number on which the location service daemon listens.

- SSL port

The port number on which the location service daemon listens for SSL connections.

Important: Choose the IP name and port number carefully, because these names are difficult to change.

- Register daemon with WLM DNS

If you use the WLM DNS (connection optimization), you must register your location service daemon. Otherwise, do not register your location service daemon. Only one location service daemon per LPAR can register its domain name with WLM DNS; if you have multiple cells in the same LPAR and register more than one location service, it will fail to start.

Click **Next**.

16. In the SSL Customization window (shown in Figure 5-20), enter the SSL customization values.



Figure 5-20 SSL Customization

Enter the following information:

- Certificate authority keylabel

The name that identifies the certificate authority (CA) to be used in generating server certificates.

- Generate certificate authority (CA) certificate

Selected to generate a new CA certificate. Do not select this option to have an existing CA certificate generate server certificates.

- Expiration date for certificates

Used for any X509 Certificate Authority certificates created during customization, as well as the expiration date for the personal certificates generated for WebSphere Application Server for z/OS servers. You must specify this even if you have not selected **Generate Certificate Authority (CA) certificate**.

- Default SAF keyring name

The default name given to the RACF keyring used by WebSphere Application Server for z/OS. The keyring names created for repertoires are all the same within a cell.

- Enable SSL on the location service daemon check box

Select this option if you want to support secure communications using Inter-ORB Request Protocol (IIOP) to the location service daemon using SSL. If selected, a RACF keyring is generated for the location service daemon to use.

After completing the required SSL information, click **Next**.

17. The next window allows you to select the user registry to be used for administrative security. Choose from the following options:

- z/OS security product

This option uses the z/OS system's SAF compliant security product, such as IBM RACF or equivalent, to manage WebSphere Application Server identities and authorization according to the following rules:

- The SAF security database will be used as the WebSphere user repository.
- SAF EJBROLE profiles will be used to control role-based authorization, including administrative authority.
- Digital certificates will be stored in the SAF security database.

z/OS security note: Select the z/OS security product option if you are planning to use the SAF security database as your WebSphere Application Server registry or if you plan to set up an LDAP or custom user registry whose identities will be mapped to SAF user IDs for authorization checking. For this security option, you must decide whether to set a security domain name, and choose an administrator user ID and an unauthenticated (guest) user ID.

- WebSphere Application Server security

The WebSphere Application Server administrative security option is used to manage the Application Server identities and authorization according to these rules:

- A simple file-based user registry will be built as part of the customization process.
- Application-specific role binds will be used to control role-based authorization.
- The WebSphere Application Server console users and groups list will control administrative authority.
- Digital certificates will be stored in the configuration file system as keystores.

Digital certificates note: Choose this option if you plan to use an LDAP or custom user registry without mapping to SAF user IDs. (The file-based user registry is not a best practice for production use.)

- No security

Although it is not a best practice, you can disable administrative security. If you choose this security option, there are no other choices to make. Your WebSphere Application Server environment will not be secured until you configure and enable security manually. You can enable security manually later using the administrative console or using Jython scripts.

For our scenario, we used the “Use z/OS security product” option.

Click **Next**.

18. Enter the parameters shown in Figure 5-21 after you select the z/OS security product option.

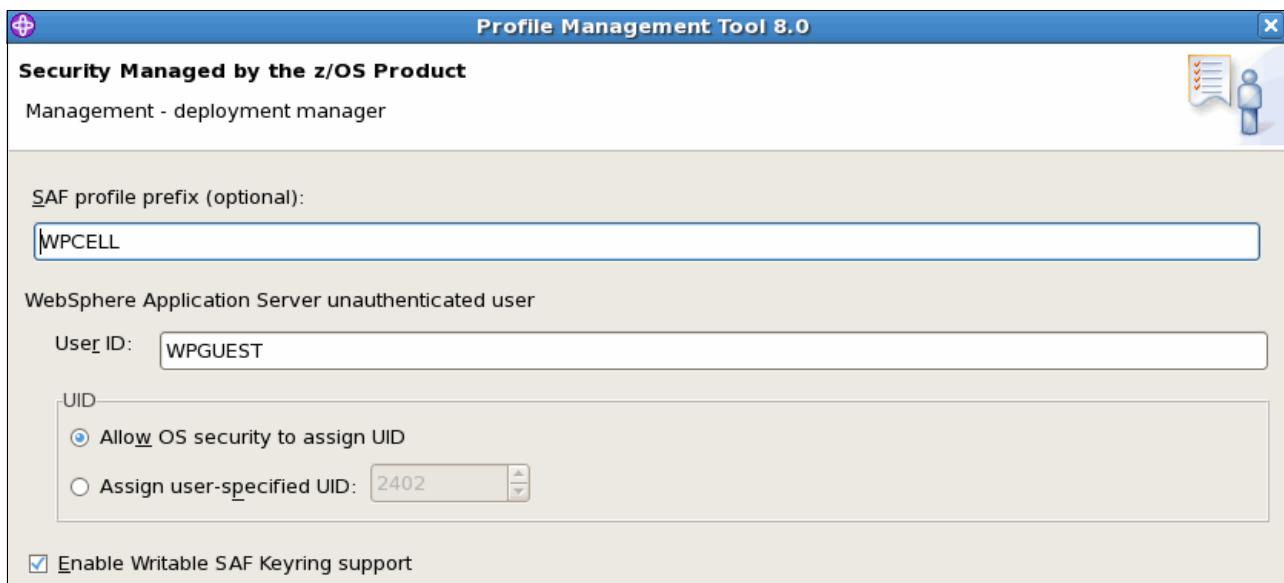


Figure 5-21 Security Managed by the z/OS Product

Complete the following information:

- (Optional) SAF profile prefix (formally known as the *Security domain identifier*)

This optional parameter is used to distinguish between APPL or EJBROLE profiles based on security domain name. It provides an alphanumeric security domain name of one to eight characters. Internally, this sets SecurityDomainType to the string cellQualified.

All servers in the cell will prepend the security domain name you specify to the application-specific J2EE role name to create the SAF EJBROLE profile for checking. The security domain name is not used, however, if role checking is performed using WebSphere Application Server for z/OS bindings.

The security domain name is also used as the APPL profile name and inserted into the profile name used for CBIND checks. The RACF jobs that the Customization Dialog generates create and authorize the appropriate RACF profiles for the created nodes and servers.

If you do not want to use a security domain identifier, leave this field blank.

- WebSphere Application Server unauthenticated user ID

Associated with unauthenticated client requests. It is sometimes referred to as the “guest” user ID. It should be given the RESTRICTED attribute in RACF, to prevent it from inheriting UACC-based access privileges. The UNIX System Services UID number for the user ID is specified here and is associated with unauthenticated client requests. The UID value must be unique numeric values between 1 and 2,147,483,647.

- Enable Writable SAF Keyring support

This feature allows administrative console to create and sign certificates by a CA, connect to keyrings, remove from keyrings, and import, export, and renew.

All certificates created with the writable keyring support are generated and signed by Java code and not by SAF. In this case, the writable keyring support only uses SAF to store the generated certificates.

The writable keyring support is completely optional. New keystores and truststores marked as read-only can be created independently from the writable keyring support. When using the read-only JCERACFKS and JCECCARACFS keystores, the certificates in the appropriate SAF keyring can still be viewed in the administrative console.

Click **Next**.

19. The next window allows you to tailor the JCL for the customization jobs. Enter a valid job statement for your installation on this window. The profile creation process updates the job name for you in all the generated jobs, so you need not be concerned with that portion of the job statement. If continuation lines are needed, replace the comment lines with continuation lines. Click **Next**.
20. The last window shows a short summary of the customization, including profile type and where the generated jobs will be stored. To change the characteristics of this profile, click **Back**. Otherwise, click **Create** to generate your z/OS customization jobs.
21. The Profile Management tool displays a summary window that indicates whether the jobs were created successfully or not. If the jobs were not created, a log file containing failure information will be identified. Click **Finish** to return to the Profile Management tool main window. The new deployment manager definition is listed in the Customization Definitions tab.

Uploading the jobs to the z/OS system

Next, upload these jobs and the associated instructions to a pair of z/OS partitioned data sets:

1. On the main window, select the customization definition for the profile and click **Process**. To upload the generated jobs to the target z/OS system, select the desired option. The available options are:
 - Upload to target z/OS system using FTP
 - Upload to target z/OS system using FTP over SSL
 - Upload to target z/OS system using secure FTP
 - Export to local file system

Click **Next**.

2. If you choose to upload the customization using FTP, in the upload customization definition window, enter the target z/OS system. This path must be fully qualified or the upload will fail. You must also specify the user ID and password and FTP server port.

Select the **Allocate target z/OS data sets** option to specify whether to allocate the data sets if they do not exist. If the data sets exist and are to be reused, clear this option.

Click **Finish** A progress bar displays while the upload is occurring.

Executing the jobs

After the customization profile is uploaded, the next step is to execute the jobs. The instructions for preparing for and executing the jobs can be found in the Profile Management Tool. Select the customization definition and click the **Customization Instructions** tab (Figure 5-22). These instructions are also contained in a job that has been loaded to the host.

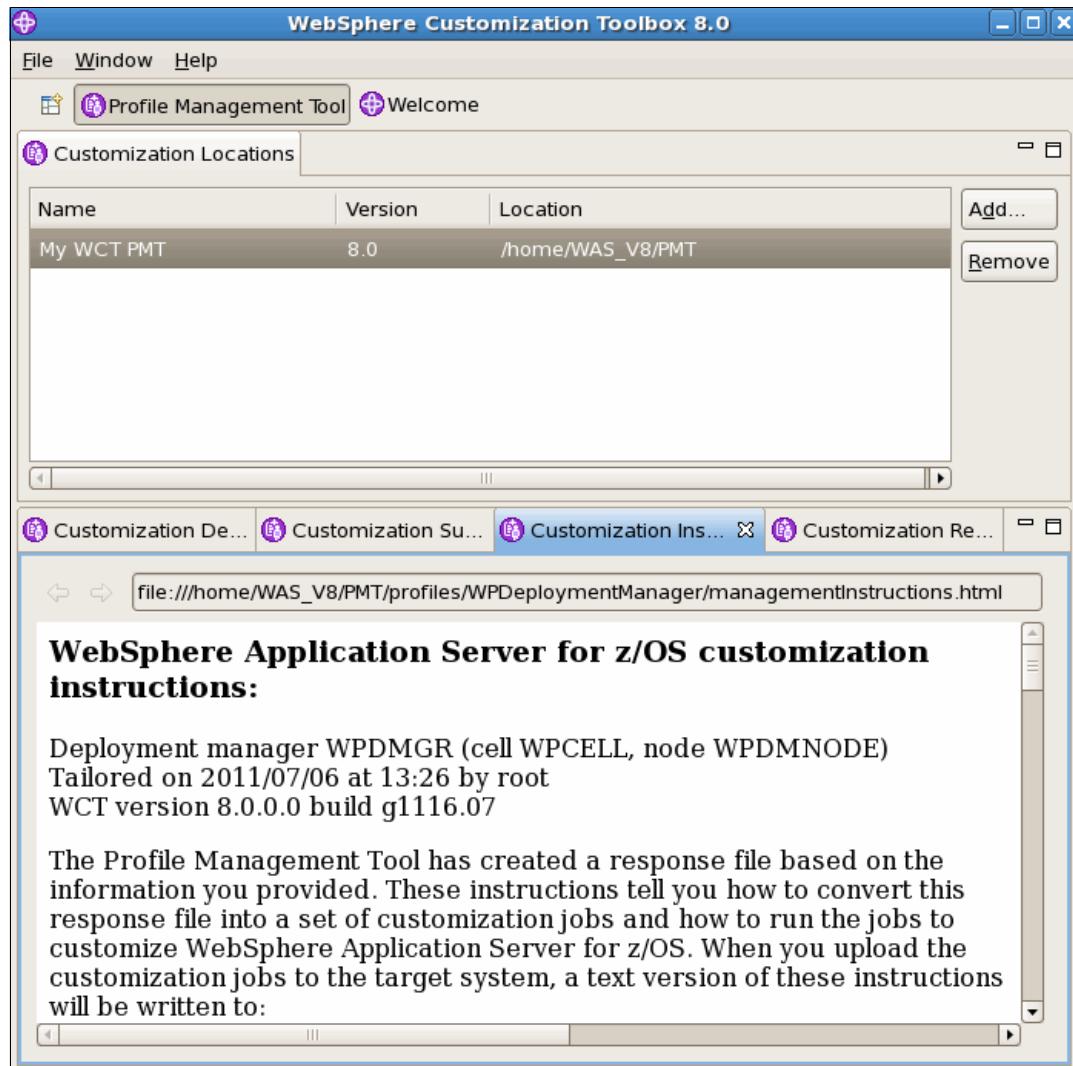


Figure 5-22 Customization definition instructions

The instructions help you determine what jobs to run, the order to run them in, and the expected results. They also tell you how to start the environment after you are done.

After the jobs run successfully, the deployment manager profile is complete.

Run the following jobs:

BBOSBRAK	Create common groups and user IDs.
BBOSBRAM	Create home directories to user IDs.
BBODBRAK	Create users and profiles required by WebSphere node.
BBODCFS	Create the mount point directory, allocate the file system, and mount it.
BBODHFS	Populate the configuration file system and prepare it for profile creation.
BBOWWPFD	Create the profile in the configuration file system.
BBODPROC	Create the procedures and copy them to proper library.

A configuration has three basic components: a file system with configuration XML, JCL start procedures, and SAF profiles. Figure 5-23 shows how the jobs are mapped to the components of the configuration.

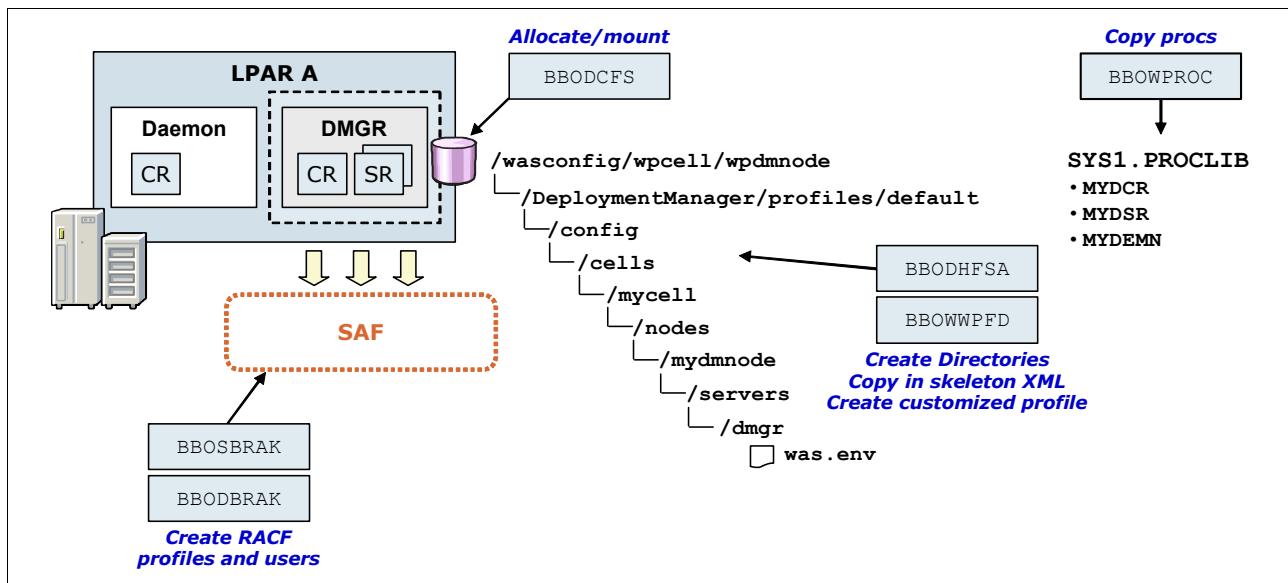


Figure 5-23 Mapping generated jobs to configuration functions

5.3.2 Creating the base application server definition

Next, create the base application server definition. Using the Profile Management tool, complete the following steps:

1. Click **Create**.
2. In the Environment Selection window (shown in Figure 5-24), click **Application server**, and click **Next**.

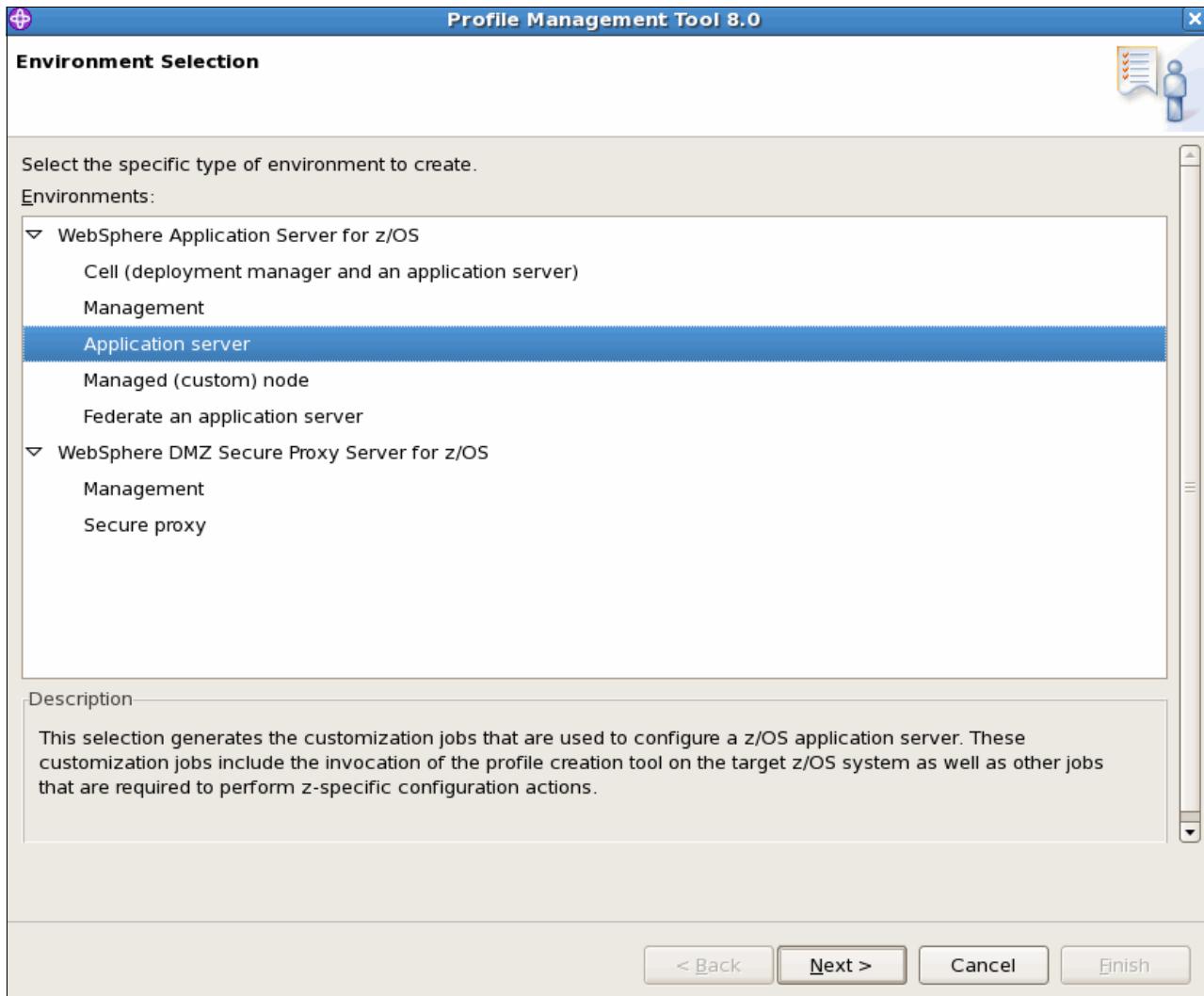


Figure 5-24 Environment Selection

3. Specify a name for the customization definition and, optionally, a response file path. The server runtime performance tuning setup contains the following selectable options:
 - Standard: Performance options set to general purpose server
 - Peak: Performance options set to environments where application updates are not often

We used the following values:

- Customization definition name: ZAppSrv01
- Server runtime performance tuning setting: Standard

Click **Next**.

4. In the Default Values window (shown in Figure 5-25), set the configuration default values.

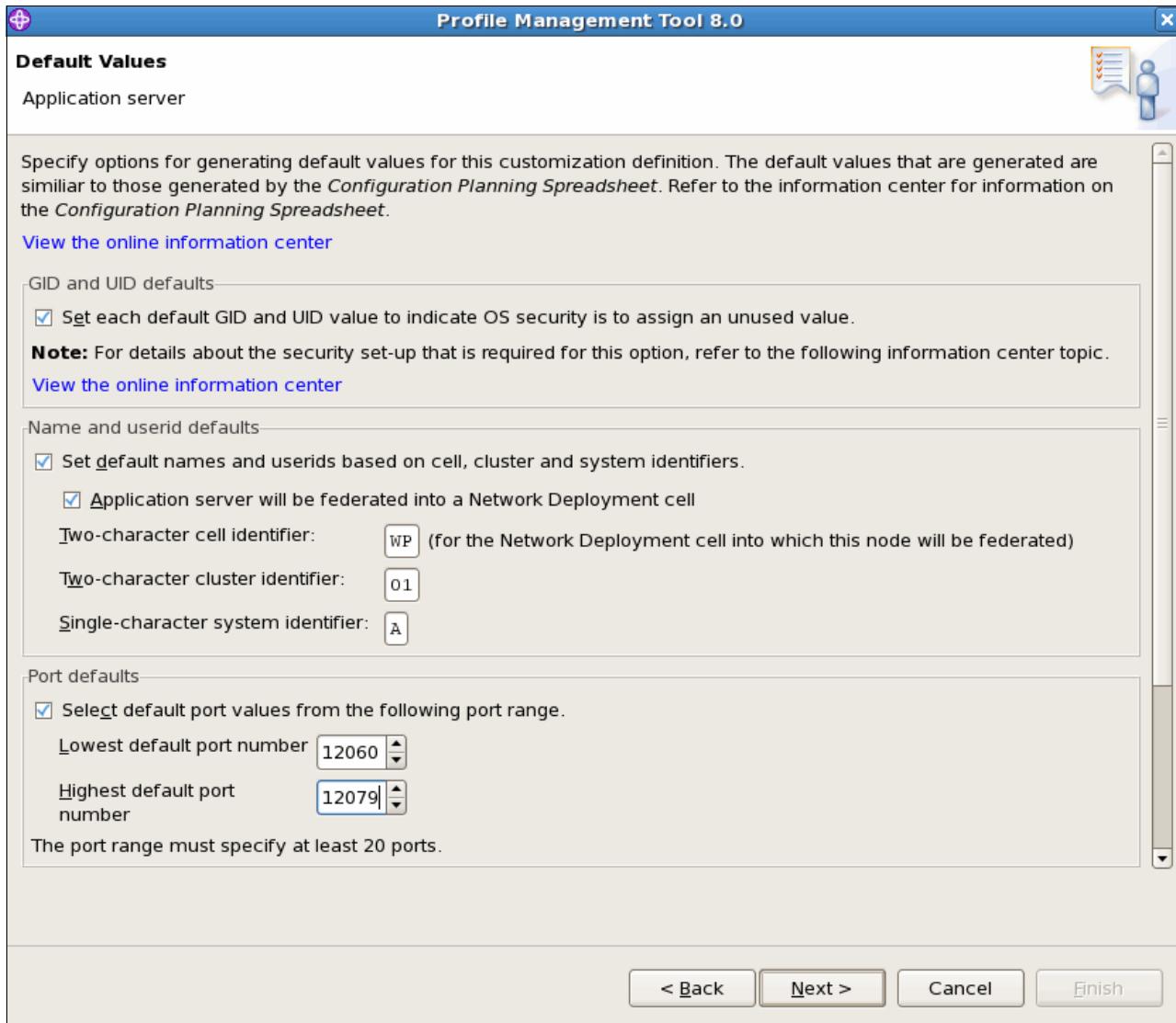


Figure 5-25 Application server default values

Important: If you specified a response file for setting default values, any default that you select here overrides the corresponding response file values.

The GID and UID defaults section contains the following selectable option:

- Set each default GID and UID value to indicate that operating-system security is to assign an unused value:

When this option is selected, each GID and UID value is defaulted to allow operating system security to assign an unused value. When this option is not selected, each GID and UID value is defaulted to an IBM-provided number.

The Name and user ID defaults contains the following selectable options:

- Set default names and user IDs based on cell, cluster, and system identifiers

When this option is selected, default cell, node, server, cluster, and procedure names as well as group names and user IDs are based on cell, cluster, and system identifiers.
 - Application server will be federated into a Network Deployment cell

Select this option to indicate that the application server will be federated into a Network Deployment cell at some point in time.
 - Two-character cell identifier

Enter a two-character cell identifier to be used to create default names and user IDs.
If you have selected the option to federate to a cell, specify the two-character cell identifier of the target Network Deployment cell.
The first character must be an alphabetic character and the second character must be an alphanumeric character. Alphabetic characters can be entered in lowercase or uppercase. The case of alphabetic characters will be adjusted as appropriate for each generated default value.
 - Two-character cluster identifier

The two-character cluster identifier to be used to create default names and user IDs.
The characters will be appended to the cluster transition name.
The characters must be alphabetic characters. The alphabetic characters can be entered in lowercase or uppercase. The case of alphabetic characters will be adjusted as appropriate for each generated default value.
 - Single-character system identifier

The single-character system identifier to be used to create default names and user IDs.
It will be appended to the short and long names for the cell, node, and server and to the appropriate process names.
The character must be an alphanumeric character. An alphabetic character can be entered in lowercase or uppercase. The case of the alphabetic character will be adjusted as appropriate for each generated default value.
 - Port defaults

Select default port values from the following port range.
When this option is not selected, each port value defaults to an IBM-provided number.
When this option is selected, each port default value is selected from the following port number range.
The port range must contain at least 20 ports.
 - Lowest default port number

The lowest number that can be assigned as a default port number.
 - Highest default port number

The highest number that can be assigned as a default port number.
- Click **Next**.
5. Next, specify the high-level qualifier for the target z/OS data sets that will contain the generated jobs and instructions that are created (refer to Figure 5-10 on page 170).

Note: You can specify a multilevel high-level qualifier as the data set high-level qualifier.

The generated batch jobs and instructions are uploaded to the following z/OS partitioned data sets:

- *HLQ.CNTL*
A partitioned data set with fixed block 80-byte records to contain customization jobs
- *HLQ.DATA*
A partitioned data set with variable-length data to contain other data contained in the customization definition

We used the WAS80.WPASND high-level qualifier.

Click **Next**.

6. Complete the information in the Configure Common Groups window, as shown in Figure 5-26.

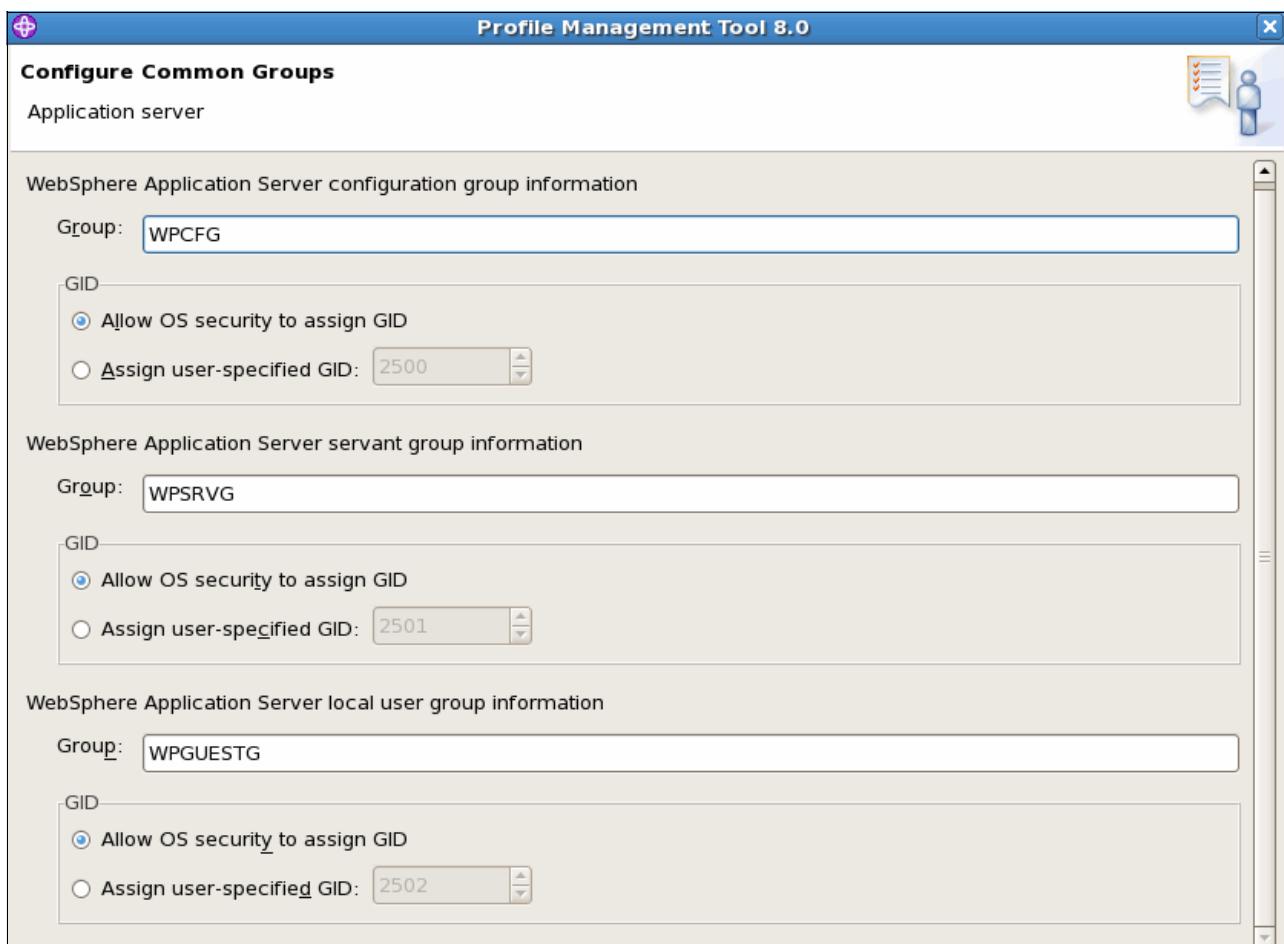


Figure 5-26 Configure Common Groups

Complete the following information:

- WebSphere Application Server configuration group information
 - Specify the default group name for the WebSphere Application Server administrator user ID and all server user IDs.
 - Select whether to allow the OS security system (RACF) to assign an unused GID value or to assign a specific GID.

- WebSphere Application Server servant group information
Specify the group name for all servant user IDs. You can use this group to assign subsystem permissions, such as DB2 authorizations, to all servants in the security domain.
Select whether to allow the OS security system (RACF) to assign an unused GID value, or assign a specific GID.
- WebSphere Application Server local user group information
Specify the group name for local clients and unauthorized user IDs (provides minimal access to the cell).
Select whether to allow the OS security system (RACF) to assign an unused GID value, or assign a specific GID.

GID values: The specified GID is the UNIX System Services GID number for the WebSphere Application Server configuration group. GID values must be unique numeric values between 1 and 2,147,483,647.

Click **Next**.

7. In the Configure Common Users window, provide information about the asynchronous administration user ID, as shown in Figure 5-27.



Figure 5-27 Configure Common Users - Asynchronous administration user ID

Provide the following information:

- Common controller user ID

UIDs: UIDs must be unique numbers between 1 and 2,147,483,647 within the system.

Enter the user ID to be associated with all the control regions and the daemon. This user ID will also own all of the configuration file systems. If you are using a non-IBM security system, the user ID might have to match the procedure name. Refer to your security system's documentation.

Select whether to allow the OS security system (RACF) to assign an unused UID value, or assign a specific UID to be associated with the control region user ID.

- Common servant user ID

Enter the user ID to be associated with the servant and control adjunct regions.

Select whether to allow the OS security system (RACF) to assign an unused UID value, or assign a specific UID to be associated with the servant region user ID.

- WebSphere Application Server administrator

Enter the user ID for the initial WebSphere Application Server administrator. The user ID must have the WebSphere Application Server configuration group as its default UNIX System Services group.

Select whether to allow the OS security system (RACF) to assign an unused UID value, or assign a specific UID to be associated with the administrator user ID.

- Asynchronous administration user ID

Enter the user ID to be used to run the asynchronous administration operations procedure. This ID must be a member of the WebSphere Application Server configuration group.

Select whether to allow the OS security system (RACF) to assign an unused UID value, or assign a specific UID for the asynchronous administration task user ID.

- WebSphere Application Server user ID home directory

This field identifies a new or existing file system directory in which home directories for WebSphere Application Server for z/OS user IDs will be created by the customization process. This directory does not need to be shared among z/OS systems in a WebSphere Application Server cell.

We used the following values:

- Common controller user ID: WPACRU
- UID: Allow OS security to assign UID
- Common servant user ID: WPASRU
- UID: Allow OS security to assign UID
- WebSphere Application Server administrator: WPADMIN
- UID: Allow OS security to assign UID
- Asynchronous administration user ID: WPADMSH
- UID: Allow OS security to assign UID
- WebSphere Application Server user ID home directory: /var/WebSphere/home

Click **Next**.

8. Provide the system and data set names to be used (See Figure 5-13 on page 173):
 - Specify the system and sysplex name for the target z/OS system on which you will configure WebSphere Application Server for z/OS.
 - Enter the name of an existing procedure library where the WebSphere Application Server for z/OS cataloged procedures are added.

We used the following values:

- System name: SC58
- Sysplex name: PLEX58
- PROCLIB data set name: SYS1.PROCLIB

Click **Next**.

9. Next, in the Cell, Node and Server names window, provide the necessary information, as shown in Figure 5-28.

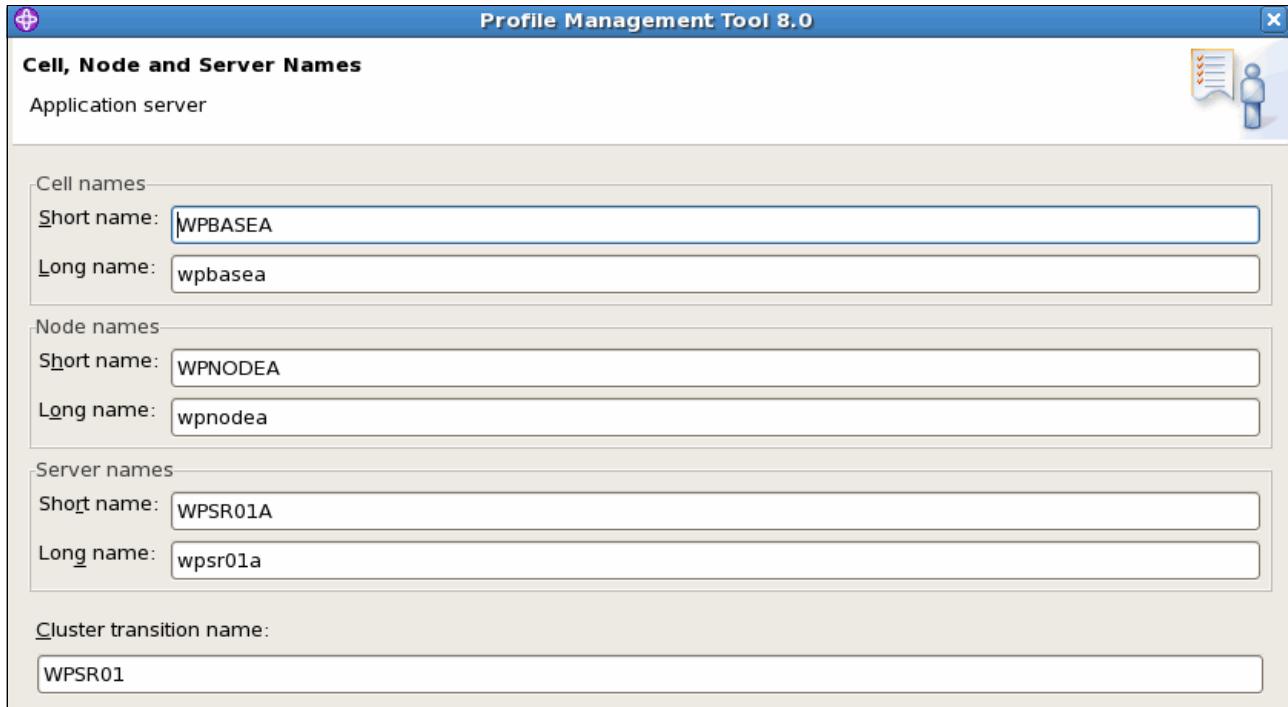


Figure 5-28 Cell, Node and Server Names

Provide the following information:

- Specify the long and short names for the cell, node, and servers. Short names identify the process to z/OS facilities, such as SAF. Long names are used as the primary external identification for the process. This is the name you will see in the administrative console.
- Cluster transition name

If this server is converted into a clustered server, this name becomes the cluster short name. The cluster short name is the WLM APPLENV name for all servers that are part of the same cluster.

Click **Next**.

10. Enter the Configuration File System values, as shown in Figure 5-16 on page 176.

- Mount point

Application server configuration file system mount point: Specifies the Read/write file system directory where the application data and environment files are written. This field is not writable here, but was specified earlier on the “System Environment: Configuration file system information” window.

- Directory path name relative to mount point

The relative path name of the directory within the configuration file system in which the application server configuration resides.

- Data set name

The file system data set that you will create and mount at the specified mount point.

- File system type
Select to allocate and mount your configuration file system data set using HFS or zFS.
- Volume, or '*' for SMS
Specify either the DASD volume serial number to contain the above data set or "*" to let SMS select a volume. Using "*" requires that SMS automatic class selection (ACS) routines be in place to select the volume. If you do not have SMS set up to handle data set allocation automatically, list the volume explicitly.
- Primary allocation in cylinders
The initial size allocation for the configuration file system data set. In the application server, the total space needed for this data set increases with the size and number of the installed applications. The minimum suggested size is 250 cylinders (3390).
- Secondary allocation in cylinders
The size of each secondary extent. The minimum suggested size is 100 cylinders.

We used the following values:

- Mount point: /wasv8config/wpcell/wpnodea
- Directory path name relative to mount point: AppServer
- Data set name: OMVS.WAS80.WPCELL.WPNODEA.ZFS
- File system type: zSeries File System (zFS)
- Volume, or '*' for SMS: *
- Primary allocation in cylinders: 420
- Secondary allocation in cylinders: 100

Click **Next**.

11.Specify the information for the product file system (see Figure 5-16 on page 176).

- Specify the name of the directory where the product files for WebSphere Application Server for z/OS were stored during installation.
- Select the option to allow to set up an intermediate symbolic link and specify the path name.

We used the following values:

- Product file system directory: /opt/zWebSphere/V8R0
- Intermediate symbolic link: Create intermediate symbolic link
- Path name of intermediate symbolic link: /wasv8config/wpcell/wpnodea/wasInstall

Click **Next**.

12.In the next window, select the applications to deploy to the environment that you are creating:

- Administrative console (best practice)
- Default application

We enabled both options. Click **Next**.

13. Enter the process information into the window shown in Figure 5-29. The job names for the processes are provided in the window and cannot be changed. Specify the procedure name for each process.

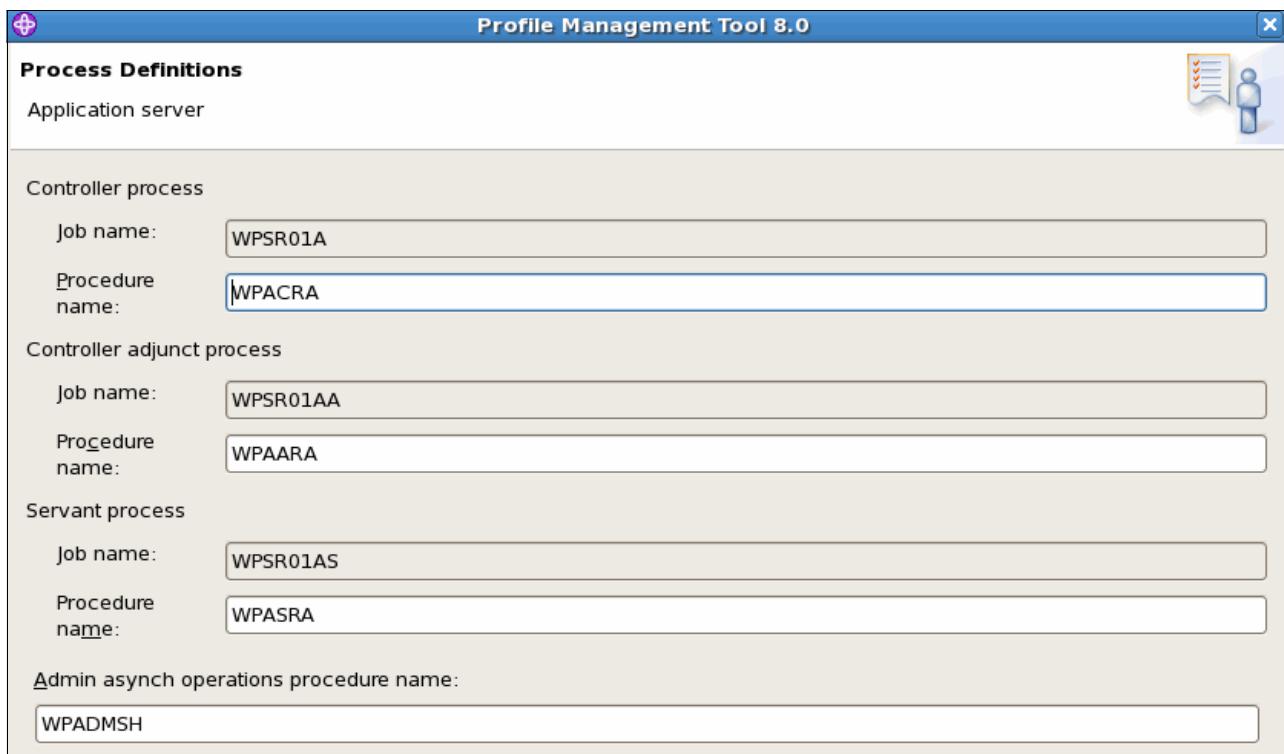


Figure 5-29 Application server names

Enter the following information:

- Controller process job and procedure name

The job name for the control region is the same as the server short name. This is the name used in the MVS START command to start the region.

- Controller adjunct process job and procedure name

The job name is used by WLM to start the control region adjunct. This is set to the server short name followed by the letter "A."

- Servant process job and procedure name

The job name is used by WLM to start the servant regions. This is set to the server short name followed by the letter "S."

- Admin asynch operations procedure name

Specify the JCL procedure name of a started task to be launched by way of the START command by node agents or application servers to perform certain asynchronous administrative operations (such as node synchronization) and add and remove a node.

Click **Next**.

14. Specify the application server port values in the window shown in Figure 5-30.

Good planning is important to avoid port conflicts. Ensure that you have all the values that you need to complete the information in this window.

Click **Next**.

Node host name or IP address:	wtsc58.itso.ibm.com
JMX SOAP connector port:	12062
ORB listener IP address:	*
ORB port:	12063
ORB SSL port:	12064
HTTP transport IP address (2):	*
Administrative console port:	12065
Administrative console secure port:	12066
HTTP transport port:	12067
HTTPS transport port:	12068
Administrative interprocess communication port (X):	12069
High Availability Manager Communication Port (DCS):	12070
Service Integration Port:	12071
Service Integration Secure Port:	12072
Service Integration MQ Interoperability Port:	12073
Service Integration MQ Interoperability Secure Port:	12074
Session Initiation Protocol (SIP) Port:	12075
Session Initiation Protocol (SIP) Secure Port:	12076

Figure 5-30 Application server port values

15. Enter the following information about the Location Service Daemon Definitions, as shown in Figure 5-19 on page 179:

- Daemon home directory

Directory in which the location service daemon resides. This is set to the configuration file system mount point / daemon and cannot be changed.

- Daemon job name

Specifies the job name of the location service daemon, specified in the JOBNAMES parameter of the MVS start command used to start the location service daemon.

- Procedure name

Name of the member in your procedure library to start the location service daemon.

- IP Name

The fully qualified IP name, registered with the Domain Name Server (DNS), that the location service daemon uses.

- Listen IP
Address at which the daemon listens.
- Port
Port number on which the location service daemon listens.
- SSL port
Port number on which the location service daemon listens for SSL connections.
- Register daemon with WLM DNS
If you use the WLM DNS (connection optimization), you must select this option to register your location service daemon with it; otherwise, do not select it.

We used the following values:

- Daemon home directory: /wasv8config/wpcel1/wpnodea/Daemon
- Daemon job name: WPDEMNA
- Procedure name: WPDEMNA
- IP name: wtsc58.itso.ibm.com
- Listen IP address: *
- Port: 12060
- SSL Port: 12061
- Register daemon with WLM DNS: not enabled

Click **Next**.

16. Enter the information required for SSL connections.

We used the following values:

- Certificate authority keylabel: WebSphereCA
- Generate certificate authority (CA) certificate: enabled
- Expiration date for certificates: 2021/12/31
- Default SAF keyring name: WASKeyring.WPCELL
- Use virtual keyring for z/OS SSL clients: Not enabled
- Enable SSL on location service daemon: Not enabled

Click **Next**.

17. Select the user registry that will be used to manage user identities and authorization policy.

Tip: If you plan to federate this application server, set the application server's SAF profile prefix to be the same as that of the Network Deployment Cell.

Select from one of the following choices:

- z/OS security product

This option uses the z/OS system's SAF compliant security product, such as IBM RACF or equivalent, to manage WebSphere Application Server identities and authorization according to the following rules:

- The SAF security database is used as the WebSphere user repository.
- SAF EJBROLE profiles will be used to control role-based authorization, including administrative authority.
- Digital certificates will be stored in the SAF security database.

Important: Select the z/OS security product option if you are planning to use the SAF security database as your WebSphere Application Server registry or if you plan to set up an LDAP or custom user registry whose identities will be mapped to SAF user IDs for authorization checking. For this security option, you must decide whether to set a security domain name, and choose an administrator user ID and an unauthenticated (guest) user ID.

- WebSphere Application Server security

The WebSphere Application Server administrative security option is used to manage the Application Server identities and authorization as follows:

- A simple file-based user registry will be built as part of the customization process.
- Application-specific role binds will be used to control role-based authorization.
- The WebSphere Application Server console users and groups list will control administrative authority.
- Digital certificates will be stored in the configuration file system as keystores.

Tip: Choose this option if you plan to use an LDAP or custom user registry without mapping to SAF user IDs. (The file-based user registry is not recommended for production use.)

- No security

Although it is not a best practice, you can disable administrative security. If you choose this security option, there are no other choices to make. Your WebSphere Application Server environment will not be secured until you configure and enable security manually. You can enable security manually later using the administrative console or using Jython scripts.

We select the **Use z/OS security product** option.

Click **Next**.

18. In the next window, chose one of the following options:

- SAF profile prefix (formally known as *Security domain identifier*)

This optional parameter is used to distinguish between APPL or EJBROLE profiles based on security domain name. It provides an alphanumeric security domain name of one to eight characters. Internally, this sets SecurityDomainType to the string cellQualified.

All servers in the cell will prepend the security domain name you specify to the application-specific J2EE role name to create the SAF EJBROLE profile for checking. The security domain name is not used, however, if role checking is performed using WebSphere Application Server for z/OS bindings.

The security domain name is also used as the APPL profile name and inserted into the profile name used for CBIND checks. The RACF jobs that the Customization Dialog generates create and authorize the appropriate RACF profiles for the created nodes and servers.

If you do not want to use a security domain identifier, leave this field blank.

- WebSphere Application Server unauthenticated user ID

Associated with unauthenticated client requests. It is sometimes referred to as the “guest” user ID. It should be given the RESTRICTED attribute in RACF, to prevent it from inheriting UACC-based access privileges. The UNIX System Services UID number for the user ID is specified here and is associated with unauthenticated client requests. The UID value must be unique numeric values between 1 and 2,147,483,647.

- Enable Writable SAF Keyring support

This feature allows administrative console to create and sign certificates by a CA, connect to keyrings, remove from keyrings, and import, export, and renew.

All certificates created with the writable keyring support are generated and signed by Java code and not by SAF. In this case, the writable keyring support only uses SAF to store the generated certificates.

The writable keyring support is completely optional. New keystores and truststores marked as read-only can be created independently from the writable keyring support. When using the read-only JCERACFKS and JCECCARACFS keystores, the certificates in the appropriate SAF keyring can still be viewed in the administrative console.

We used the following values:

- SAF profile prefix (optional): WPCELL
- WebSphere Application Server unauthenticated user ID: WPGUEST
- UID: Allow OS security to assign UID
- Enable Writable SAF Keyring support: enabled

Click **Next**.

19. The next window allows you to create web server definitions; however, they are not needed at this time. Click **Next**.
20. The next window allows you to tailor the JCL for the customization jobs. Enter a valid job statement for your installation on this window. The profile creation process will update the job name for you in all the generated jobs, so you need not be concerned with that portion of the job statement. If continuation lines are needed, replace the comment lines with continuation lines. Click **Next**.
21. The Customization Summary is the final window. Click **Create** to store this application server environment definition for later transfer to the intended z/OS host system.
22. Click **Finish** when the jobs are complete.

If successful, the next step in the z/OS customization process is to upload these jobs and the associated instructions to a pair of z/OS partitioned data sets:

1. On the main window, select the customization definition for the profile and click **Process**. To upload the generated jobs to the target z/OS system, select from the following options:
 - Upload to target z/OS system using FTP
 - Upload to target z/OS system using FTP over SSL
 - Upload to target z/OS system using secure FTP
 - Export to local file system
 Click **Next**.
2. If you choose to upload the customization using FTP, in the upload customization definition window, enter the target z/OS system. This path name must be fully qualified or the upload will fail. You must also specify the user ID and password and FTP server port.

Select the **Allocate target z/OS data sets** option to specify whether to allocate the data sets if they do not exist. If the data sets exist and are to be reused, clear this option.

Click **Finish**, and a progress bar displays while the upload is occurring.

3. After the customization profile is uploaded, select the customization definition in the Profile Management Tool, and click the **Customization Instructions** tab. This tab provides complete instructions about how to build the profile using the jobs.

These instructions can help you determine the jobs to run, the order to run them in, and the expected results. It also explains how to start the environment after you are finished.

After the jobs run successfully, the application server profile is complete.

5.3.3 Federating an application server

This definition is used to federate the base application server into the previously created deployment manager node. To begin, run the Profile Management tool to create the customization definition:

1. Click **Create** on the Profile Management Tool main window.

2. Click **Federate an application server** in the Environment Selection window, as shown in Figure 5-31. Click **Next**.

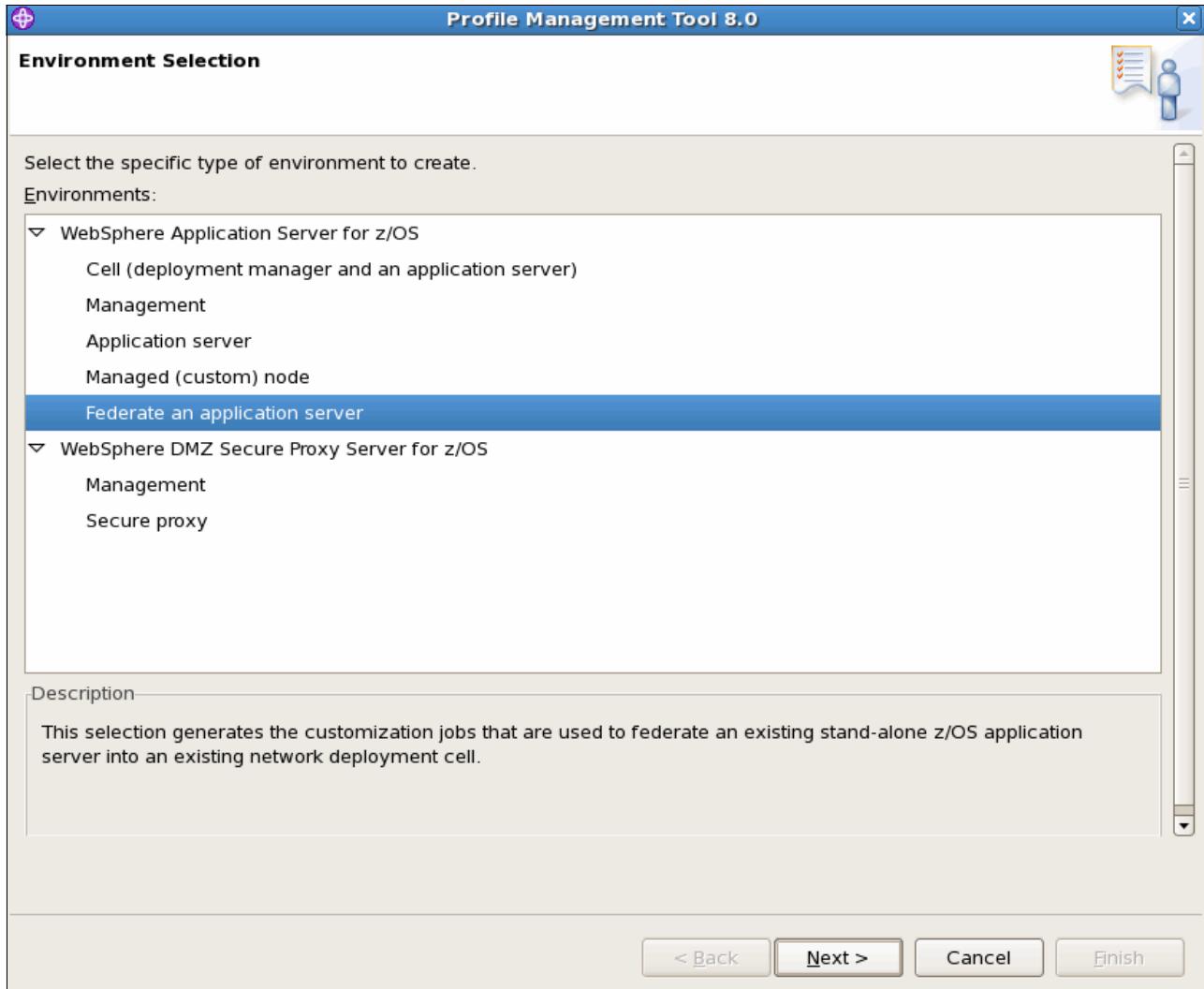


Figure 5-31 Federate an application server selection

3. Specify a name for the customization definition and, optionally, a response file path. We used ZFederate01 as the name. Click **Next**.

4. In the Default Values window, specify a default range of ports to be used for the node agent, as shown in Figure 5-32. When this option is not selected, each port value defaults to an IBM-provided number. Click **Next**.

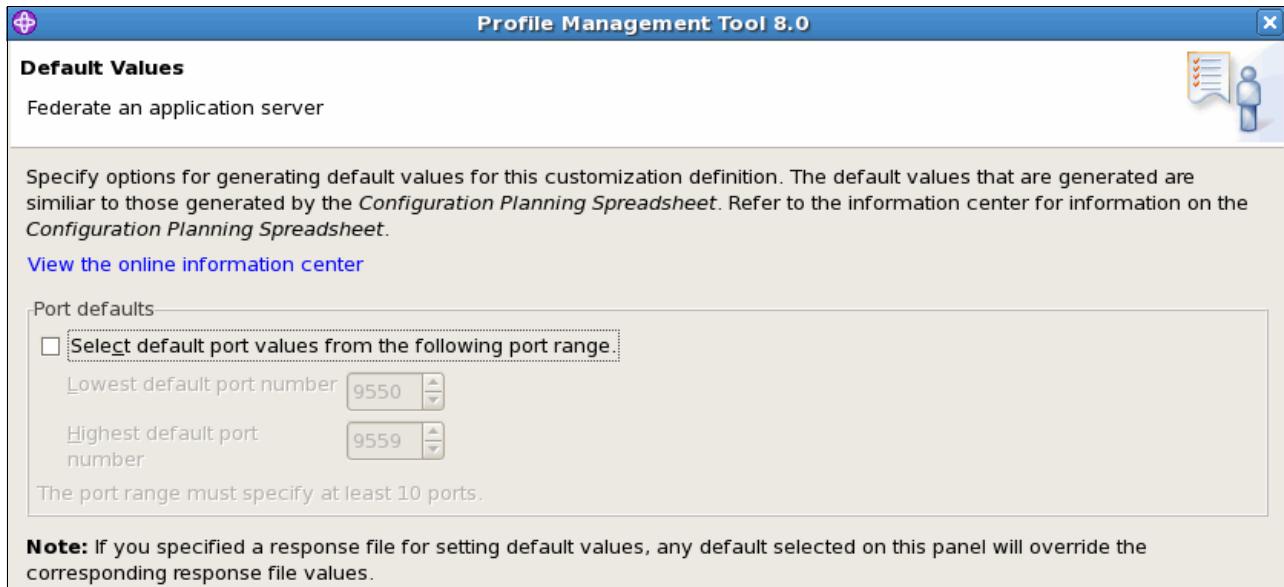


Figure 5-32 Default port settings

5. Next, specify the high-level qualifier for the target z/OS data sets that will contain the generated jobs and instructions that are created (see Figure 5-10 on page 170).

The generated batch jobs and instructions are uploaded to the following z/OS partitioned data sets:

- *HLQ.CNTL*
Partitioned data set with fixed block 80-byte records to contain customization jobs
- *HLQ.DATA*
Partitioned data set with variable-length data to contain other data contained in the customization definition (Figure 5-33 on page 202)

Tip: You can specify a multi-level high-level qualifier as the data set high-level qualifier.

- We used the *WAS80.WPFED* high-level qualifier.

Click **Next**.

- In the Federate Application Server (Part 1) window, specify the information that is needed to access the application server and the deployment manager, as shown in Figure 5-33.

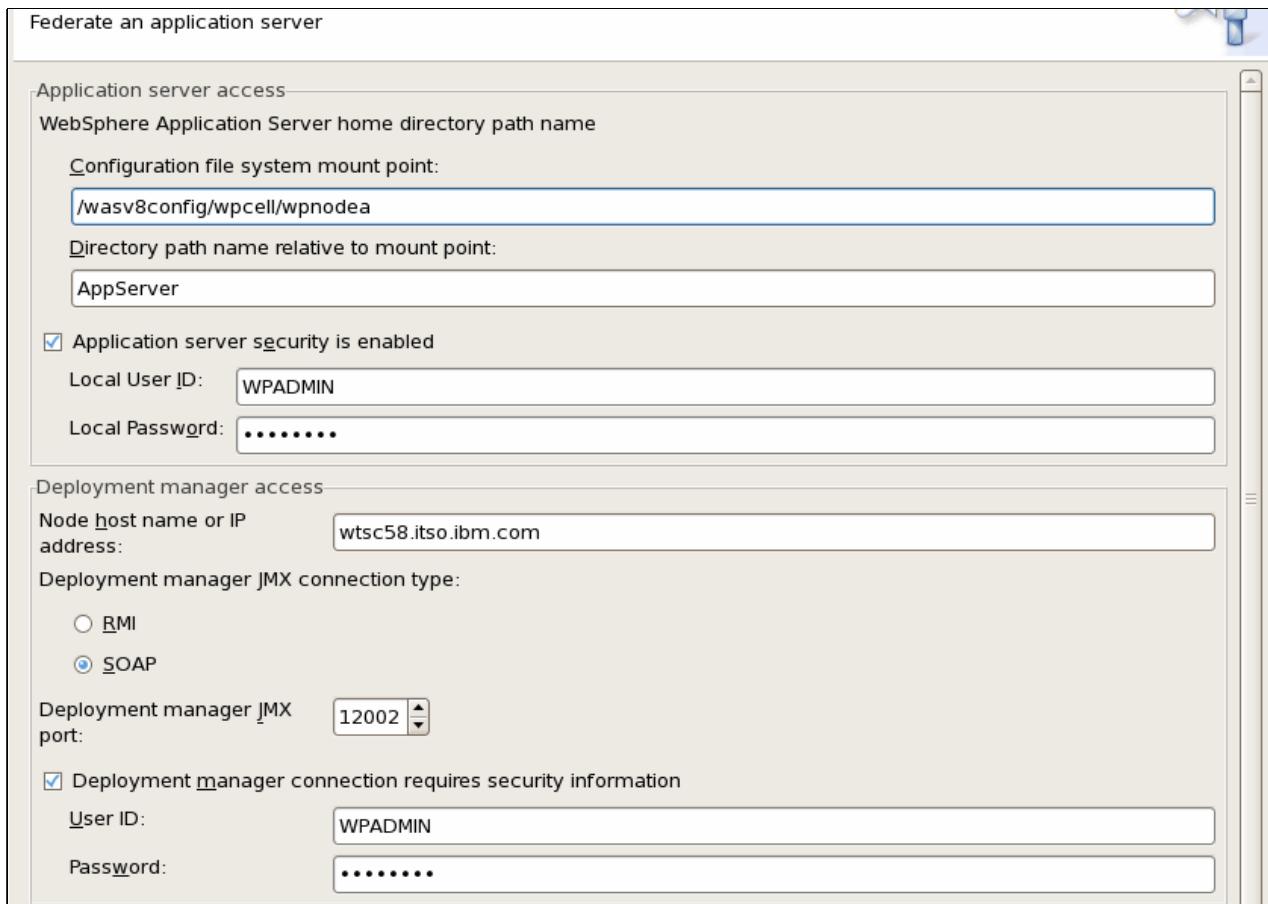


Figure 5-33 Specify Application server settings (Part 1)

Enter the following information:

- Application server access information

This section contains the information needed to find and access the application server node configuration file system. This information includes the file system mount point, the directory path name, and, if administrative security is enabled on the application server, the administrator user ID and password.

- Deployment manager access information

Federating the application server requires that the deployment manager be running and accessible by the federation process. This section provides the information required to connect to the deployment manager, including the host name or IP address, JMX connection type and port, and the administrator user ID and password.

The node host name must always resolve to an IP stack on the system where the deployment manager runs. The node host name cannot be a DVIPA or a DNS name that, in any other way, causes the direction of requests to more than one system.

The user ID and password are required when global security is enabled on the Network Deployment cell unless an RMI connector is being used. If an RMI connector is being used, the identity information will be extracted from the thread of execution of the addNode job if the user ID and password are not specified.

Click **Next**.

7. Configure the node agent that is required for the application server to participate in the distributed environment cell, as shown in Figure 5-34.

The screenshot shows the 'Node Agent Definitions' window with the following configuration details:

- Server name (short):** WPAGNTA
- Server name (long):** nodeagent
- JMX SOAP connector port:** 12022
- ORB listener IP address:** *
- ORB port (5):** 12023
- ORB SSL port (6):** 12024
- Node Discovery port (3):** 12037
- Node Multicast Discovery port (8):** 12039
- Node IPv6 Multicast Discovery port (9):** 12034
- Administrative local port (2):** 12029
- High availability manager communication port (DCS) (7):** 12030
- Application server's new ORB port:** 12063
- Node group name:** DefaultNodeGroup
- Configuration group name:** WPCFG
- Configuration user ID:** WSADMIN

At the bottom, there are three checked checkboxes:

- Include Apps
- Launch the node agent after node federation
- Federate service integration busses that exist on this node

Figure 5-34 Specify Application server settings (Part 2)

Enter the following information:

- Specify the short name for the node agent process.

The short name is the server's job name, as specified in the MVS START command JOBNAME parameter. (The node agent server long name is set to the fixed value of nodeagent.)

- Specify the IP addresses and ports to be used by the node agent and a new ORB port to be used by the application server.
- The node group, configuration group, and configuration user ID.
- Select the relevant federation options for your environment. If you have installed applications on the application server or defined service integration buses, you can choose to have those included in the newly federated application server.

Click **Next**.

8. The next window allows you to tailor the JCL for the customization jobs. Enter a valid job statement for your installation on this window. The profile creation process will update the job name for you in all the generated jobs, so you need not be concerned with that portion of the job statement. If continuation lines are needed, replace the comment lines with continuation lines. Click **Next**.

9. The Customization Summary window is the final window. Click **Create**, and then click **Finish**.

5.3.4 Uploading jobs and associated instructions

If successful, the next step in the z/OS customization process is to upload these jobs and the associated instructions to a pair of z/OS partitioned data sets:

1. On the main window, select the customization definition for the profile and click **Process**. To upload the generated jobs to the target z/OS system, select from the following options:

- Upload to target z/OS system using FTP
- Upload to target z/OS system using FTP over SSL
- Upload to target z/OS system using secure FTP
- Export to local file system

Click **Next**.

2. If you choose to upload the customization using FTP, in the upload customization definition window, enter the target z/OS system. This path must be fully qualified or the upload will fail. You must also specify the user ID and password and FTP server port.

Select the **Allocate target z/OS data sets** option to specify whether to allocate the data sets if they do not exist. If the data sets exist and are to be reused, clear this option.

Click **Finish**, and a progress bar displays while the upload is occurring.

3. After the customization profile is uploaded, select the customization definition in the Profile Management Tool and click the **Customization Instructions** tab. This tab provides complete instructions about how to build the profile using the jobs.

These instructions help you determine the jobs to run, the order in which to run them, and the expected results. It also explains how to start the environment after you are finished.

After the jobs run successfully, the application server profile is complete.

Attention: The user that will run job BBOWADDN must have READ access on profiles IRR.DIGTCERT.LIST and IRR.DIGTCERT.LISTRING.

5.4 Creating a job manager profile

In a flexible management environment, the job manager allows you to asynchronously submit and administer jobs for large numbers of unfederated application servers and deployment managers over a geographically dispersed area. Many of the management tasks that you can perform with the job manager are tasks that you can already perform with the product, such as application management, server management, and node management. However, with the job manager, you can aggregate the tasks and perform the tasks across multiple application servers or deployment managers.

5.4.1 Creating the customization definition

To begin, run the profile management tool to create the customization definition:

1. Click **Create** on the Profile Management Tool main window.
2. To generate the definitions for the job manager, select **Management** in the Profile Management tool environment selection window and click **Next**.

3. In the next window, select **Job manager** as the type and click **Next**.

Specify a name for the customization definition, and click **Next**. We specified a customization definition name of *ZJManager01*.

4. The next window allows you to specify defaults for GID and UID values, name and user ID defaults based on a two character prefix that identifies the cell, and allows you to specify a default range for ports assigned to the process. Click **Next**.
5. You are now prompted to specify the high-level qualifier for the target z/OS data sets that contain the generated jobs and instructions that are created (refer to Figure 5-10 on page 170).

The generated batch jobs and instructions are uploaded to the following z/OS partitioned data sets:

- *HLQ.CNTL*

Partitioned data set with fixed block 80-byte records to contain customization jobs

- *HLQ.DATA*

Partitioned data set with variable-length data to contain other data contained in the customization definition

Tip: You can specify a multi-level high-level qualifier as the data set high-level qualifier.

We used WAS80.WPJGMGR as the high-level qualifier.

Click **Next**.

6. In the Configure Common Groups window, enter the following information:

- WebSphere Application Server configuration group information
 - Specify the default group name for the WebSphere Application Server administrator user ID and all server user IDs.
 - Select whether to allow the OS security system (RACF) to assign an unused GID value, or assign a specific GID.
- WebSphere Application Server servant group information
 - Specify the group name for all servant user IDs. You can use this group to assign subsystem permissions, such as DB2 authorizations, to all servants in the security domain.
 - Select whether to allow the OS security system (RACF) to assign an unused GID value, or assign a specific GID.
- WebSphere Application Server local user group information
 - Specify the group name for local clients and unauthorized user IDs (provides minimal access to the cell).
 - Select whether to allow the OS security system (RACF) to assign an unused GID value, or assign a specific GID.

GID values: The specified GID is the UNIX System Services GID number for the WebSphere Application Server configuration group. GID values must be unique numeric values between 1 and 2,147,483,647.

We used the following values:

- WebSphere Application Server configuration group information: WSCFG1
- GID: Allow OS security to assign GID
- WebSphere Application Server servant group information: WSSR1
- GID: Allow OS security to assign GID
- WebSphere Application Server local user group information: WSCLGP
- GID: Allow OS security to assign GID

Click **Next**.

7. Next, configure the user ID settings:

- Common controller user ID

Defines the user ID that is associated with all the control regions and the daemon. This user ID also owns the configuration file systems.

- Common servant user ID

Defines the user ID that is associated with the servant regions.

- WebSphere Application Server administrator user ID

Specifies the initial WebSphere Application Server administrator. The ID must have the WebSphere Application Server configuration group as its default UNIX System Services group. The UNIX System Services UID number for the administrator user ID is specified here, and must be a unique numeric value between 1 and 2,147,483,647.

We used the following values:

- Common controller user ID: WJACR
- UID: Allow OS security to assign UID
- Common servant user ID: WJASR
- UID: Allow OS security to assign UID
- WebSphere Application Server administrator: WJADMIN
- UID: Allow OS security to assign UID
- WebSphere Application Server user ID home directory: /var/WebSphere/home

Click **Next**.

8. Provide the system and data set names to be used:

- Specify the system and sysplex name for the target z/OS system on which you will configure WebSphere Application Server for z/OS.
- Enter the name of an existing procedure library where the WebSphere Application Server for z/OS cataloged procedures are added.

We used the following values:

- System name: SC58
- Sysplex name: PLEX58
- PROCLIB data set name: SYS1.PROCLIB

Click **Next**.

9. In the Cell, Node and Server names window, enter the following information:
 - Specify the long and short names for the cell, node, and servers. Short names identify the process to z/OS facilities, such as SAF. Long names are used as the primary external identification for the process. This is the name you will see in the administrative console. (The job manager server long name is set to the fixed value of jobmgr.)

Tip: Assign each management server (administrative agent, deployment manager, or job manager) its own cell name that is different from that of any other WebSphere Application Server cell on the same z/OS sysplex.

- Cluster transition name

If this server is converted into a clustered server, this name becomes the cluster short name. The cluster short name is the WLM APPLENV name for all servers that are part of the same cluster.

We used the following values:

- Cell short name: WJCELL
- Cell long name: wjcell
- Node short name: WJNODEA
- Node long name: wjnodela
- Server short name: WJS01A
- Server long name: jobmgr
- Cluster transition name: WJC01

Click **Next**.

10. Enter the Configuration File System values:

- Mount point

Application server configuration file system mount point: Specifies the Read/write file system directory where the application data and environment files are written. This field is not writable here but was specified earlier on the “System Environment: Configuration file system information” window.

- Directory path name relative to mount point

The relative path name of the directory within the configuration file system in which the application server configuration resides.

- Data set name

The file system data set you will create and mount at the specified mount point above.

- File system type

Select to allocate and mount your configuration file system data set using HFS or zFS.

- Volume, or '*' for SMS:

Specify either the DASD volume serial number to contain the above data set or "*" to let SMS select a volume. Using "*" requires that SMS automatic class selection (ACS) routines be in place to select the volume. If you do not have SMS set up to handle data set allocation automatically, list the volume explicitly.

- Primary allocation in cylinders

The initial size allocation for the configuration file system data set. In the application server, the total space needed for this data set increases with the size and number of the installed applications. The minimum suggested size is 250 cylinders (3390).

- Secondary allocation in cylinders

The size of each secondary extent. The minimum suggested size is 100 cylinders.

We entered the following values:

- Mount point: /wasv8config/wjcell/wjnnodea
- Directory path name relative to mount point: JobManager
- Data set name: OMVS.WAS80.WJCELL.WJNODEA.ZFS
- File system type: zSeries File System (zFS)
- Volume, or “*” for SMS: *
- Primary allocation in cylinders: 420
- Secondary allocation in cylinders: 100

Click **Next**.

11.Specify the information for the product file system:

- Specify the name of the directory where the product files for WebSphere Application Server for z/OS were stored during installation.
- Select the option to allow to set up an intermediate symbolic link and specify the path name.

We used the following values:

- Product file system directory: /opt/zWebSphere/V8R0
- Intermediate symbolic link: Create intermediate symbolic link
- Path name of intermediate symbolic link: /wasv8config/wjcell/wjnnodea/wasInstall

Click **Next**.

12.Enter the process information. The job names for the processes are provided in the window and cannot be changed. Specify the procedure name for each process.

- Controller process job and procedure name

The job name for the control region is the same as the server short name. This is the name used in the MVS START command to start the region.

- Servant process job and procedure name

The job name is used by WLM to start the servant regions. This is set to the server short name followed by the letter “S.”

We used the following values:

- Controller process job name: WJS01A
- Controller process procedure name: WJACRA
- Servant process job name: WJS01AS
- Servant process procedure name: WJASRA

Click **Next**.

13. Specify the application server port values shown in Figure 5-35. Good planning is important to avoid port conflicts. Ensure that you have all values that you need to complete the information in this window.

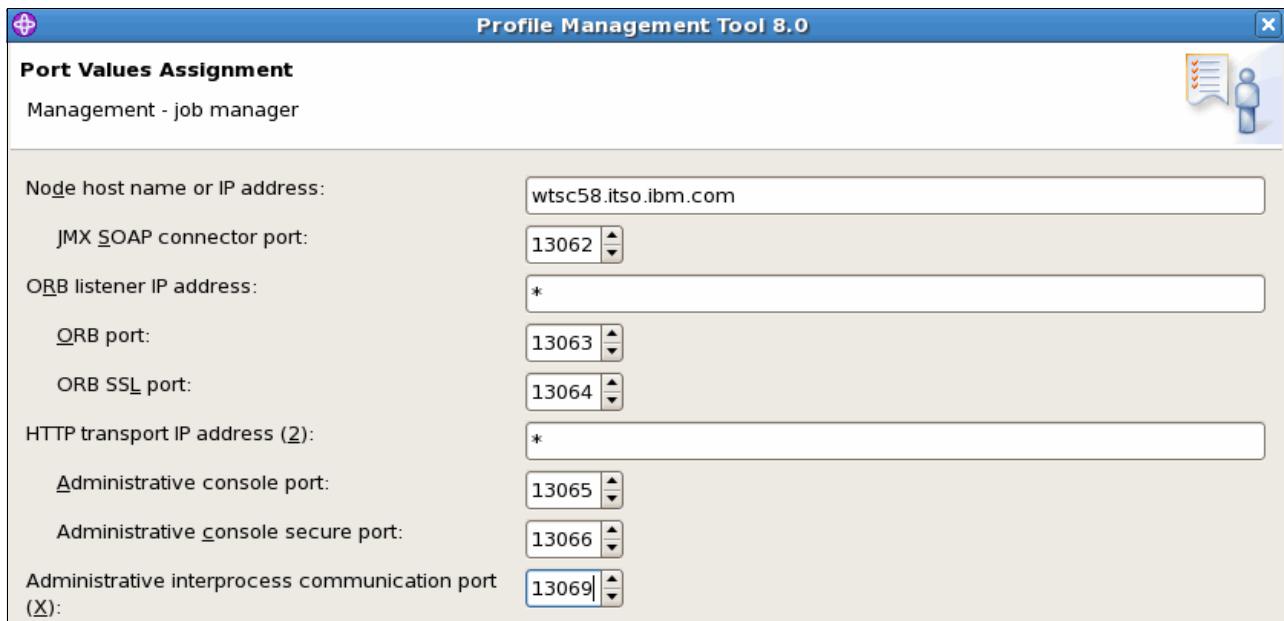


Figure 5-35 Job manager port values

14. Enter the Location Service Daemon Definitions:

- Daemon home directory

Specifies the directory in which the location service daemon resides. This directory is set to the configuration file system mount point / daemon and cannot be changed.

- Daemon job name

Specifies the job name of the location service daemon, specified in the JOBNAME parameter of the MVS start command used to start the location service daemon.

- Procedure name

Defines the name of the member in your procedure library to start the location service daemon.

- IP Name

Specifies the fully qualified IP name that is registered with the Domain Name Server (DNS), which the location service daemon uses.

- Listen IP

Defines that address at which the daemon listens.

- Port

Specifies the port number on which the location service daemon listens.

- SSL port

Specifies the port number on which the location service daemon listens for SSL connections.

- Register daemon with WLM DNS

If you use the WLM DNS (connection optimization), select this option to register your location service daemon with it. Otherwise, do not select it.

We used the following values:

- Daemon home directory: /wasv8config/wjcell/wjnnodea/Daemon
- Daemon job name: WJDEMNA
- Procedure name: WJDEMNA
- IP name: wtsc58.itso.ibm.com
- Listen IP address: *
- Port: 13060
- SSL Port: 13061
- Register daemon with WLM DNS: not enabled

Click **Next**.

15. Enter the information that is required for SSL connections.

We used the following values:

- Certificate authority keylabel: WebSphereCAJM
- Generate certificate authority (CA) certificate: Enabled
- Expiration date for certificates: 2021/12/31
- Default SAF keyring name: WASKeyring.WJCELL
- Use virtual keyring for z/OS SSL clients: Not enabled
- Enable SSL on location service daemon: Not enabled

Click **Next**.

16. Select the user registry that will be used to manage user identities and authorization policy from one of the following choices:

- z/OS security product

This option uses the z/OS system's SAF compliant security product, such as IBM RACF or equivalent, to manage WebSphere Application Server identities and authorization according to the following rules:

- The SAF security database will be used as the WebSphere user repository.
- SAF EJBROLE profiles are used to control role-based authorization, including administrative authority.
- Digital certificates are stored in the SAF security database.

Important: Select the z/OS security product option if you are planning to use the SAF security database as your WebSphere Application Server registry or if you plan to set up an LDAP or custom user registry whose identities will be mapped to SAF user IDs for authorization checking. For this security option, you must decide whether to set a security domain name, and choose an administrator user ID and an unauthenticated (guest) user ID.

- WebSphere Application Server security

The WebSphere Application Server administrative security option is used to manage the Application Server identities and authorization as follows:

- A simple file-based user registry will be built as part of the customization process.
- Application-specific role binds will be used to control role-based authorization.
- The WebSphere Application Server console users and groups list will control administrative authority.
- Digital certificates will be stored in the configuration file system as keystores.

Tip: Choose this option if you plan to use an LDAP or custom user registry without mapping to SAF user IDs. (The file-based user registry is not a best practice for production use.)

- No security

Although it is not a best practice, you can disable administrative security. If you choose this security option, there are no other choices to make. Your WebSphere Application Server environment will not be secured until you configure and enable security manually. You can enable security manually later using the administrative console or using Jython scripts.

We used the “Use z/OS security product” option.

Click **Next**.

17. In the next window, chose one of the following options:

- SAF profile prefix (formally known as *Security domain identifier*)

This optional parameter is used to distinguish between APPL or EJBROLE profiles based on security domain name. It provides an alphanumeric security domain name of one to eight characters. Internally, this sets SecurityDomainType to the string cellQualified.

All servers in the cell will prepend the security domain name you specify to the application-specific J2EE role name to create the SAF EJBROLE profile for checking. The security domain name is not used, however, if role checking is performed using WebSphere Application Server for z/OS bindings.

The security domain name is also used as the APPL profile name and inserted into the profile name used for CBIND checks. The RACF jobs that the Customization Dialog generates create and authorize the appropriate RACF profiles for the created nodes and servers.

If you do not want to use a security domain identifier, leave this field blank.

- WebSphere Application Server unauthenticated user ID

Associated with unauthenticated client requests. It is sometimes referred to as the “guest” user ID. It should be given the RESTRICTED attribute in RACF, to prevent it from inheriting UACC-based access privileges. The UNIX System Services UID number for the user ID is specified here and is associated with unauthenticated client requests. The UID value must be unique numeric values between 1 and 2,147,483,647.

- Enable Writable SAF Keyring support

This feature allows administrative console to create and sign certificates by a CA, connect to keyrings, remove from keyrings, and import, export, and renew.

All certificates created with the writable keyring support are generated and signed by Java code and not by SAF. In this case, the writable keyring support only uses SAF to store the generated certificates.

The writable keyring support is completely optional. New keystores and truststores marked as read-only can be created independently from the writable keyring support. When using the read-only JCERACFKS and JCECCARACFS keystores, the certificates in the appropriate SAF keyring can still be viewed in the administrative console.

We used the following values:

- SAF profile prefix (optional): WJ
- WebSphere Application Server unauthenticated user ID: WJGUEST
- UID: Allow OS security to assign UID
- Enable Writable SAF Keyring support: Enabled

Click **Next**.

18. Next, you can tailor the JCL for the customization jobs. Enter a valid job statement for your installation on this window. The profile creation process will update the job name for you in all the generated jobs, so you need not be concerned with that portion of the job statement. If continuation lines are needed, replace the comment lines with continuation lines. Click **Next**.

19. In the Customization Summary, click **Create** to store this application server environment definition for later transfer to the intended z/OS host system.

20. Click **Finish** when the jobs are complete.

5.4.2 Uploading the jobs and the associated instructions

Next, upload these jobs and the associated instructions to a pair of z/OS partitioned data sets:

1. On the main window, select the customization definition for the profile and click **Process**. To upload the generated jobs to the target z/OS system, select from the following options:
 - Upload to target z/OS system using FTP
 - Upload to target z/OS system using FTP over SSL
 - Upload to target z/OS system using secure FTP
 - Export to local file system

Click **Next**.

2. If you choose to upload the customization using FTP, in the upload customization definition window, enter the target z/OS system. This path name must be fully qualified or the upload will fail. You must also specify the user ID and password and FTP server port.

Select **Allocate target z/OS data sets** to specify whether to allocate the data sets if they do not exist. If the data sets exist and are to be reused, clear this option.

Click **Finish**, and a progress bar displays while the upload is occurring.

3. After the customization profile is uploaded, select the customization definition in the Profile Management Tool, and click the **Customization Instructions** tab. This tab provides complete instructions about how to build the profile using the jobs.

These instructions help you determine the jobs to run, the order in which to run them, and the expected results. It also explains how to start the environment after you are finished.

After the jobs run successfully, the application server profile is complete.

5.5 Creating an administrative agent profile

The administrative agent provides a single interface in which to administer multiple application server nodes, such as development, unit test, or server farm environments. Using a single interface to administer your application servers reduces the impact of running administrative services in every application server.

5.5.1 Creating the customization definition

To begin, run the profile management tool to create the customization definition:

1. Click **Create** on the Profile Management Tool main window.
2. To generate the definitions for the Job Manager, select **Management** in the Profile Management tool environment selection window and click **Next**.
3. In the next window, select **Administrative agent** as the type, and click **Next**.
Specify a name for the customization definition, and click **Next**. We defined a customization definition name of ZAdminAg01.
4. Specify defaults for GID and UID values, name, and user ID defaults based on a two character prefix that identifies the cell, and allows you to specify a default range for ports assigned to the process. Click **Next**.
5. Specify the high-level qualifier for the target z/OS data sets that will contain the generated jobs and instructions that are created.

The generated batch jobs and instructions are uploaded to the following z/OS partitioned data sets:

- *HLQ.CNTL*

Partitioned data set with fixed block 80-byte records to contain customization jobs

- *HLQ.DATA*

Partitioned data set with variable-length data to contain other data contained in the customization definition

Tip: You can specify a multi-level high-level qualifier as the data set high-level qualifier.

We used WAS80.WPADMAG as the high-level qualifier.

Click **Next**.

6. In the Configure Common Groups window, enter the following information:
 - WebSphere Application Server configuration group information
 - Specify the default group name for the WebSphere Application Server administrator user ID and all server user IDs.
 - Select whether to allow the OS security system (RACF) to assign an unused GID value, or assign a specific GID.
 - WebSphere Application Server servant group information
 - Specify the group name for all servant user IDs. You can use this group to assign subsystem permissions, such as DB2 authorizations, to all servants in the security domain.
 - Select whether to allow the OS security system (RACF) to assign an unused GID value, or assign a specific GID.
 - WebSphere Application Server local user group information
 - Specify the group name for local clients and unauthorized user IDs (provides minimal access to the cell).
 - Select whether to allow the OS security system (RACF) to assign an unused GID value, or assign a specific GID.

GID values: The specified GID is the UNIX System Services GID number for the WebSphere Application Server configuration group. GID values must be unique numeric values between 1 and 2,147,483,647.

We used the following values:

- WebSphere Application Server configuration group information: WACFG
- GID: Allow OS security to assign GID
- WebSphere Application Server servant group information: WASRG
- GID: Allow OS security to assign GID
- WebSphere Application Server local user group information: WAGUESTG
- GID: Allow OS security to assign GID

Click **Next**.

7. Next, configure the user ID settings:

- Common controller user ID
 - Specifies the user ID that is associated with all the control regions and the daemon. This user ID also owns the configuration file systems.
- Common servant user ID
 - Specifies the user ID that is associated with the servant regions.
- WebSphere Application Server administrator user ID
 - Defines the initial WebSphere Application Server administrator. The ID must have the WebSphere Application Server configuration group as its default UNIX System Services group. The UNIX System Services UID number for the administrator user ID is specified here, and must be a unique numeric value between 1 and 2,147,483,647.

We used the following values:

- Common controller user ID: WAACR
- UID: Allow OS security to assign UID
- Common servant user ID: WAASR
- UID: Allow OS security to assign UID
- WebSphere Application Server administrator: WAADMIN
- UID: Allow OS security to assign UID
- WebSphere Application Server user ID home directory: /var/WebSphere/home

Click **Next**.

8. Provide the system and data set names to be used:

- Specify the system and sysplex name for the target z/OS system on which you will configure WebSphere Application Server for z/OS.
- Enter the name of an existing procedure library where the WebSphere Application Server for z/OS cataloged procedures are added.

We used the following values:

- System name: SC58
- Sysplex name: PLEX58
- PROCLIB data set name: SYS1.PROCLIB

Click **Next**.

9. In the Cell, Node and Server names window, enter the following information:
- Specify the long and short names for the cell, node, and servers. Short names identify the process to z/OS facilities, such as SAF. Long names are used as the primary external identification for the process. This is the name you will see in the administrative console. (The job manager server long name is set to the fixed value of adminagent.)

Tip: Assign each management server (administrative agent, deployment manager, or job manager) its own cell name that is different from that of any other WebSphere Application Server cell on the same z/OS sysplex.

- Cluster transition name

If this server is converted into a clustered server, this name becomes the cluster short name. The cluster short name is the WLM APPLENV name for all servers that are part of the same cluster.

We used the following values:

- Cell short name: WAADMA
- Cell long name: waadma
- Node short name: WANODEA
- Node long name: wanodea
- Server short name: WAS01A
- Server long name: adminagent
- Cluster transition name: WAC01

Click **Next**.

10. In the Configuration File System window, enter the following information:

- Mount point

Application server configuration file system mount point: Specifies the read/write file system directory where the application data and environment files are written. This field is not writable here, but was specified earlier on the “System Environment: Configuration file system information” window.

- Directory path name relative to mount point

The relative path name of the directory within the configuration file system in which the application server configuration resides.

- Data set name

The file system data set that you create and mount at the specified mount point.

- File system type

Select to allocate and mount your configuration file system data set using HFS or zFS.

- Volume, or '*' for SMS

Specify either the DASD volume serial number to contain the above data set or "*" to let SMS select a volume. Using "*" requires that SMS automatic class selection (ACS) routines be in place to select the volume. If you do not have SMS set up to handle data set allocation automatically, list the volume explicitly.

- Primary allocation in cylinders

The initial size allocation for the configuration file system data set. In the application server, the total space needed for this data set increases with the size and number of the installed applications. The minimum suggested size is 250 cylinders (3390).

- Secondary allocation in cylinders

The size of each secondary extent. The minimum suggested size is 100 cylinders.

We used the following values:

- Mount point: /wasv8config/waadma/wanodea
- Directory path name relative to mount point: AdminAgent
- Data set name: OMVS.WAS80.WAADMA.WANODEA.ZFS
- File system type: zSeries File System (zFS)
- Volume, or “*” for SMS: *
- Primary allocation in cylinders: 420
- Secondary allocation in cylinders: 100

Click **Next**.

11.Specify the information for the product file system:

- Specify the name of the directory where the product files for WebSphere Application Server for z/OS were stored during installation.
- Select the option to allow to set up an intermediate symbolic link and specify the path name.

We used the following values:

- Product file system directory: /opt/zWebSphere/V8R0
- Intermediate symbolic link: Create intermediate symbolic link
- Path name of intermediate symbolic link: /wasv8config/waadma/wanodea/wasInstall

Click **Next**.

12.Enter the process information. The job names for the processes are provided in the window and cannot be changed. Specify the procedure name for each process.

- Controller process job and procedure name

The job name for the control region is the same as the server short name. This is the name used in the MVS START command to start the region.

- Servant process job and procedure name

The job name is used by WLM to start the servant regions. This is set to the server short name followed by the letter “S.”

We used the following values:

- Controller process job name: WAS01A
- Controller process procedure name: WAACRA
- Servant process job name: WAS01AS
- Servant process procedure name: WAASRA

Click **Next**.

13. Specify the application server port values, as shown in Figure 5-36. Good planning is important to avoid port conflicts. Ensure that you have all values that you need to complete the information in this window.

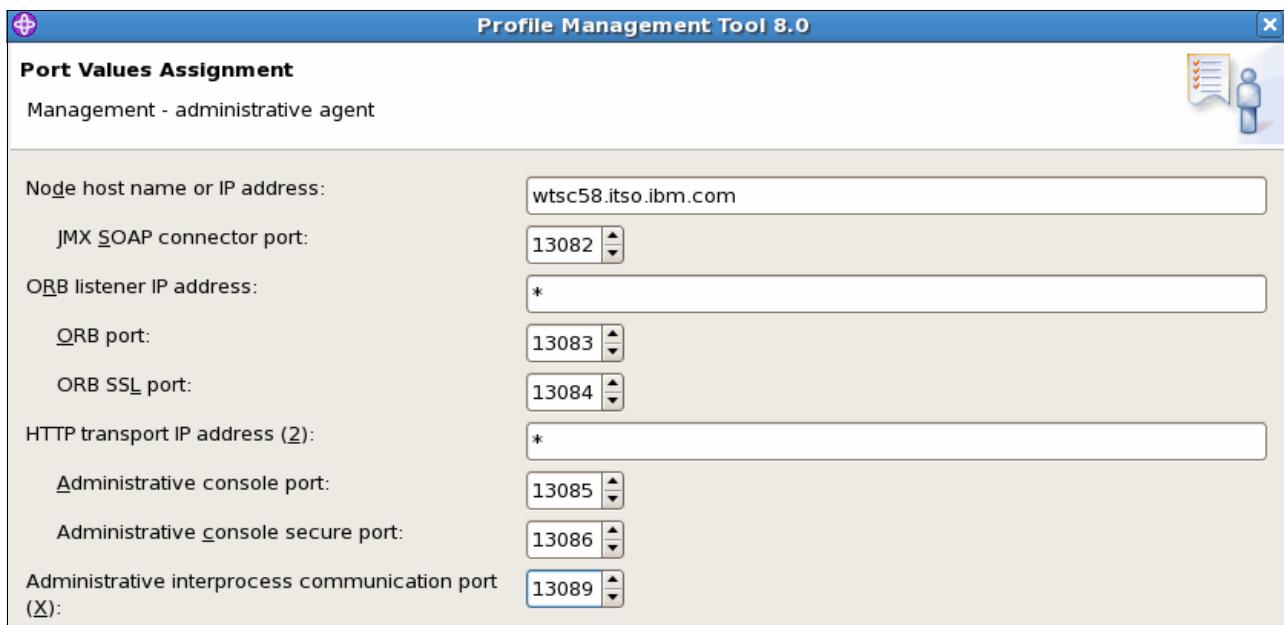


Figure 5-36 Admin Agent port values

14. In the Location Service Daemon Definitions window, enter the following information:

- Daemon home directory
 - Specifies the directory in which the location service daemon resides. This is set to the configuration file system mount point / daemon and cannot be changed.
- Daemon job name
 - Specifies the job name of the location service daemon, specified in the JOBNAM parameter of the MVS start command used to start the location service daemon.
- Procedure name
 - Defines the name of the member in your procedure library to start the location service daemon.
- IP Name
 - Specifies the fully qualified IP name that is registered with the Domain Name Server (DNS), which the location service daemon uses.
- Listen IP
 - Specifies the address at which the daemon listens.
- Port
 - Provides the port number on which the location service daemon listens.
- SSL port
 - Provides the port number on which the location service daemon listens for SSL connections.
- Register daemon with WLM DNS
 - If you use the WLM DNS (connection optimization), you must select this option to register your location service daemon with it; otherwise, do not select it.

We used the following values:

- Daemon home directory: /wasv8config/waadma/wanodea/Daemon
- Daemon job name: WADEMNA
- Procedure name: WADEMNA
- IP name: wtsc58.itso.ibm.com
- Listen IP address: *
- Port: 13080
- SSL Port: 13081
- Register daemon with WLM DNS: Not enabled

Click **Next**.

15. Enter the information that is required for SSL connections.

We entered the following values:

- Certificate authority keylabel: WebSphereCAAA
- Generate certificate authority (CA) certificate: Enabled
- Expiration date for certificates: 2021/12/31
- Default SAF keyring name: WASKeyring.WAADMA
- Use virtual keyring for z/OS SSL clients: Not enabled
- Enable SSL on location service daemon: Not enabled

Click **Next**.

16. Select the user registry that will be used to manage user identities and the authorization policy. Select one of the following options:

- z/OS security product

This option uses the z/OS system's SAF compliant security product, such as IBM RACF or equivalent, to manage WebSphere Application Server identities and authorization according to the following rules:

- The SAF security database will be used as the WebSphere user repository.
- SAF EJBROLE profiles will be used to control role-based authorization, including administrative authority.
- Digital certificates will be stored in the SAF security database.

Important: Select the z/OS security product option if you are planning to use the SAF security database as your WebSphere Application Server registry or if you plan to set up an LDAP or custom user registry whose identities will be mapped to SAF user IDs for authorization checking. For this security option, you must decide whether to set a security domain name, and choose an administrator user ID and an unauthenticated (guest) user ID.

- WebSphere Application Server security

The WebSphere Application Server administrative security option is used to manage the Application Server identities and authorization as follows:

- A simple file-based user registry will be built as part of the customization process.
- Application-specific role binds will be used to control role-based authorization.
- The WebSphere Application Server console users and groups list will control administrative authority.
- Digital certificates will be stored in the configuration file system as keystores.

Tip: Choose this option if you plan to use an LDAP or custom user registry without mapping to SAF user IDs. (The file-based user registry is not recommended for production use.)

- No security

Although it is not a best practice, you can disable administrative security. If you choose this security option, there are no other choices to make. Your WebSphere Application Server environment will not be secured until you configure and enable security manually. You can enable security manually later using the administrative console or using Jython scripts.

We used the “Use z/OS security product” option.

Click **Next**.

17. Next, select one of the following options:

- SAF profile prefix (formally known as *Security domain identifier*)

This optional parameter is used to distinguish between APPL or EJBROLE profiles based on security domain name. It provides an alphanumeric security domain name of one to eight characters. Internally, this sets SecurityDomainType to the string cellQualified.

All servers in the cell will prepend the security domain name you specify to the application-specific J2EE role name to create the SAF EJBROLE profile for checking. The security domain name is not used, however, if role checking is performed using WebSphere Application Server for z/OS bindings.

The security domain name is also used as the APPL profile name and inserted into the profile name used for CBIND checks. The RACF jobs that the Customization Dialog generates create and authorize the appropriate RACF profiles for the created nodes and servers.

If you do not want to use a security domain identifier, leave this field blank.

- WebSphere Application Server unauthenticated user ID

Associated with unauthenticated client requests. It is sometimes referred to as the “guest” user ID. It should be given the RESTRICTED attribute in RACF, to prevent it from inheriting UACC-based access privileges. The UNIX System Services UID number for the user ID is specified here and is associated with unauthenticated client requests. The UID value must be unique numeric values between 1 and 2,147,483,647.

- Enable Writable SAF Keyring support

This feature allows administrative console to create and sign certificates by a CA, connect to keyrings, remove from keyrings, and import, export, and renew.

All certificates created with the writable keyring support are generated and signed by Java code and not by SAF. In this case, the writable keyring support only uses SAF to store the generated certificates.

The writable keyring support is completely optional. New keystores and truststores marked as read-only can be created independently from the writable keyring support. When using the read-only JCERACFKS and JCECCARACFS keystores, the certificates in the appropriate SAF keyring can still be viewed in the administrative console.

We used the following values:

- SAF profile prefix (optional): WA
- WebSphere Application Server unauthenticated user ID: WAGUEST
- UID: Allow OS security to assign UID
- Enable Writable SAF Keyring support: Enabled

Click **Next**.

18. Next, tailor the JCL for the customization jobs. Enter a valid job statement for your installation on this window. The profile creation process will update the job name for you in all the generated jobs, so you need not be concerned with that portion of the job statement. If continuation lines are needed, replace the comment lines with continuation lines. Click **Next**.
19. In the Customization Summary panel, click **Create** to store this application server environment definition for later transfer to the intended z/OS host system.
20. Click **Finish** when the jobs are complete.

5.5.2 Uploading jobs and the associated instructions

Next, upload these jobs and the associated instructions to a pair of z/OS partitioned data sets:

1. On the main window, select the customization definition for the profile, and click **Process**. To upload the generated jobs to the target z/OS system select from the following options:
 - Upload to target z/OS system using FTP
 - Upload to target z/OS system using FTP over SSL
 - Upload to target z/OS system using secure FTP
 - Export to local file systemClick **Next**.
2. If you choose to upload the customization using FTP, in the upload customization definition window, enter the target z/OS system. This path name must be fully qualified or the upload will fail. You must also specify the user ID and password and FTP server port. Click **Allocate target z/OS data sets** to specify whether to allocate the data sets if they do not exist. If the data sets exist and are to be reused, clear this option.
Click **Finish**, and a progress bar displays while the upload is occurring.
3. After the customization profile is uploaded, select the customization definition in the Profile Management Tool and click the **Customization Instructions** tab. This tab provides complete instructions about how to build the profile using the jobs.
These instructions helps you determine the jobs to run, the order in which to run them, and the expected results. It also explains how to start the environment after you are finished.
After the jobs run successfully, the application server profile is complete.



Part 2

Administration and configuration techniques



Administration consoles and commands

WebSphere Application Server properties are stored in the configuration repository as XML files. It is not a good idea to manually edit any of the configuration files because this bypasses the validation of any changes and could lead to synchronization-related problems. Rather, WebSphere Application Server provides administrative tools that help you administer the environment. These tools manage modifications to the files in the repository.

In this chapter, we introduce the administrative consoles and command-line administration. Information about scripting is provided in Chapter 8, “Administration with scripting” on page 343.

We cover the following topics:

- ▶ Introducing the WebSphere administrative consoles
- ▶ Securing the administrative console
- ▶ Job manager console
- ▶ Using command-line tools

6.1 Introducing the WebSphere administrative consoles

The WebSphere Integrated Solutions Console, often referred to as the administrative console, is a graphical, web-based tool that you use to configure and manage the resources within your WebSphere environment. The administrative console application name is *isclite*, and it is a system application. A system application indicates that the application is central to a WebSphere Application Server product and it is installed when the product is installed. In this case, the administrative console application is installed during profile creation if selected, or afterwards using the command line. You do not see system applications in the list of installed applications when using the administrative console. As a system application, you cannot stop or start the application directly, nor can you uninstall the application.

With the introduction of the flexible management topologies, there are multiple administrative consoles available in a WebSphere solution:

- ▶ Administrative console hosted by an application server or deployment manager:

This administrative console is used to manage an entire WebSphere cell. It supports the full range of product administrative activities, such as creating and managing resources and applications, viewing product messages, and so on.

In a stand-alone server environment, the administrative console is located on the application server and can be used to configure and manage the resources of that server only.

In a distributed server environment, the administrative console is located in the deployment manager server, dmgr. In this case, the administrative console provides centralized administration of multiple nodes. Configuration changes are made to the master repository and pushed to the local repositories on the nodes by the deployment manager.

- ▶ Administrative agent console:

An administrative agent hosts the administrative console for application server nodes that are registered to it.

When you access the URL for the administrative console, you can select the node type to manage. After your selection is made, you will be directed to the appropriate administrative console where you can log in:

- Administrative console for the administrative agent:

This console allows you to manage the administrative agent, including security settings, and registering nodes that it controls with the job manager.

- Administrative console for an application server:

This console is the administrative console for the application server.

- ▶ Job manager administrative console (referred to as the job manager console):

The job manager console provides the interface to manage the job manager itself, including security settings and mail resources. Its primary function, though, allows you to submit jobs for processing on the nodes that are registered to it.

6.1.1 Starting and accessing the consoles

The way that you access the administrative console is the same whether you have a stand-alone server environment or a distributed server environment. However, the location and how you start the necessary processes will vary.

Finding the URL for the console

Each server process that hosts the administrative console has two admin ports that are used to access the administrative console. These ports are referred to as:

- ▶ WC_adminhost
- ▶ WC_adminhost_secure (for SSL communication)

These ports are assigned at profile creation. If you do not know what the port number is for the administrative console, you can look in the following location:

profile_home/properties/portdef.props

You can always use the following URL to access the administrative console:

`http://<hostname>:<WC_adminhost>/ibm/console`

If administrative security is enabled, you will automatically be redirected to the secure port.

Administrative console in a stand-alone server environment

In a single application server installation, the administrative console is hosted on the application server. The server must start to access the administrative console.

To access the administrative console, complete the following steps:

1. Make sure that application server, server1, is running by entering the following command:

`serverStatus.sh -all`

The **serverStatus** command is used to obtain the status of one or all of the servers configured on the node. You provide the server name as an argument, or use the **-all** argument. Example 6-1 shows the output from the command.

Example 6-1 Example output for the serverStatus command

```
[/opt/IBM/WebSphere/AppServer/profiles/AppServer01/bin] serverStatus.sh -all
ADMU0116I: Tool information is being logged in file /IBM/WebSphere
    /AppServer/profiles/AppServer01/logs/serverStatus.log
ADMU0120I: Starting tool with the AppSrv01 profile
ADMU0503I: Retrieving server status for all servers
ADMU0505I: Servers found in configuration:
ADMU0506I: Server name: server1
ADMU0509I: The Application Server "server1" cannot be reached. It appears to be
stopped.
[/opt/IBM/WebSphere/AppServer/profiles/AppServer01/bin]
```

2. If the status of server1 is not STARTED, start it with the following command:

`startServer.sh server_name`

3. Open a web browser to the URL of the administrative console. For example:

`http://<hostname>:9060/ibm/console`

`<hostname>` is your host name for the machine running the application server.

4. The administrative console loads and you are prompted to log in.

Administrative console in a distributed environment

If you are working with a deployment manager and its managed nodes, the administrative console is hosted on the deployment manager. You must start the deployment manager to use the administrative console.

To access the administrative console, complete the following steps:

1. Make sure that deployment manager, (dmgr) is running by entering the following command:

```
serverStatus.sh -all
```

2. If the dmgr status is not STARTED, start it with the following command:

```
startManager.sh
```

3. Open a web browser to the URL of the administrative console. For example:

```
http://<hostname>:9060/admin
```

<hostname> is your host name for the machine running the deployment manager.

4. The administrative console loads and you are prompted to log in.

Job manager console

To access the job manager administrative console, complete the following steps:

1. Make sure that job manager process (jobmgr) is running by using the following command:

```
serverStatus.sh -all
```

2. If the status of jobmgr is not STARTED, start it with the following command:

```
startServer.sh jobmgr
```

3. Open a web browser to the URL of the administrative console. For example:

```
http://<hostname>:9960/ibm/console
```

4. The administrative console loads and you are prompted to log in.

Administrative agent administrative console

To access the administrative agent administrative console, complete the following steps:

1. Make sure that administrative agent process (adminagent) is running by using the following command:

```
serverStatus.sh -all
```

2. If the status of adminagent process is not STARTED, start it with the following command:

```
startServer.sh adminagent
```

3. Open a web browser to the URL of the administrative console. For example:

```
http://<hostname>:9060/ibm/console
```

If you have nodes registered with the administrative agent, you will be prompted to select which node to you would like to administer, which includes the administrative agent.

4. The administrative console loads and you are prompted to log in.

6.1.2 Logging in to an administrative console

When you access the administrative console, you need to log in by providing a user ID. If WebSphere administrative security is enabled, you also need to provide a password.

The user ID specified during login is used to track configuration changes made by the user. This allows you to recover from unsaved session changes made under the same user ID, for example, when a session times out or the user closes the web browser without saving. The configuration files are copied from the master repository and cached in the temporary workspace, as you navigate through different console areas. Configuration changes are stored in the `wstemp` temporary workspace directory until the changes are merged with the

master repository during a save operation in the administrative console. Workspaces are not removed when you log out, so they can be reused in another login session for the same login user ID.

Note: You cannot log in to two instances of administrative consoles that are running on the same machine from a single browser type. For example, if you use Firefox to log in to the deployment manager administrative console, you cannot also log in to a job manager running on the same machine.

There is a limitation that cookies are unique per domain rather than a combination of domain and port. Therefore, the cookies that control the session and authentication data in the first browser tab or window get overwritten when logging into the other console in a new browser tab or window. However, you should be able to log into two consoles simultaneously from two completely different browsers for example, Firefox and Internet Explorer.

- ▶ If WebSphere administrative security is not enabled:

You can enter any user ID, valid or not, to log in to the administrative console. The user ID is used to track changes to the configuration, but is not authenticated. You can also simply leave the User ID field blank and click the **Log In** button.

Note: Logging in without an ID is not a good idea if you have multiple administrators.

- ▶ If WebSphere administrative security is enabled:

You must enter a valid user ID and password that has been assigned an administrative security role.

If you enter a user ID that is already in session, you will receive the message “Another user is currently logged in with the same User ID”, and you will be prompted to do one of the following actions:

- Log out the other user with the same user ID. You will be allowed to recover changes that were made in the other user’s session.
- Return to the login page and specify a different user ID.

Figure 6-1 illustrates the login window.



Figure 6-1 Administrative console login window

Note: You can also get this message if a previous session ended without a logout. For example, if the user closed a web browser during a session and did not log out first or if the session timed out.

Recovering from an interrupted session

Until you save the configuration changes you make during a session, the changes do not become effective. During a save operation, the changes propagate and merge with the master configuration repository. If a session is closed without saving the configuration changes made during the session, these changes are remembered and you are given the chance to pick up where you left off. The changes are currently stored in the `wstemp` temporary workspace directory.

When unsaved changes for the user ID exist during login, you are prompted to complete one of the following actions:

- ▶ Work with the master configuration:
Selecting this option specifies that you want to use the last saved administrative configuration. Changes made to the user's session since the last saving of the administrative configuration will be lost.
- ▶ Recover changes made in a prior session:
Selecting this option specifies that you want to use the same administrative configuration last used for the user's session. It recovers all changes made by the user since the last saving of the administrative configuration for the user's session.

As you work with the configuration, the original configuration file and the new configuration file are stored in a folder at:

`<profile_home>/wstemp`

After you save the changes, these files are removed from the `wstemp` directory and merged with the master configuration repository.

6.1.3 Changing the administrative console session timeout

You might want to change the session timeout for the administrative console application. The session timeout is the time it takes for the administrative console session to time out after a period of idleness. The default is 15 minutes. The timeout value can be modified by using a JACL script that is available at the Information Center.

To change the session timeout value, see the page at the Information Center found at the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.nd.doc/info/ae/isc/cons_sessionto.html

6.1.4 The graphical interface

The WebSphere administrative consoles have the same layout pattern. In each administrative console, you can find the following main areas:

- ▶ Banner
- ▶ Navigation tree
- ▶ Work area, including the messages and help display areas.

Each area can be resized as desired. The difference in the administrative console types will be noted in the Navigation tree. The options that you find there will vary depending on the administrative console type.

Figure 6-2 shows the administrative console hosted on a deployment manager to illustrate the console layout.

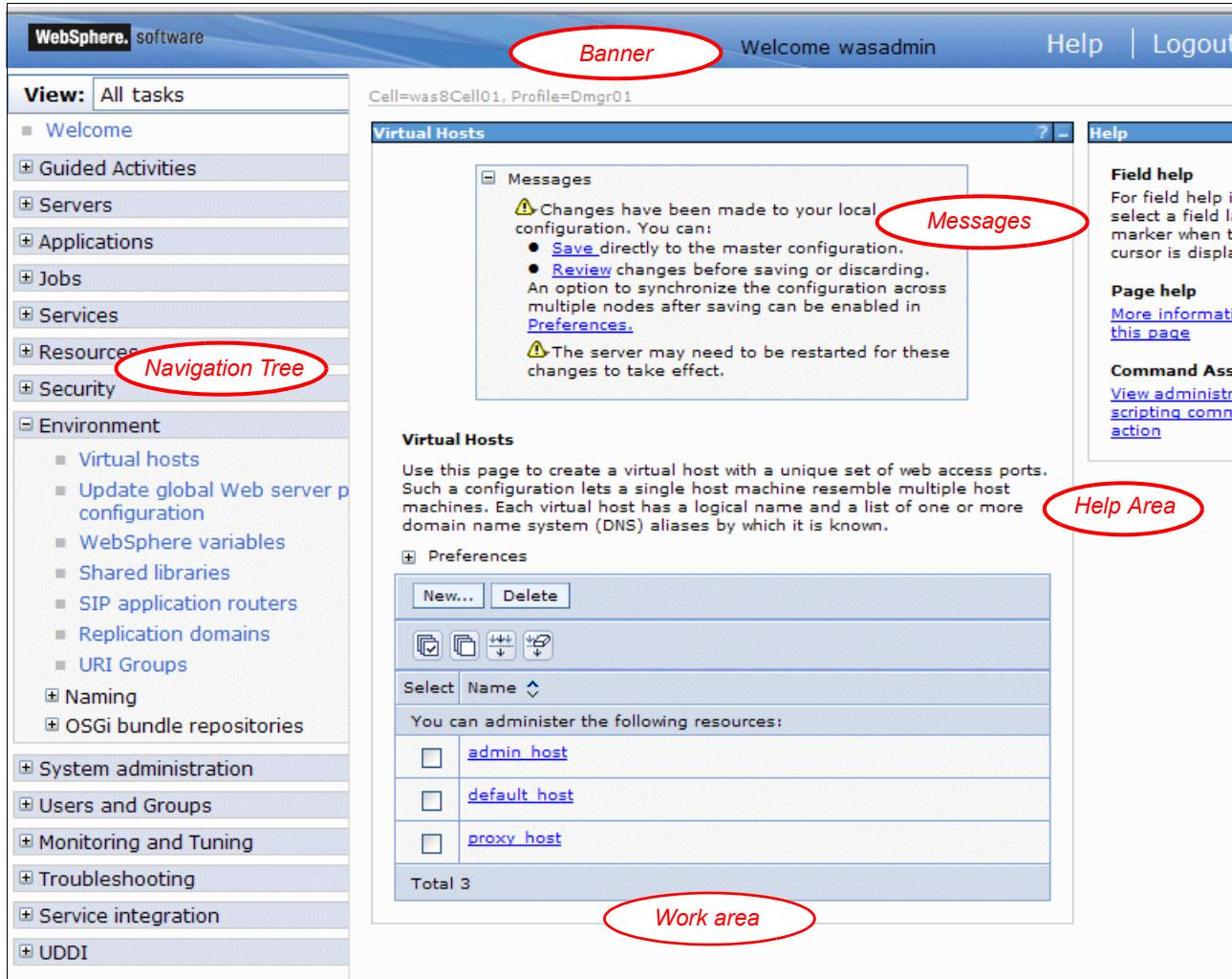


Figure 6-2 The administrative console graphical interface

Banner

The banner is the horizontal bar near the top of the administrative console. It includes a greeting to the user who is logged in. The banner also provides the following actions:

- ▶ Logout logs you out of the administrative console session and displays the Login page. If you have changed the administrative configuration since last saving the configuration to the master repository, the Save page displays before returning you to the Login page. Click **Save** to save the changes, **Discard** to return to the administrative console, or **Logout** to exit the session without saving changes.
- ▶ Help opens a new web browser with detailed online help for the administrative console. This is not part of the Information Center.

Console identity

The banner displays a user ID and it can be customized to show a unique identifier for the administrative console. This can be helpful in cases where administrators log on to multiple administrative consoles. Glancing at the banner lets you know which system you are logged on to. You can add a Console Identity from the administrative console.

To customize the banner, click **System administration** → **Console Identity**. Select **Custom** and enter the identity string, as shown in Figure 6-3.

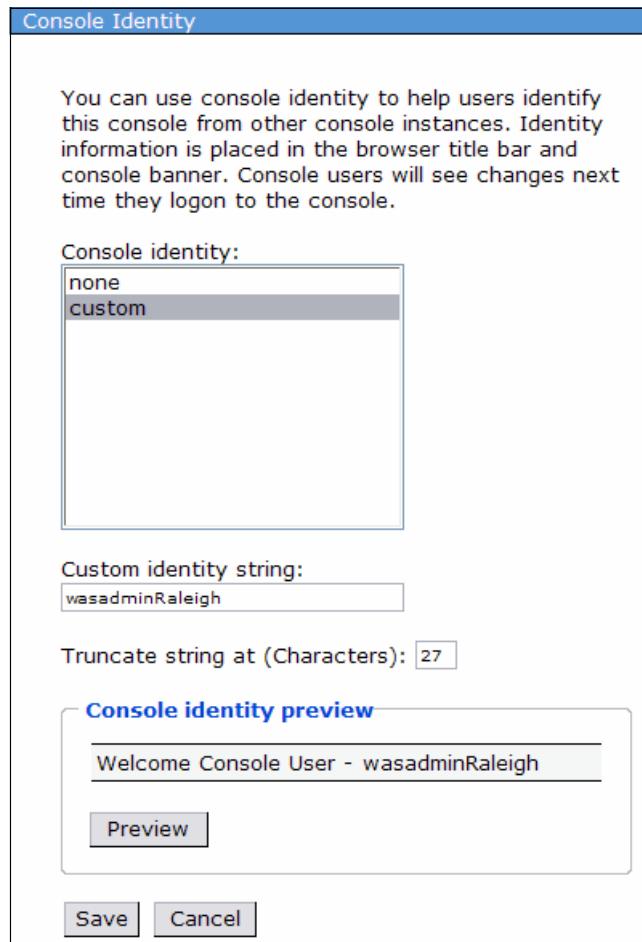


Figure 6-3 Changing the administrative console identity

Save the changes, and log out of the administrative console, then log back in. After you log back in, you will see the new console identity in the banner, as shown in Figure 6-4.



Figure 6-4 Banner with a customized administrative console identity

In an administrative agent configuration, the changes are applied to the administrative agent and all of its registered application servers, regardless of where the changes were actually saved.

Navigation tree

The navigation tree on the left side of the administrative console offers links for you to view, select, and manage components. Figure 6-5 shows the navigation tree.

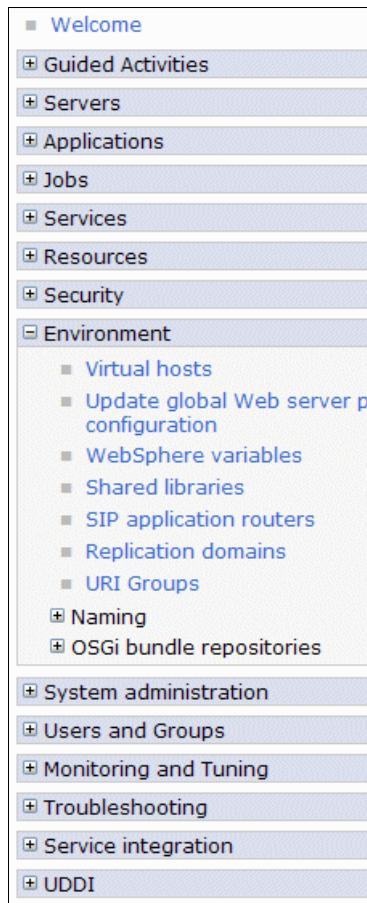


Figure 6-5 Navigation tree

Clicking a + beside a tree folder or item expands the tree for the folder or item. Clicking a - collapses the tree for the folder or item. Double-clicking an item toggles its state between expanded and collapsed.

The content displayed on the right side of the administrative console, the *workspace*, depends on the folder or item selected in the tree view.

Guided Activities

The navigation tree includes a category called “Guided Activities”. This section contains step-by-step assistance for performing some common tasks. These activities can be accomplished by performing each task separately, but using the Guided Activities option provides additional assistance.

Workspace

The workspace, on the right side of the administrative console, allows you to work with your administrative configuration after selecting an item from the administrative console navigation tree.

When you click a folder in the tree view, the workspace lists information about instances of that folder type, the collection page. For example, clicking **Servers** → **Server Types** → **WebSphere application servers** shows all the application servers configured in this cell. Selecting an item, an application server in this example, displays the detail page for that item. The detail page can contain multiple tabs. For example, you might have a Runtime tab for displaying the runtime status of the item, and a Configuration tab for viewing and changing the configuration of the displayed item.

Messages

When you perform administrative actions, messages are shown at the top of the workspace to display the progress and results. These messages are limited in nature, so if an action fails, review the JVM process logs for more detailed information.

When configuration changes have been made, the message area will contain links that you can click to review or save the changes.

Breadcrumb trail

As you navigate into multiple levels of a configuration page, a breadcrumb trail is displayed at the top of the workspace. It indicates how you reached the current page and provides links that allow you to go back to previous pages easily without starting the navigation trail over, as shown in Figure 6-6.

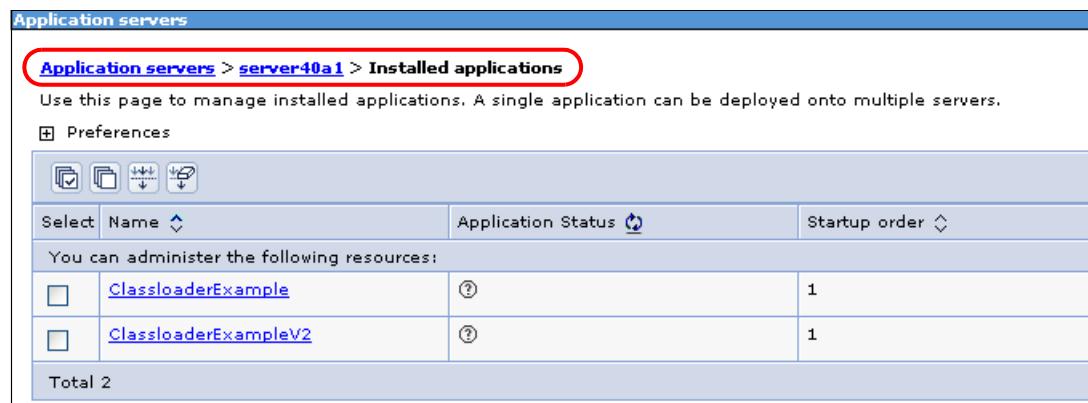


Figure 6-6 Breadcrumb trail

Help area

As you are working in the administrative console, help is available in multiple ways. As you hover the mouse over a field, help text will be displayed for that field.

On the right side of the administrative console, the help portlet displays the Help area. Most pages will have a **More information about this page** link in the Help area. Clicking the link will open the online help in a separate browser. And finally, many pages will have a **View administrative scripting command for last action** link. Clicking this link will display an equivalent scripting command for the action you just performed, as shown in Figure 6-7.



Figure 6-7 Help portlet

The visibility of the help portlet is controlled by console preferences.

Setting console preferences

The look of the administrative console can be altered by setting console preferences. The preference you see will vary slightly depending on the console type. For example, the preference to synchronize changes with nodes is only applicable to an administrative console on a deployment manager. Figure 6-8 displays the console preferences for a deployment manager's administrative console.

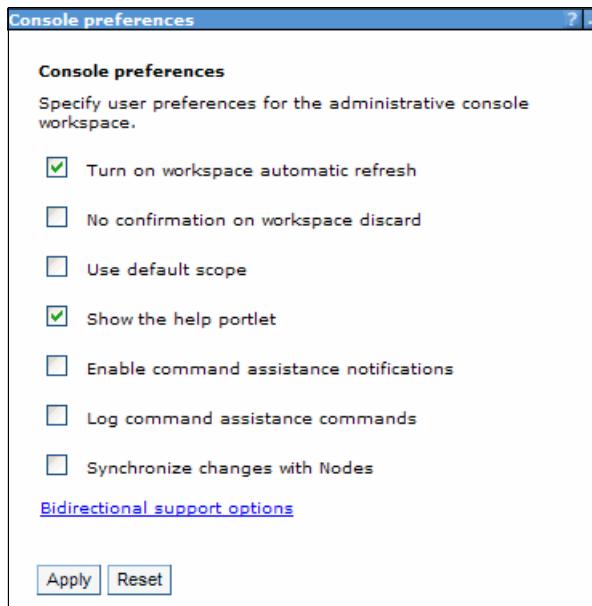


Figure 6-8 Console preferences

To set administrative console preferences, click **System administration** → **Console Preferences** in the navigation tree. You have the following options:

- ▶ **Turn on WorkSpace Auto-Refresh** specifies that the view automatically refreshes after a configuration change. If it is not selected, you must re-access the page to see the changes.
- ▶ **No Confirmation on Workspace Discard** specifies that a confirmation window be displayed if you elect to discard the workspace. For example, if you have unsaved changes and log out of the administrative console, you will be asked whether you want to save or discard the changes. If this option is not selected and you elect to discard your changes, you will be asked to confirm the discard action.
- ▶ **Use default scope** (administrative console node) sets the default scope to the node of the administration console. If you do not enable this setting, the default is all scopes.
- ▶ **Show the help portlet** displays the help portlet at right of the administrative console.
- ▶ **Enable command assistance notifications** allows you to send JMX notifications that contain command data. These notifications can be monitored in a Rational Application Developer workspace, providing assistance in creating scripts.
- ▶ **Log command assistance commands** specifies whether to log all the command assistance wsadmin data for the current user.

When you select this option, script commands matching actions you take in the administrative console are logged to the following location:

profile_root/logs/AssistanceJythonCommands_<user_name>.log

- ▶ **Synchronize changes with Nodes** synchronizes changes that are saved to the deployment manager profile with all the nodes that are running.

Select the boxes to select which preferences you want to enable and click **Apply**.

New in Version 8: The bidirectional support options link allows you to specify bidirectional text preferences for the administrative console. Text is supported in both directions for different types of alphabets. This means the path hierarchy is displayed left-to-right even if elements of the path have right-to-left text.

You can enable Global Preferences, which enables this option for all users and also selects the Current User Preferences option (see Figure 6-9). If you only want to enable this support for the current user logged into the administrative console, enable only the Current User Preferences.

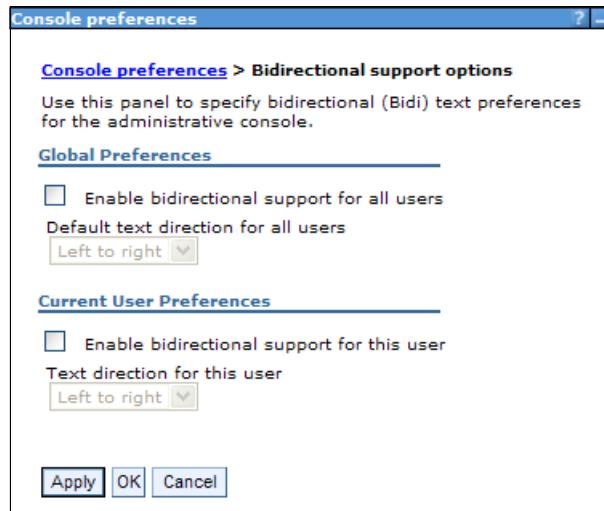


Figure 6-9 Bidirectional support options

6.1.5 Finding an item in the administrative console

To work with items in the administrative console, complete the following steps:

1. Select the associated task from the navigation tree. For example, to display JDBC providers that have been defined, click **Resources** → **JDBC** → **JDBC providers**. The window shown in Figure 6-10 on page 237 opens.
2. Certain resources are defined at a scope level. If applicable, select the scope from the drop-down menu.
3. Set the preferences to specify how you would like information to be displayed on the page.

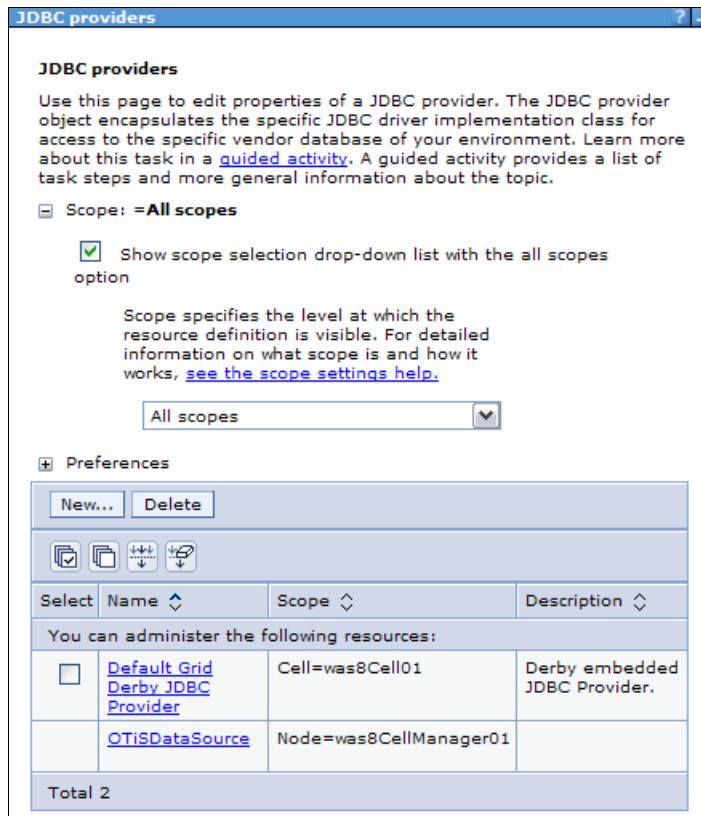


Figure 6-10 Administrative console scope area

Selecting a scope

The scope level determines which applications or application servers will see and use that configuration. The scope setting is available for all resource types, WebSphere variables, shared libraries, and name space bindings.

Scope levels

Configuration information is defined at the following levels: cell, cluster, node, server, and application. Here, we list these scopes in overriding sequence. Because you see application scope first, anything defined at this scope overrides any conflicting configuration you might find in the higher level scopes.

1. Resources and variables scoped at the *application* level apply only to that application. Resources and variables are scoped at the application level by defining them in an enhanced EAR. They cannot be created from the WebSphere administrative tools, but can be viewed and modified (in the administrative console, navigate to the details page for the enterprise application and click **Application scoped resources** in the References section).
2. Resources scoped at the *server* level apply only to that server. If a node and server combination is specified, the scope is set to that server. Shared libraries configured in an enhanced EAR are automatically scoped at the server level.
3. Resources scoped at the *node* level apply to all servers on the node.
4. Resources scoped at the *cluster* level apply to all application servers in the cluster. New cluster members automatically have access to resources scoped at this level. If you do not have any clusters defined, you will not see this option.
5. Resources scoped at the *cell* level apply to all nodes and servers in the cell.

Stand-alone application servers: Although the concept of cells and nodes is more relevant in a managed server environment, scope is also set when working with stand-alone application servers. Because there is only one cell, node, and application server, and no clusters, simply let the scope default to the node level.

Configuration information is stored in the repository directory that corresponds to the scope. For example, if you scope a resource at the node level, the configuration information for that resource is in:

```
<profile_home>/config/cells/cell_name/nodes/<node>/resources.xml
```

If you scoped that same resource at the cell level, the configuration information for that resource is in:

```
<profile_home>/config/cells/cell_name/resources.xml
```

Setting scope levels in the administrative console

Collection pages that contain items that require a scope level to be identified provide two different options for defining the scope. Setting the scope level sets the level for any resources you create and limits what is displayed in the collection page.

Clicking the **Show scope selection drop-down list with the all scopes** option provides a drop-down box with all scopes from which you can select, including the “All scopes” option, as shown in Figure 6-11. Selecting a scope from the drop-down list changes the scope automatically.

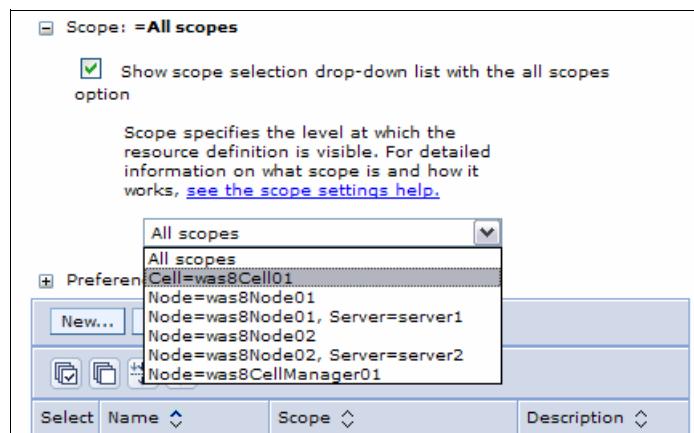


Figure 6-11 Scope level selection

The second option for setting the scope is to clear **Show scope selection drop-down list with the all scopes**. Instead of a drop-down menu, you have fields for each scope level where you can browse a list of applicable entries at that scope level. Click **Apply** to complete the selection, as shown in Figure 6-12 on page 239.

The scope is set to the lowest level entry you select (a red arrow to the left of the field indicates the current scope). To move to a higher scope, simply clear the lower field. For example, if you select a server as the scope level, and want to change the scope to the node level, clear the server field and click **Apply**.

This option is useful in cells that contain a large number of nodes, servers, or clusters. In those situations, the drop-down menu can be difficult to navigate. However, note that the option to view all scopes is not available.

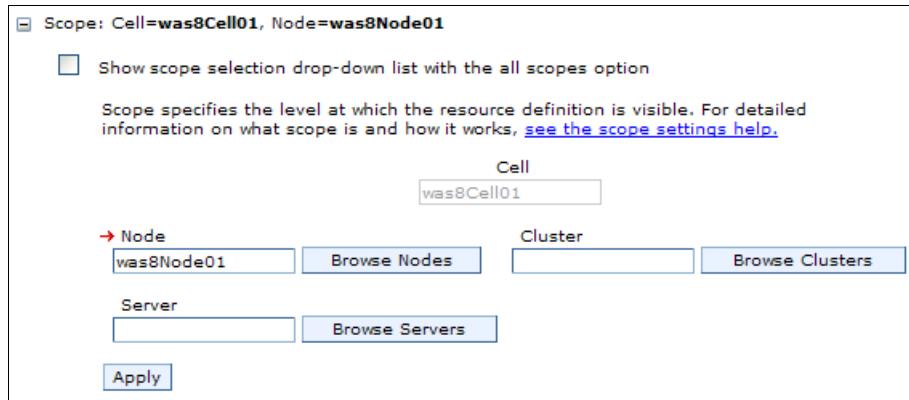


Figure 6-12 Selecting the scope with individual fields

Setting preferences for viewing the administrative console page

After selecting a task and a scope, the administrative console page shows a collection table with all the objects created at that particular scope. You can change the list of items you see in this table by using the filter and preference settings.

The preference settings available will vary by the type of item you are displaying. A list of the preference settings and their use is available in the Information Center.

The Administrative console preference settings can be found at the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.nd.doc/info/ae/ae/rcon_preferences.html

Figure 6-13 illustrates the preference settings you would see when displaying a list of JDBC providers.

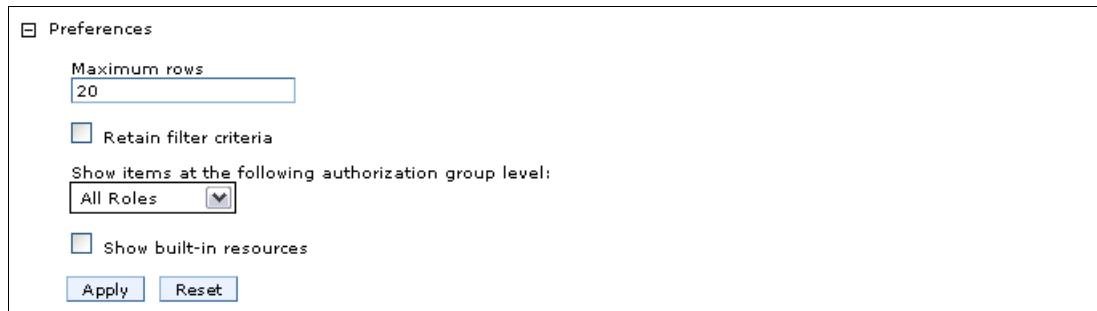


Figure 6-13 Filter and preference settings

The filter options can be displayed or set by clicking the Show Filter Function icon  at the top of the table, as shown in Figure 6-14.

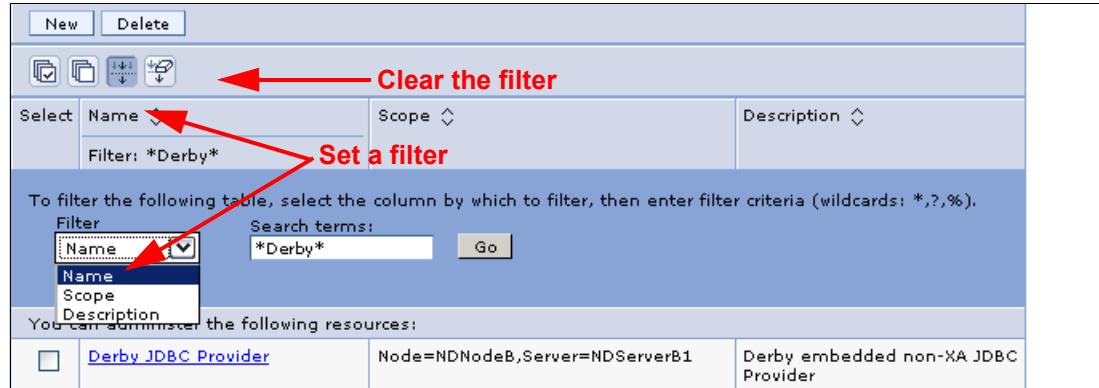


Figure 6-14 Setting filters and preferences

When you click the icon, a new area appears at the top of the table, allowing you to enter filter criteria. To filter entries, complete the following steps:

1. Select the column to filter on. For example, in Figure 6-15 on page 241, the display table has three columns from which to choose. Your options vary depending on the type of item you are filtering.
2. Enter the filter criteria. The filter criteria is case sensitive and wild cards can be used. In our example, to see only providers with names starting with "S", select the Name column to filter on and enter S* as the filter.
3. Click **Go**.
4. After you have set the filter, click the **Show Filter** icon again to remove the filter criteria from view. You still have a visual indication that the filter is set at the top of the table.

Setting the filter is temporary and only lasts for as long as you are in that collection. To keep the filter active for that collection, select the **Retain filter criteria** box in the Preferences section and click **Apply**. To clear the filter criteria, click the .

For more help on using the filtering feature, see the Administrative console buttons section at the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=/com.ibm.websphere.nd.multiplatform.doc/info/ae/ae/rcon_buttons.html

6.1.6 Updating existing items

To edit the properties of an existing item, complete the following steps:

1. Select the category and type in the navigation tree. For example, select **Servers** → **Server Types** → **WebSphere application servers**.
2. A list of the items of that type in the scope specified will be listed in a collection table in the workspace area. Click an item in the table. This action opens a detail page for the item.
3. In some cases, you see a Configuration tab and a Runtime tab on this page. In others, you only see a Configuration tab.

Updates are done under the Configuration tab. Specify new properties or edit the properties already configured for that item. The configurable properties will depend on the type of item selected.

For example, if you select a WebSphere Application Server cluster, this action opens a detail page resembling Figure 6-15.

Figure 6-15 Editing an application server cluster properties

A Local Topology tab is sometimes displayed that displays the topology that is currently in use for the this administrative object. Some detail pages do not have a Local Topology tab.

The detail page provides fields for configuring or viewing the more common settings and links to configuration pages for additional settings.

4. Click **OK** to save your changes to the workspace and exit the page. Click **Apply** to save the changes without exiting. The changes are still temporary. They are only saved to the workspace, not to the master configuration. Those changes still needs to be done.
5. As soon as you save changes to your workspace, you will see a message in the Messages area reminding you that you have unsaved changes, as shown in Figure 6-16.

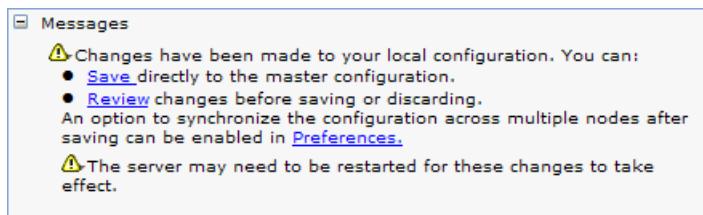


Figure 6-16 Save message

At intervals during your configuration work and at the end, you should save the changes to the master configuration by clicking **Save** in the Messages area, or by clicking **System administration** → **Save Changes to master repository** in the navigation tree.

To discard changes, use the same options. These options simply display the changes you have made and give you the opportunity to save or discard.

6.1.7 Adding new items

To create new instances of most item types, complete the following tasks:

1. Select the category and type in the navigation tree.
2. Click **Scope**. (When creating a new item, you cannot click the **All** option for scope.)
3. Click the **New** button above the collection table in the workspace, as shown in Figure 6-17.

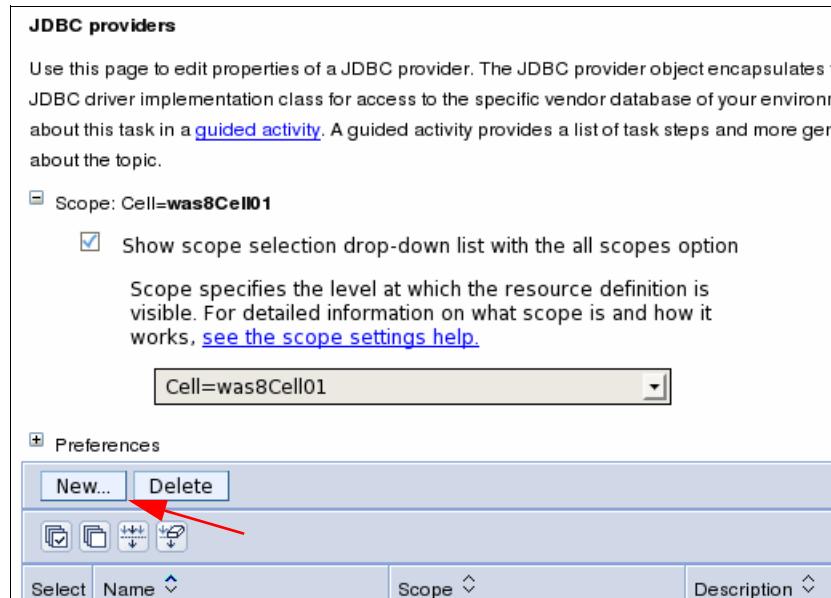


Figure 6-17 Create a new provider

When you click **New** to add an item, one of two things will happen, depending on the type of item you are creating. A wizard will start to guide you through the definitions, or a new details page will open allowing you to fill in the basic details. In the latter case, enter the required information and click **Apply**. This action will usually activate additional links to detail pages required to complete the configuration.

Note: In the configuration pages, you can click **Apply** or **OK** to store your changes in the workspace. If you click **OK**, you will exit the configuration page. If you click **Apply**, you will remain in the configuration page. As you are becoming familiar with the configuration pages, you should always click **Apply** first. If there are additional properties to configure, you will not see them if you click **OK** and leave the page.

4. Click **Save** in the task bar or in the Messages area when you are finished.

6.1.8 Removing items

To remove an item, complete the following steps:

1. Find the item.
2. Select the item in the collection table by selecting the box next to it.
3. Click **Delete**.
4. If asked whether you want to delete it, click **OK**.
5. Click **Save** in the Messages area when you are finished.

For example, to delete an existing JDBC provider, click **Resources** → **JDBC** → **JDBC providers**. Select the provider you want to remove and click **Delete**, as shown in Figure 6-18.

The screenshot shows the 'JDBC providers' page in an administrative console. At the top, there is a descriptive text about JDBC providers. Below it, a section titled 'Scope' is expanded, showing a checked checkbox for 'Show scope selection drop-down list with the all scopes option'. A note explains that 'Scope' specifies the level at which the resource definition is visible. A dropdown menu is set to 'Cell=was8Cell01'. Below this, a 'Preferences' section is expanded, showing 'New...' and 'Delete' buttons. A red arrow points to the 'Delete' button. The main table lists one provider: 'Default Grid Derby JDBC Provider' under 'Name', 'Cell=was8Cell01' under 'Scope', and 'Derby embedded JDBC Provider.' under 'Description'. The table has columns for 'Select', 'Name', 'Scope', and 'Description'. A footer at the bottom of the table says 'Total 1'.

Figure 6-18 Deleting an item

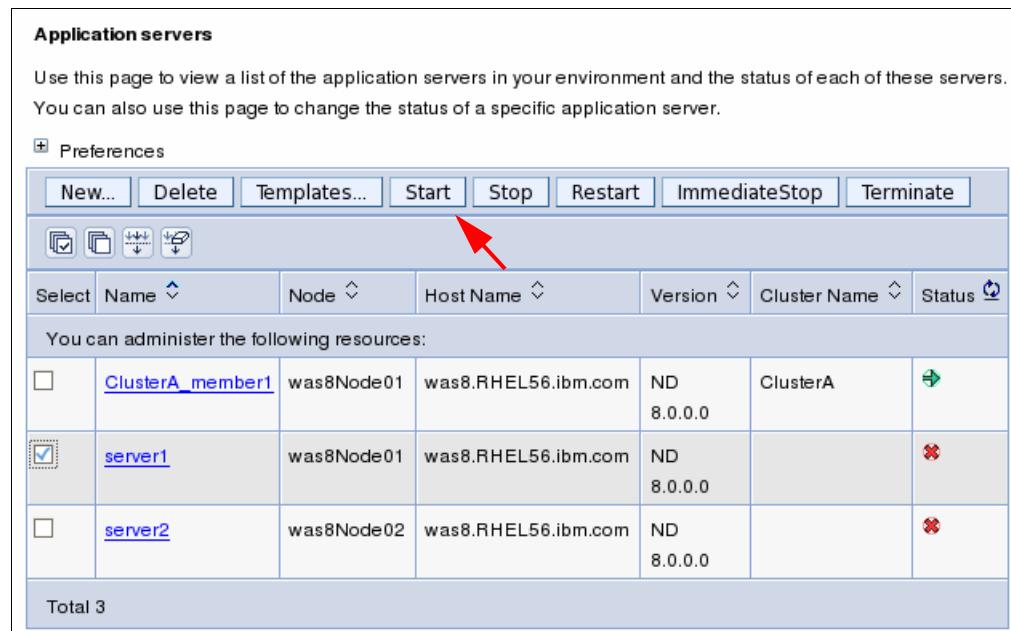
6.1.9 Starting and stopping items

To start or stop an item using the administrative console, complete the following steps:

1. Select the category and type in the navigation tree.
2. Select the item in the collection table by selecting the box next to it.

3. Click **Start** or **Stop**. The collection table shows the status of the item, as shown in Figure 6-19.

For example, to start an application server in a distributed server environment, click **Servers** → **Server Types** → **WebSphere application servers**. Select the check box beside the application server you want and click **Start**.



The screenshot shows the 'Application servers' page. At the top, there is a toolbar with buttons for New..., Delete, Templates..., Start, Stop, Restart, ImmediateStop, and Terminate. A red arrow points to the 'Start' button. Below the toolbar is a search bar with fields for Select, Name, Node, Host Name, Version, Cluster Name, and Status. Underneath is a section titled 'You can administer the following resources:' containing a table. The table has columns for a checkbox, Name, Node, Host Name, Version, Cluster Name, and Status. It lists three items: 'ClusterA_member1' (unchecked), 'server1' (checked), and 'server2' (unchecked). The status for 'server1' is marked with a red 'X'. At the bottom of the table, it says 'Total 3'.

Select	Name	Node	Host Name	Version	Cluster Name	Status
<input type="checkbox"/>	ClusterA_member1	was8Node01	was8.RHEL56.ibm.com	ND 8.0.0.0	ClusterA	
<input checked="" type="checkbox"/>	server1	was8Node01	was8.RHEL56.ibm.com	ND 8.0.0.0		
<input type="checkbox"/>	server2	was8Node02	was8.RHEL56.ibm.com	ND 8.0.0.0		

Figure 6-19 Starting and stopping a server

Not all items can be started and stopped from the administrative console. For example, the deployment manager and nodes must be started independently from the administrative console. Also, there can be multiple options for starting and stopping an item (restart, stop immediate, and so on.).

6.1.10 Using variables

WebSphere variables are name and value pairs used to represent variables in the configuration files, which makes it easier to manage a large configuration.

To set a WebSphere variable, complete the following steps:

1. Select **Environment** → **WebSphere variables**, as shown in Figure 6-20.

The screenshot shows the 'WebSphere Variables' configuration interface. At the top, there is a descriptive text about substitution variables and their scope levels. Below this, a section titled 'Scope' is expanded, showing 'Cell=was8Cell01, Node=was8Node01'. Under this, a checkbox 'Show scope selection drop-down list with the all scopes option' is checked, and a dropdown menu shows 'Node=was8Node01'. A link 'see the scope settings help.' is provided. The 'Preferences' section is collapsed. Below this, there is a table for managing resources, with a row for 'APP_INSTALL_ROOT' set to value '\${USER_INSTALL_ROOT}/installedApps' and scope 'Node=was8Node01'. There are also icons for creating new resources and deleting existing ones.

Figure 6-20 WebSphere variables

2. To add a new variable, click **New**, or click a variable name to update its properties.
3. Enter a name and value and click **Apply**, as shown in Figure 6-21.

This is a 'General Properties' dialog box. It contains fields for 'Name' (set to 'DB2_JDBC_DRIVER_PATH'), 'Value' (set to '/opt/IBM/sqllib/java'), and 'Description' (set to 'The directory that contains the DB2 JDBC Driver'). At the bottom, there are four buttons: 'Apply', 'OK', 'Reset', and 'Cancel'. The 'Apply' button is highlighted.

Figure 6-21 New variable

6.1.11 Saving work

As you work with the configuration, your changes are saved to temporary workspace storage. For the configuration changes to take effect, they must be saved to the master configuration.

If you have a distributed server environment, a second step is required to *synchronize*, or send, the configuration to the nodes. Consider the following possibilities.

If you work on a page, and click **Apply** or **OK**, the changes are saved in the workspace under your user ID. This action allows you to recover changes under the same user ID if you exit the session without saving.

You need to save changes to the master repository to make them permanent. You have several options:

- ▶ Use the Save window in the Messages area. If it is open, it is the quickest method.
- ▶ Click **System administration** → **Save Changes to master repository**.
- ▶ When you log in, if you logged out without saving the changes, you will be given the option to save the changes.

The Save window presents you with the following options:

- ▶ Save.
- ▶ Discard: This option reverses any changes made during the working session and reverts to the master configuration.
- ▶ Cancel: This option does not reverse changes made during the working session. It just cancels the action of saving to the master repository for now.
- ▶ Synchronize changes with nodes: This action distributes the new configuration to the nodes in a distributed server environment.

Before deciding whether you want to save or discard changes, you can see the changed items by expanding **Total changed documents** in the Save window.

Important: All the changes made during a session are cumulative. Therefore, when you decide to save changes to the master repository, all changes are committed. There is no way to be selective about what changes are saved to the master repository

6.1.12 Getting help

Help is available to you in several different ways:

- ▶ Click **Help** on the administrative console banner. This action opens a new web browser with online help for the administrative console. It is structured by administrative tasks, as shown in Figure 6-22 on page 247.

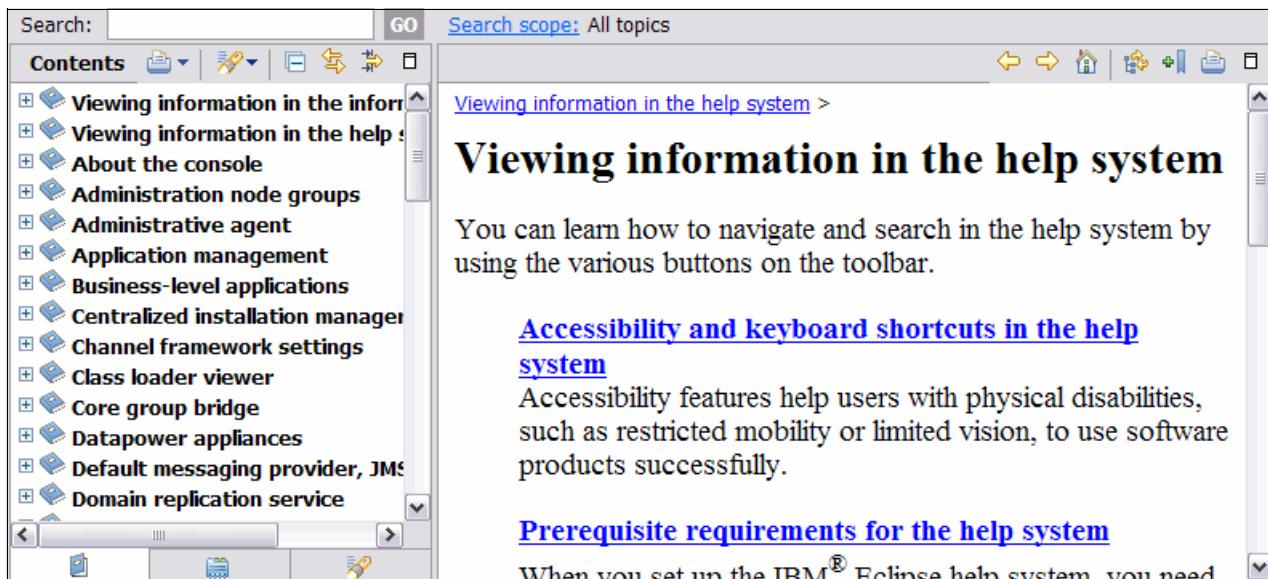


Figure 6-22 Administrative console help

- ▶ With the option **Show the help portlet** enabled, you can see the Help window in the workspace. Click **More information about this page**. This action will open the help system to a topic-specific page.
- ▶ The Information Center can be viewed online or downloaded from the following website:
http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.nd.doc/info/ae/ae/welcome_nd.html

The help in the administrative console banner provides a searchable index for administrative tasks. There are a number of icon-based help control functions that allow you to navigate the help area.

6.2 Securing the administrative console

WebSphere Application Server provides the ability to secure the administrative consoles so only authenticated users can use them by enabling administrative security. Administrative security determines whether security is used at all, provides authentication of users using the WebSphere administrative function, the type of registry against which authentication takes place, and other values. Enabling administrative security activates the settings that protect your server from unauthorized users. Note that enabling administrative security does not enable application security.

Before enabling any type of security for a production system, you should be familiar with WebSphere security and have a plan for securing your WebSphere environment. Security encompasses many components, including administrative security, application security, infrastructure security, and specialized resource security options. This section only provides an overview of administrative security.

The first decision you have to make is to select the user registry you will use. If you enable security when you create a profile for distributed systems, a file-based registry is automatically created and populated with one administrative user ID. On z/OS platforms, you have the option of using the file-based registry or the z/OS system's SAF-compliant security database.

While a file-based user registry is not a good choice for securing applications, you can federate additional registries to the existing file-based registry to manage users and groups for application security.

If you are using a registry other than the WebSphere Application Server federated user registry, you must create at least one user ID to be used for the WebSphere administrator.

Although you might have heard about security domains that have been introduced in WebSphere Application Server V7, these domains are used for application security (not administrative security).

Before implementing security in a production environment, be sure to consult *WebSphere Application Server V7 Security Guide*, SG24-7660.

6.2.1 Enabling security after profile creation

You can enable administrative security after profile creation through the administrative console by navigating to **Security → Global security**. Performing this action allows you more flexibility in specifying security options. You need to complete the configuration items for authentication, authorization, and realm (user registry). You will also need to populate the chosen user registry with at least one user ID to be used as an administrator ID. Figure 6-23 illustrates the security settings page.

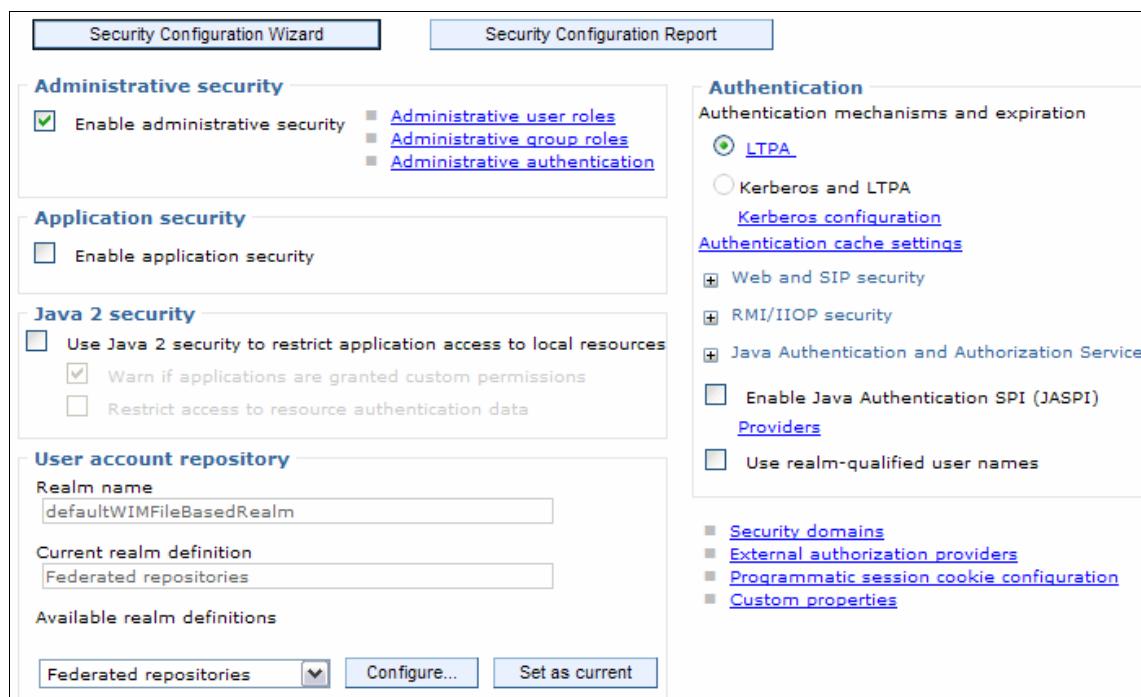


Figure 6-23 Security settings page

Attention: Be aware that when you select the box to enable administrative security, the application security check box is enabled automatically. Application security requires that administrative be enabled. If you are not prepared to use application security at this time, be sure to clear the box. Selecting either of these options no longer selects Java 2 security. Java 2 security can be selected at any time.

Tip: If you enable administrative security, and then find you cannot log in, you can disable security through scripting or by manually editing the security.xml profile. This action allows you to go back through the security configuration to see where things went wrong.

Because editing an XML configuration file manually is not a good idea, you can use scripting to enable or disable administrative and application security, and modify a few other security settings. To disable administrative security through scripting, complete the following steps:

1. Navigate to the `dmgr_profile_home/bin` directory.
2. Start the `wsadmin` scripting client with the `-conntype none` argument.
3. Enter the `securityoff` command in JACL mode or `securityoff ()` command in Jython mode.
4. Exit the `wsadmin` scripting client.
5. Restart your processes.

When starting the `wsadmin` scripting client with the `-conntype none` argument, the `securityoff` command toggles the `enabled="true"` setting in `security.xml` to `enabled="false"`. The `wsadmin` session is in local mode and in this case acts as a text editor to make the needed configuration change.

To manually edit the `security.xml` file, complete the following steps:

1. Open the `security.xml` file at `dmgr_profile_home/config/cells/cell_name`.
2. Edit the second line, changing `enabled="true"` to `enabled="false"` as follows:

```
<security:Security xmi:version="2.0" xmlns:xmi="http://www.omg.org/XMI"
xmlns:orb.securityprotocol="http://www.ibm.com/websphere/appserver/schemas/5.0/
orb.securityprotocol.xmi"
xmlns:security="http://www.ibm.com/websphere/appserver/schemas/5.0/security.xmi"
" xmi:id="Security_1" useLocalSecurityServer="true"
useDomainQualifiedUserNames="false" enabled="false" cacheTimeout="600"
issuePermissionWarning="false" activeProtocol="BOTH"
enforceJava2Security="false" enforceFineGrainedJCASecurity="false"
appEnabled="false" dynamicallyUpdateSSLConfig="true" allowBasicAuth="true"
activeAuthMechanism="LTPA_1" activeUserRegistry="WIMUserRegistry_1"
defaultSSLSettings="SSLConfig_1">
```

If administrative security is enabled, each time you log in to the administrative console, you must authenticate with the user ID that was identified as having an administrative role. Entering commands from a command window will also prompt you for a user ID and password. You can add additional administrative users and assign authorization levels from the administrative console.

6.2.2 Administrative security roles

Administrative security is based on identifying users or groups that are defined in the active user registry and assigning roles to each of those users. When you log in to the administrative console or issue administrative commands, you must use a valid administrator user ID and password. The role of the user ID determines the administrative actions the user can perform.

Fine-grained administrative security

Prior to WebSphere Application Server V6.1, users granted administrative roles could administer all of the resource instances under the cell. With Version 6.1, administrative roles are now per resource instance rather than to the entire cell. Resources that require the same privileges are placed in a group called the *authorization group*. Users can be granted access to the authorization group by assigning to them the required administrative role within the group.

A cell-wide authorization group exists for backward compatibility. Users who are assigned to administrative roles in the cell-wide authorization group can still access all of the resources within the cell.

The following administrative security roles are available:

- ▶ Administrator:

The administrator role has operator permissions, configurator permissions, and the permission required to access sensitive data, including server password, Lightweight Third Party Authentication (LTPA) password and keys, and so on.

- ▶ Auditor:

The auditor role has permission to view and change the configuration settings for the security auditing subsystem.

- ▶ Configurator:

The configurator role has monitor permissions and can change the WebSphere Application Server configuration.

- ▶ Operator:

The operator role has monitor permissions and can change the runtime state. For example, the operator can start or stop services.

- ▶ Monitor:

The monitor role has the least permissions. This role primarily confines the user to viewing the WebSphere Application Server configuration and current state.

- ▶ Deployer:

The deployer role has permission to perform both configuration actions and runtime operations on applications.

- ▶ Admin Security Manager:

The Admin Security Manager role gives permissions to users granted this role the ability to map other users to administrative roles. When fine-grained administrative security is used, users granted this role can manage authorization groups. A user mapped to the administrator role does not have permissions to map users to administrative roles.

- ▶ ISC Admins:

The ISC Admins role has administrator privileges for managing users and groups from within the administrative console only.

You can find more information about each of these roles at the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.nd.doc/info/ae/ae/rsec_adminroles.html

Assigning administrative roles to users and groups

If you are using a file-based repository, you can add users and groups through the console by clicking **Users and Groups** → **Manage Users** or **Users and Groups** → **Manage Groups**. Otherwise, the users and groups must be added to the user registry using the tools provided by the registry product.

Role assignments for users and groups are managed through the administrative console.

Click **Users and Groups** → **Administrative user roles** or **Users and groups** → **Administrative group roles**. Use these windows to assign an administrative role to a user or group.

Fine-grained security

WebSphere Application Server administrative security is fine-grained, meaning that access can be granted to each user per resource instance. For example, users can be granted configurator access to a specific instance of a resource only (an application, an application server, or a node). The administrative roles are now per resource instance rather than to the entire cell.

To achieve the instance-based security or fine-grained security, resources that require the same privileges are placed in a group called the administrative authorization group or authorization group. Users can be granted access to the authorization group by assigning to them the required administrative role.

You can define groups of resources that will be treated collectively by clicking **Security** → **Administrative Authorization Groups**. The resource instances which are added to an authorization group can be the following types:

- ▶ Clusters
- ▶ Nodes
- ▶ Servers, including application servers and web servers
- ▶ Applications, including business level applications
- ▶ Node groups
- ▶ Assets

After the authorization group has been created, you can assign users or groups an administrative role for the authorization group.

Many administrative console pages have a preference setting that allows you to restrict the items that you can see to those that are valid for your authorization group level. The roles that you can choose from depend on the role of the user ID you are logged into the administrative console with.

6.3 Job manager console

The job manager console has many of the basic options that you find in the administrative console, including global security settings, the option to add users and groups to the federated user repository, WebSphere variable settings, and others that are common to any administrative environment. What is unique to the job manager administrative console is the ability to submit jobs to nodes registered to it.

New in Version 8: In WebSphere Application Server V8, you can also complete job manager actions and run jobs from the deployment managers administrative console. The deployment manager administrative console has a Jobs navigation tree option that is similar to that in the job manager console, where you have the following options:

- ▶ Submit a job.
- ▶ Review the status of a job.
- ▶ Identify job manager target for job, target resources used in job, and target groups for administrative jobs.

Figure 6-24 shows a job manager administrative console. The option selected in the Navigation tree is **Jobs** → **Targets**. You can see in this example that one application server node has been registered from an admin agent. Two deployment manager nodes are registered as well.

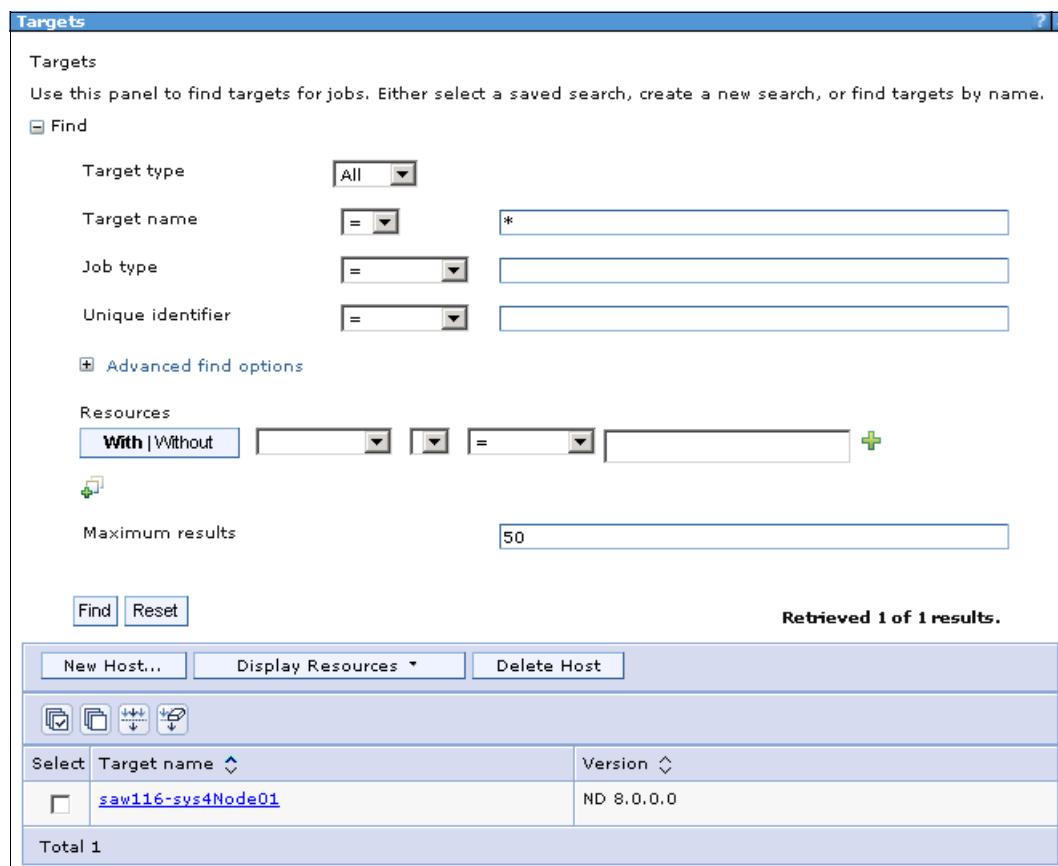


Figure 6-24 Job manager administrative console - List of nodes

Groups of nodes: You can create “groups of nodes” that contain the nodes you will work with from the job manager (click **Jobs** → **Groups of nodes**). A group of nodes can be used as the target of administrative jobs.

When you submit a job, you can select one or more groups from a drop-down menu. The alternative is to type in the name of the node or use the Find feature to select each node. Using the Find feature takes several steps.

So, even if you do not plan to use multiple nodes as the target of a job, creating a group for each node allows you to easily select a node rather than typing it in or searching for it.

If you include multiple nodes in the group, beware that all the nodes have to have a common user ID and password. When you submit a job, you only have one place where you can enter the user ID and password.

6.3.1 Submitting a job with the job manager

The job manager provides the following job types:

- ▶ Run a wsadmin script
- ▶ Manage applications
 - distributeFile
 - collectFile
 - removeFile
 - startApplication
 - stopApplication
 - installApplication
 - updateApplication
 - uninstallApplication
- ▶ Manage servers
 - createApplicationServer
 - deleteApplicationServer
 - createProxyServer
 - deleteProxyServer
 - createCluster
 - deleteCluster
 - createClusterMember
 - deleteClusterMember
 - configureProperties
- ▶ Manage the server run time
 - startServer
 - stopServer
 - startCluster
 - stopCluster

Details about each of these job types can be found at:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=/com.ibm.websphere.nd.doc/info/ae/ae/rxml_7jobtypes.html

Complete the following steps:

1. Start the job manager and log in to the job manager console:

`http://<job_manager_host>:9960/ibm/console`

- To submit jobs, nodes must have already been registered with the job manager. To verify which nodes have been registered, expand **Jobs** in the navigation window, and click **Targets**. If this is the first time you are using the job manager, you might not see all the nodes displayed. To refresh the view, enter * as the value for Node name and click **Find**, as shown in Figure 6-25.

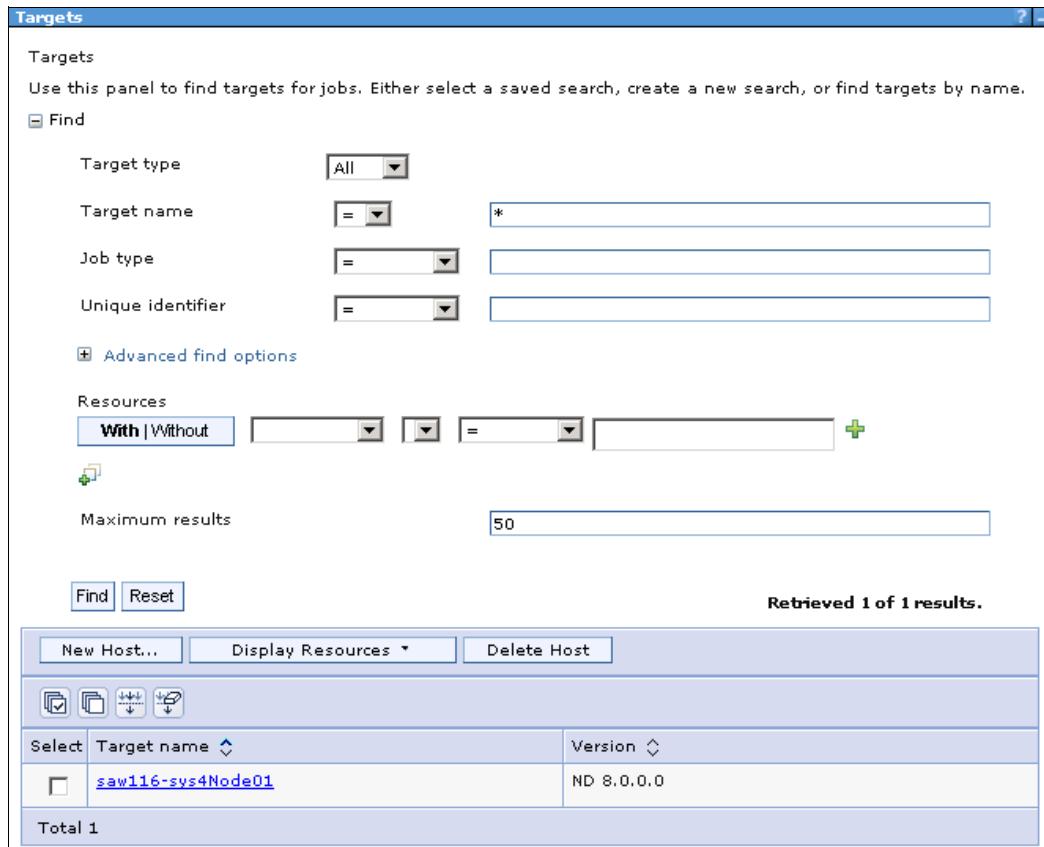


Figure 6-25 List of all nodes registered with job manager

- Click **Jobs** → **Submit** to select the type of job to submit and click **Next**, as shown in Figure 6-26.

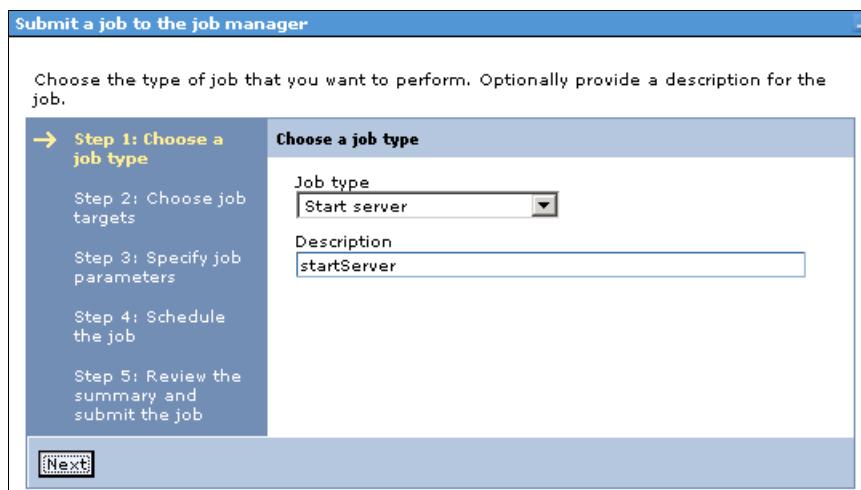


Figure 6-26 Select the job type

4. Select the node on which you want to run the job. You can select the node from a node group by using the drop-down menu next to the **Groups of nodes** option, or you can select specific nodes using the **Node names** option. Enter the user ID and password for the node that you will run the job against, as shown in Figure 6-27.

The screenshot shows a 'Choose job targets' dialog box with the following interface:

- Step 1: Choose a job type**: Step 2: Choose job targets is highlighted.
- Job type: Start server**
- Target groups**: Radio button is unselected. A dropdown menu shows "-- No groups --".
- Target names**: Radio button is selected. An input field contains "saw116-sys4Node01". Buttons for "Add" and "Find..." are visible.
- Remove**: A button located at the bottom right of the target names section.
- Target authentication**
- User name**: Input field contains "wasadmin".
- Password authentication**: Selected radio button. Fields for *** Password** and *** Confirm password** both contain "*****".
- Public-private key authentication**: Unselected radio button. Fields for *** Full path to keystore**, **Passphrase**, and **Confirm passphrase** are present but empty.
- Previous**, **Next**, and **Cancel** buttons at the bottom.

Figure 6-27 Target name defined

To use a specific node, select **Target names** and either enter the node name and click **Add**, or click **Find**. Using the Find option opens a new window where you can search and select nodes, as shown in Figure 6-28.

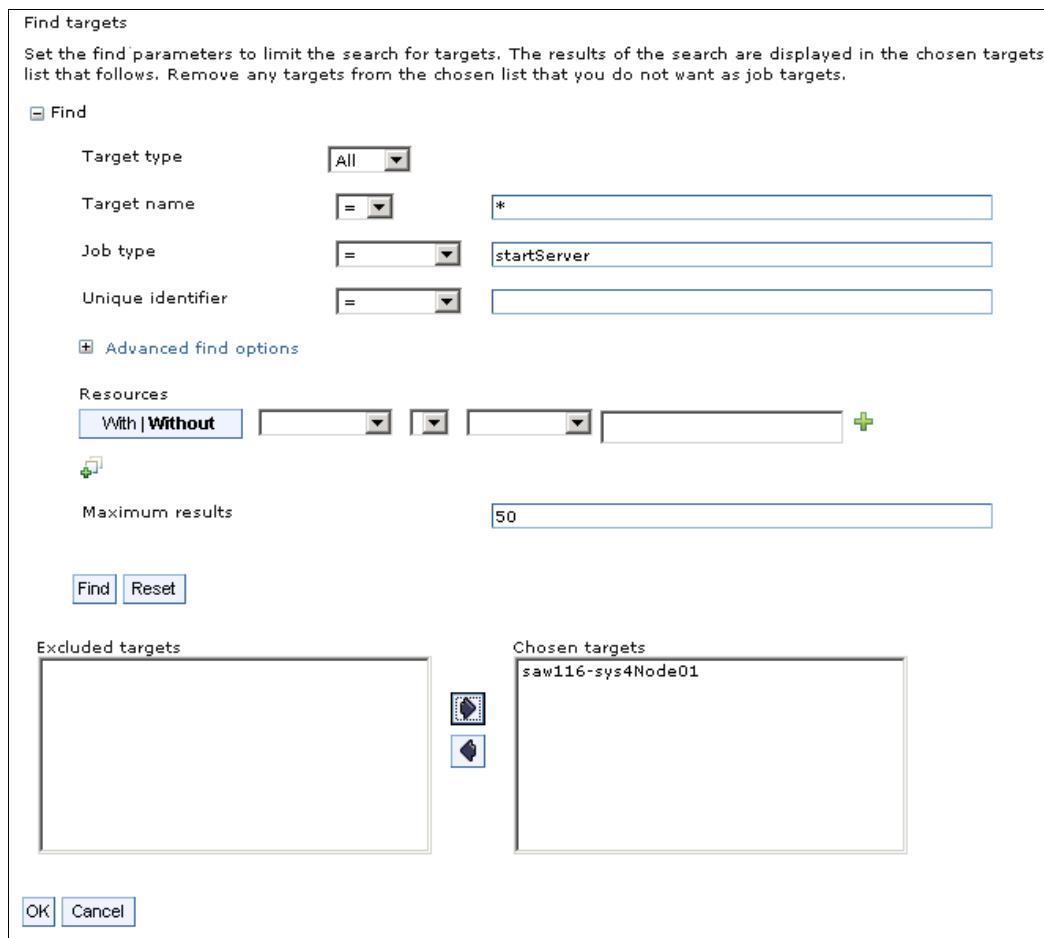


Figure 6-28 Search and select nodes

The simplest method of searching is to enter an “*” in the Node name field and click **Find**.

The list of nodes is shown in the **Excluded nodes** box. Select the nodes you want and use the arrow button to move them to the **Chosen nodes** box. You can hold the Shift key down to select multiple nodes, or move them one at a time.

Click **OK**. This action returns you to step 2 of the wizard with the node name entered.

Click **Next** to continue the job submit process.

- Specify the job parameters. These parameters will vary widely depending on the type of job. The parameters provide the additional information the job will need to perform the task. For example, if you are running a job to start a server, you have selected the node in the previous step, but the server name must be entered as a parameter. You can also click **Find** to search for parameter, as shown in Figure 6-29.

Step 1: Choose a job type

Step 2: Choose job targets

→ Step 3: Specify job parameters

Step 4: Schedule the job

Step 5: Review the summary and submit the job

Specify job parameters

Job type: Start server

* Server name:

Node name:

Figure 6-29 Job type parameters

Click **Next**.

- The next step contains fields that specify how and when the job should run, and if a notification email should be sent, as shown in Figure 6-30.

Step 1: Choose a job type

Step 2: Choose job targets

Step 3: Specify job parameters

→ Step 4: Schedule the job

Step 5: Review the summary and submit the job

Schedule the job

Job type: Start server

Notification

E-mail addresses:

Initial Availability

Specify when this job is first available.

Make the job available now.

Schedule availability

Date (MM/dd/yyyy) / / Time (HH:mm:ss) : :

Expiration

Specify when this job is no longer available.

Use default expiration - 1 days.

Expire the job based on a date

Date (MM/dd/yyyy) / / Time (HH:mm:ss) : :

Expire the job based on a duration

Expire after minutes

Job Availability Interval

Jobs can run repeatedly based on an interval. Specify the interval that the job is available.

Availability interval:

Figure 6-30 Specify job scheduling information

The fields, shown in Figure 6-31, are:

- Notification: The email address specified will receive a notification when the job is finished. To use this field, you must configure a mail provider and mail session.
- Initial availability: You can make the job available now (it will run immediately after you have finished with the job submission process), or you can specify a date and time it will be available.
- Expiration: Specify an expiration date for the job.
- Job availability interval: This field allows you to repeat job submission at intervals. Depending on the selection, you will have an additional field displayed that allows you to choose the days, start and stop time, and so on.



Figure 6-31 Job interval options

If you select **Make this job available now** and **Run once**, the job runs right away and the Expiration settings have no meaning. The alternative is to set an Initial availability, Expiration date or duration, and select an interval at which the job will run.

7. Review the summary and submit the job.

When a job is submitted from the job manager, the job details are saved in a database local to the job manager. The endpoint (deployment manager or administrative agent) pings the job manager at a predefined interval and fetches jobs that are to be executed. If the job submitted is a **wsadmin** job, the **wsadmin** script is executed. Otherwise, a corresponding job handler will execute the necessary admin code.

8. The Job status window allows you to monitor the results. Use the Refresh icon in the Status summary column () to update the status. The color in the Status summary field will indicate the success or failure of the job, as shown in Figure 6-32.

The screenshot shows the 'Job status' window with the following interface elements:

- Job status**: A header section with a 'Find' button and a 'Preferences' link.
- Status summary key:** A row of colored buttons: Succeeded (green), Partially succeeded (yellow), Failed (red), and Incomplete (grey).
- Suspend**, **Resume**, and **Delete** buttons.
- Toolbar icons** for search, refresh, and other operations.
- Table headers**: Select, Job ID, Description, State, Activation Time, Expiration Time, Status Summary, and a refresh icon.
- Data rows**: One row is visible, showing Job ID 13100867724619887, Description startServer, State Active, Activation Time 07/07/2011 20:57:56, Expiration Time 07/08/2011 20:57:56, Status Summary (green bar with red border), and a refresh icon.
- Total 1**: A summary at the bottom of the table.

Figure 6-32 Status summary

- Click the Job ID to see more information about the job, as shown in Figure 6-33.

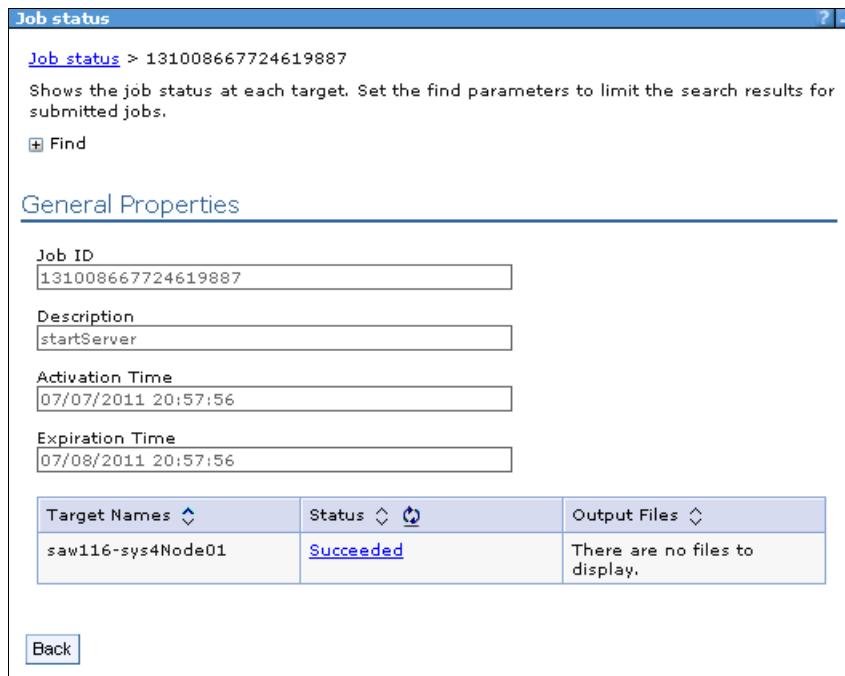


Figure 6-33 Job status information

The job status is always sent back to the job manager. Clicking the message in the Status column (Succeeded in this case) shows you additional information.

In the event of an error, you will see any messages produced by the job. Additional messages might be available in the logs for the server where the administrative action was to take place, as shown in Figure 6-34.

Time stamp	Status	Message
2009-04-01T12:47:49-0400	Distributed	
2009-04-01T12:47:51-0400	In progress	
2009-04-01T12:49:24-0400	Succeeded	CWWWSY0328I: Server sasrv40 was started on node SAsrv40Node

Figure 6-34 Job output

When you execute a configuration type job (for example, create server) from the job manager to a deployment manager, the configuration will be saved if the job is successful. A job that submits a **wsadmin** script, however, will not save the configuration (the **wsadmin** script needs to do that).

Executing a job to a deployment manager does not cause node synchronization to occur. Synchronization will happen at the next automatic synchronization interval, or a **wsadmin** script can be submitted to synchronize.

6.3.2 Distributing files using the job manager

Some job types require that files be transferred to the node where the job will be run. The Distribute file job type can be used to transfer these files.

This type is normally necessary in the following circumstances:

- ▶ When you want to run a `wsadmin` script on the node. The script must be distributed to the node before you can use the Run wsadmin script job.
- ▶ When you want to install or update an application. The EAR file must be distributed to the node before you can use the Install application or Update application jobs.

The following steps show how to distribute a file to a node for use in later jobs. This example distributes a `wsadmin` script file to an admin agent.

1. The file to be distributed from the job manager must be in the `/config/temp/JobManager` directory of the job manager profile.

Create the `jobmgr_profile_root/config/temp/JobManager` directory and copy the file into it.

If you are developing a script or application in Rational Application Developer, you can export the file directly to the directory.

2. The distribute file job stores the file into the `downloadedContent` directory of the administrative agent or deployment manager profile. The destination parameter is relative to the `downloadedContent` directory. You must create this directory on the admin agent or deployment manager:

- `adminagent_profile_home/downloadedContent`
- `dmgr_profile_root/downloadedContent`

3. In the Job manager administrative console, click **Job** → **Submit**. This action will launch the Job properties wizard.

- a. Select **Distribute file** as the job type and click **Next**.

- b. Enter the script file location on the job manager and the location to store the script file on the target node.

In this example, the `appInstall.py` script was stored in the following location:

`jobmgr_profile_root/config/temp/JobManager/appInstall.py`

On the admin agent it will be stored as:

`adminagent_profile_home/downloadedContent/appInstall.py`

The arguments are entered as shown in Figure 6-35.

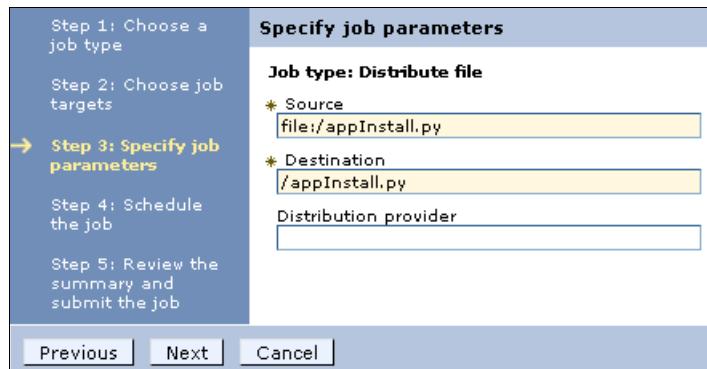


Figure 6-35 File distribution parameters

- Click **Next**.
- c. Use the defaults for the job schedule. The defaults will execute the distribute file job once. Click **Next**.
 - d. Click **Finish**. Monitor the status of the job and ensure it completes successfully.

6.4 Using command-line tools

WebSphere Application Server provides various administrative commands that can be run from a command line. These commands can be used for many administrative tasks, for example, to start, view, or stop a WebSphere process. Many commands have an equivalent GUI interface, either created specifically for the command, through an administrative console, or through a First Steps console. However, it is often convenient to simply enter these commands manually from a command line.

Examples of commands are as follows:

- ▶ **startServer** to start a server process
- ▶ **stopServer** to stop a server process
- ▶ **serverStatus** to obtain the status of servers
- ▶ **registerNode** to register a node with the administrative agent
- ▶ **addNode** to add a node to a cell configuration
- ▶ **startNode** to start the node agent process

New in Version 8: In WebSphere Application Server V8, two additional commands were added to the command-line tools:

- ▶ **addNode -asExistingNode** to recover damaged nodes or move nodes to a new machine or different operating system
- ▶ **managesdk** to manage the software development kits available to a WebSphere Application Server installation

More information about the **addNode -asExistingNode** command is provided in 7.5, “Working with nodes in a distributed environment” on page 301.

6.4.1 Command location

Command-line tools must be run on the system where the process you are entering the command for resides. They cannot operate on a remote server or node. To administer a remote server, you need to use the administrative console or a **wsadmin** scripting client script.

For the most part, the commands exist in two places:

- ▶ *install_root/bin*
Commands entered from this location will operate against the default profile unless you use the **-profileName** parameter to specify the profile.
- ▶ *profile_root/bin*
Commands entered from this location will operate against the profile defined in *profile_root*.

6.4.2 Key usage parameters

The commands are consistent across platforms, though how you enter them, case sensitivity, and the extension will vary.

Note: Parameter values that specify a server name, a node name, or a cell name are always case sensitive regardless of operating system.

There are several commonly used parameters that are valid for every command you should be aware of.

- ▶ **-profileName** specifies the profile against which the command is to run.
- ▶ **-username** specifies the user ID with the administrative privileges required to execute the command.
- ▶ **-password** specifies the password for the user ID specified in **-username**.
- ▶ **-help** will display the usage requirements and a list of parameters for the command.

6.4.3 Entering commands

In this section, we show how to enter commands on the various operating systems.

Windows operating systems

Commands in Windows operating systems have an extension of .bat. It is not necessary to use the extension. Commands are not case sensitive, though parameters and names are case sensitive.

To use a command, complete the following steps:

1. Open a command prompt window.
2. Change to the directory where the command is. For example:

C:\Program Files\IBM\WebSphere\AppServer\profiles\profile_name

3. Enter the command. For example:

serverStatus.bat -all -username <username> -password <password>

Note: When running command-line tools on the Microsoft Windows Vista or later Microsoft operating systems, on the Windows Vista, Windows Server 2008, and Windows 7 operating systems, you can install WebSphere Application Server as either Administrator or non-administrator. When it is installed as Administrator, certain operations (such as those involving Windows Services) require Administrator privileges.

To ensure that WebSphere Application Server command-line tools have sufficient privileges, run them as Administrator. When you run these command-line tools from a command prompt, run them from a command prompt window that is launched by performing the following actions:

1. Right-click a command prompt shortcut.
2. Click **Run As Administrator**.
3. When you open the command prompt window as Administrator, an operating-system dialog appears that asks you if you want to continue. Click **Continue** to proceed.

If you are using a Windows Server Core installation of Windows Server 2008, any WebSphere Application Server commands that require a graphical interface are not supported, because a Windows Server Core system does not have a graphical user interface. Therefore, commands such as **pmt.bat** or **ifgui.bat** are not supported on that type of Windows Server 2008 installation.

UNIX operating systems

Commands in UNIX operating systems have an extension of .sh and are case sensitive.

To use a command, complete the following steps:

1. Open a command prompt or terminal window.
2. Change to the directory where the command is. For example, for root users, the directory would be:
 - AIX: /usr/IBM/WebSphere/AppServer/profiles/*profile_name*/bin
 - HP, Linux, or Solaris: /opt/IBM/WebSphere/AppServer/profiles/*profile_name*/bin

For non-root users, the directory would be:

user_home/IBM/WebSphere/AppServer/profiles/bin

3. Enter the command. For example:

```
serverStatus.sh -all -username <username> -password <password>
```

IBM i operating systems

For an IBM i operating system, complete the following steps:

1. From the IBM i command line, start a Qshell session by issuing the STRQSH CL command.
2. Change to the directory where the command is. For example:
/QIBM/ProdData/WebSphere/AppServer/V8/ND/profiles/*profile_name*/bin
3. Enter the command. For example:

```
serverStatus.sh -all -username <username> -password <password>
```

z/OS operating systems

You can manage application servers on an z/OS system from a UNIX System Services environment as follows:

1. Enter **uss** (to switch to the UNIX System Services environment)
2. Change to the directory where the command is. On z/OS, this directory will be always be *app_server_root*/profiles/default, because only the profile name “default” is used in WebSphere Application Server for z/OS.
3. Enter the command. For example:

```
serverStatus.sh -all -username <username> -password <password>
```




Administration of WebSphere processes

In this chapter, we provide information about basic administration tasks. The focus of this chapter is on managing WebSphere processes, including the deployment manager, nodes and node agents, application servers, and application server clusters.

We cover the following topics:

- ▶ Working with the deployment manager
- ▶ Starting and stopping an administrative agent
- ▶ Starting and stopping the job manager
- ▶ Working with application servers
- ▶ Working with nodes in a distributed environment
- ▶ Working with clusters
- ▶ Working with virtual hosts
- ▶ Managing applications

7.1 Working with the deployment manager

In this section, we provide information about how to manage the deployment manager and introduce you to the configuration settings associated with it.

7.1.1 Deployment manager configuration settings

A deployment manager is built by creating a deployment manager profile. After it is built, there is usually not much that you need to do. However, it is good to note that there are settings that you can modify from the administration tools.

To view the deployment manager from the administrative console, click **System administration** → **Deployment manager**. You have two pages available, the Runtime tab and the Configuration tab. Figure 7-1 shows the Configuration tab.

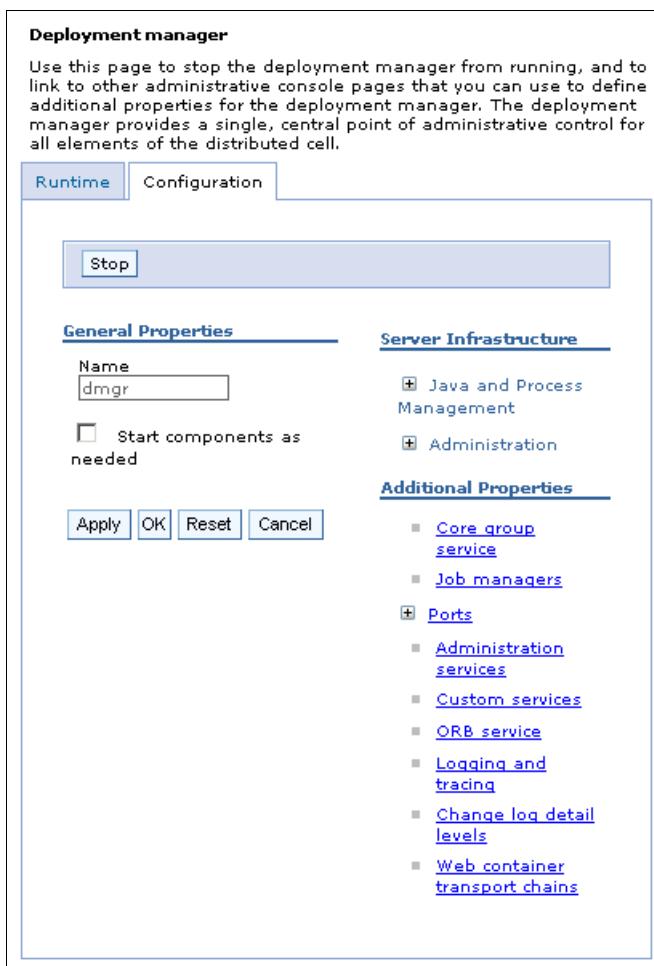


Figure 7-1 Deployment manager configuration tab

Deployment manager Configuration tab

Two options on this window to note are as follows:

- ▶ Job managers

Click this link to view work with job managers. You can view the job managers this deployment manager is registered to and you can register or unregister the deployment manager with a job manager.

► Ports

Select this link to view and manage the ports used by the deployment manager. This option is useful for diagnostics, for example, when the deployment manager does not start because of a port conflict with another process. It is also useful for finding the SOAP connector port required when federating a custom profile.

Deployment manager Runtime tab

In addition to the Configuration page, the administrative console contains a Runtime tab for the deployment manager. To view the Runtime tab, click **System administration** → **Deployment manager** and click the **Runtime** tab at the top of the page. Figure 7-2 shows the Runtime tab.

The screenshot shows the 'Deployment manager' configuration page with the 'Runtime' tab selected. The page has two main sections: 'General Properties' on the left and 'Troubleshooting' and 'Additional Properties' on the right. Under 'General Properties', there are fields for Process ID (4536), Cell name (Cell01), Node name (DeploymentManager01), State (Started), Current heap size (197 MB), and Maximum heap size (256 MB). Under 'Troubleshooting', there is a link to 'Diagnostic Provider service'. Under 'Additional Properties', there is a link to 'Product Information'. A 'Back' button is located at the bottom left of the panel.

Figure 7-2 Deployment manager Runtime tab

The fact that the state is Started does not mean much, because you would not be able to access the administrative console otherwise. Items on window panel to note are as follows:

► Diagnostic Provider service

This option allows you to query components for current configuration data, state data, and to run a self-diagnostic test routine. You would most likely use these options at the request of IBM Support.

► Product information

Selecting this link opens a new page that provides information about the level of code running on the deployment manager system and provides links for more detailed information, including the installation history for the product and maintenance.

The product information is stored as XML files in the *install_root*/properties/version/dtd folder and can be viewed without the administrative console, as shown in Figure 7-3.

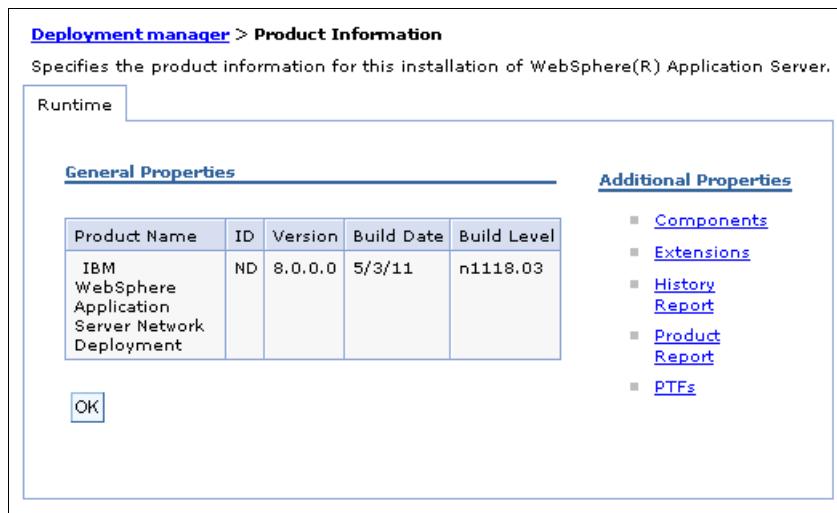


Figure 7-3 Product information

7.1.2 Starting and stopping the deployment manager

The deployment manager must be started and stopped with commands. The administrative console is not available unless it is running.

Starting the deployment manager with startManager

The **startManager** command is used to start the deployment manager on distributed systems, as shown in Example 7-1.

Example 7-1 startManager command

```
C:\WebSphere\AppServer\profiles\dmgr01\bin>startManager
ADMU0116I: Tool information is being logged in file
  C:\WebSphere\AppServer\profiles\dmgr01\logs\dmgr\startServer.log
ADMU0128I: Starting tool with the dmgr01 profile
ADMU3100I: Reading configuration for server: dmgr
ADMU3200I: Server launched. Waiting for initialization status.
ADMU3000I: Server dmgr open for e-business; process id is 1536
```

Run this command from the deployment manager *profile_root*/bin directory. If you run it from the *install_root*/bin directory, use the **-profileName** parameter to ensure that the command is run against the deployment manager profile.

Syntax of startManager

The syntax of the startManager command is:

```
startManager.bat(sh) [options]
```

The options are shown in Example 7-2.

Example 7-2 startManager options

```
Usage: startManager [options]
      options: -nowait
```

```
-quiet  
-logfile <filename>  
-replacelog  
-trace  
-script [<script filename>] [-background]  
-timeout <seconds>  
-statusport <portnumber>  
-profileName <profile>  
-recovery  
-help
```

All arguments are optional. For more information about the **startManager** command, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=/com.ibm.websphere.nd.multiplatform.doc/info/ae/ae/rxml_startmanager.html

Starting the deployment manager on z/OS (START command)

On z/OS, the deployment manager can be started using a JCL start procedure. The exact command can be found in the BBOCCINS instruction member of the JCL generated to create the profile. For example:

```
START WPDCR,JOBNAM=WPDMGR,ENV=WPCELL.WPDMNODE.WPDMGR
```

Where:

- ▶ WPDCR is the JCL start procedure.
- ▶ WPDMGR is the Job name.
- ▶ ENV is the concatenation of the cell short name, node short name, and server short name.

Starting the deployment manager will start the following components:

- ▶ A daemon. In our example, it is named WPDEMN. There will be one daemon per cell per MVS image. One of the functions of the daemon server is to provide the “location name service” for the cell. All daemons in the cell are fully aware of all the objects in the cell and use the same port values.
- ▶ A controller region. In our example, it is named WPDMGR. The controller region serves many functions, including acting as the endpoint for communications.
- ▶ A servant region. In our example, it is named WPDMGRS. The servant region contains the JVM where the applications are run.
- ▶ If you are using messaging, you will also see a control region adjunct server start.

Stopping the deployment manager

The deployment manager is stopped with the **stopManager** command, as shown in Example 7-3. If administrative security is enabled, you must provide an administrative user name and password for the command. You can provide this information by using the **-username** and **-password** options on the command line. If you do not provide the options on the command line, you are prompted to provide the credentials.

Example 7-3 stopManager command

```
C:\WebSphere\AppServer\profiles\dmgr01\bin>stopManager
```

```
ADMU0116I: Tool information is being logged in file
```

```
C:\WebSphere\AppServer\profiles\dmgr01\logs\dmgr\stopServer.log
ADMU0128I: Starting tool with the Dmgr01 profile
ADMU3100I: Reading configuration for server: dmgr
ADMU3201I: Server stop request issued. Waiting for stop status.
ADMU4000I: Server dmgr stop completed.
```

Syntax of stopManager

The syntax of the **stopManager** command is:

```
stopManager.bat(sh) [options]
```

The options are shown in Example 7-4.

Example 7-4 startManager options

```
Usage: stopManager [options]
      options: -nowait
                -quiet
                -logfile <filename>
                -replacelog
                -trace
                -timeout <seconds>
                -statusport <portnumber>
                -conntype <connector type>
                -port <portnumber>
                -username <username>
                -password <password>
                -profileName <profile>
                -help
```

All arguments are optional. For more information about the **stopManager** command, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=/com.ibm.websphere.nd.multiplatform.doc/info/ae/ae/rxml_stopmanager.html

Stopping the deployment manager on z/OS (STOP command)

To stop the deployment manager with a STOP command, use the following format:

```
STOP dmgr_job
```

For example:

```
STOP WPDMGR
```

Stopping the daemon server will stop all servers for that cell, and all the servers on that daemon instance's MVS image will be stopped in an order fashion. For example:

```
STOP WPDEMN
```

Windows start menu and services

On a Windows system, you have the option of starting and stopping the deployment manager using the Start menu. For example, click **Start** → **All Programs** → **IBM WebSphere** → **IBM WebSphere Application Server Network Deployment V8.0** → **Profiles** → *profile_name* → **Start the deployment manager**.

Also, on a Windows system, you have the option of registering the deployment manager as a Windows service. In order to have it registered, you must select this option when you create the deployment manager profile or register it later using the **WASService** command.

For more information about the **WASService** command, see the **WASService** command topic at the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=/com.ibm.websphere.nd.multiplatform.doc/info/ae/ae/rins_wasservice.html

If the deployment manager is registered as a Windows service, all other options for starting the dmgr process are unchanged from the administrator's point of view, however the command used will start or stop the process through the service. In addition, you have the option to allow the service to be started automatically when the operating system starts.

7.2 Starting and stopping an administrative agent

An administrative agent process is managed in the same manner as an application server process. The process name is **adminagent**.

To view the status of the administrative agent process, run the following command:

```
serverStatus -adminagent
```

To start an administrative agent, run the following command:

```
startServer adminagent
```

To stop an administrative agent, run the following command:

```
stopServer adminagent
```

7.3 Starting and stopping the job manager

A job manager process is managed in the same manner as an application server process. The process name is **jobmgr**.

To view the status of the **jobmgr**, run the following command:

```
serverStatus -jobmgr
```

To start a job manager, run the following command:

```
startServer jobmgr
```

To stop a job manager, run the following command:

```
stopServer jobmgr
```

7.4 Working with application servers

This section covers the following topics:

- ▶ Creating an application server
- ▶ Viewing the status of an application server
- ▶ Starting an application server
- ▶ Stopping an application server
- ▶ Viewing runtime attributes of an application server
- ▶ Customizing application servers

Terminology for application server types:

- ▶ A *stand-alone application server* is created with an application server profile and is not federated to a cell or registered with an administrative agent. A stand-alone application server hosts its own administrative services and operates independently from other WebSphere processes. It cannot participate in application server clusters. Each stand-alone application server has its own node.

This option is available on all WebSphere Application Server packages, but is the only option in the Base and Express environments.

- ▶ An *unfederated application server* is an application server that resides on a node managed from an administrative agent. Unfederated application servers have the characteristics of a stand-alone application server, in that they cannot be used in a cluster. However, multiple application servers can exist on one node.

This option is available on all WebSphere Application Server packages.

- ▶ A *managed application server* is one that resides on a node that is managed from a deployment manager. A managed server can either be an application server that was created using an application server profile and subsequently federated to the cell, or it can be created directly from the deployment manager's administrative console.

Managed application servers can be clustered for high-availability and workload balancing. This action is only possible with the Network Deployment package.

7.4.1 Creating an application server

The process to create an application server depends on your WebSphere Application Server package.

Stand-alone application servers

Stand-alone application servers are created by creating an application server profile. This action results in a profile that defines one stand-alone application server. This application server hosts the sample applications and the administrative console application.

In previous versions, a stand-alone server was always named server1. Starting with WebSphere Application Server V7, you have the opportunity to give the server a different name during profile creation.

For information about creating an application server profile, see 3.3.3, “Creating an application server profile” on page 95.

Unfederated application server

An administrative agent can monitor and control multiple unfederated application servers on one or more nodes.

Unfederated application servers can be created in multiple ways:

- ▶ The first server on a node to be managed by an administrative agent must be created with a stand-alone profile and then registered with the administrative agent. The registration process disables the administrative console on the server and makes a console for the application server node available on the administrative agent process. See 3.3.10, “Registering nodes to an administrative agent” on page 125 for more information.
- ▶ Additional unfederated application servers on that node are created from the administrative agent. See “Creating an application server from the administrative console” on page 275 for more information.
- ▶ When you use the administrative agent console to register the application server node to a job manager, additional application servers can be created on the node by submitting a job from the job manager console.

What about wsadmin?

The administrative service remains active in unfederated application servers that are registered to an administrative agent. You can connect to either the application server or the administrative agent to run `wsadmin` commands, but all admin operations performed by connecting to the application server are forwarded to the agent. So you should ideally connect to the agent to avoid that extra step.

Managed application servers

In a distributed server environment, you create an application server from the deployment manager administrative console. See “Creating an application server from the administrative console” on page 275 for more details.

If you are creating an application server with the intention of adding it to a cluster, click **Servers → Clusters → WebSphere application server cluster**. See 7.6, “Working with clusters” on page 310 for more details.

Application server options

You need to consider certain options as you create an application server. The method by which you select these options varies depending on how you are creating the server, but the values are the same.

Templates

An application server is created based on a template that defines the configuration settings. Four template options are provided:

- ▶ default: Standard production server. You get this option if you do not specify a template for a server on a distributed system.
- ▶ defaultZOS: This option is available only on z/OS platforms and is the only option until you create new templates.
- ▶ DeveloperServer: The DeveloperServer template is used when setting up a server for development use. This template will configure a JVM for a quick start by disabling bytecode verification, and performing JIT compilations with a lower optimization level. This option should not be used on a production server, where long run throughput is more important than early server startup.
- ▶ Custom template: You can create templates based on existing application servers (see “Creating an application server template (optional)” on page 281 for more details).

Ports

Each server process uses a set of ports that must be unique on the system. When you create an application server, you have the following options:

- ▶ Use the default ports:

Use this selection if you will only have one application server on the system or if this is the first application server created and port selection is not an issue.

- ▶ Have a set of ports selected that are unique to the WebSphere system installation:

This selection ensures that no two WebSphere processes in the installation will have the same port assigned. It does not guarantee that ports will be selected that are not in use by non-WebSphere processes, or by WebSphere processes installed as a separate installation.

- ▶ Specify the ports:

This option is best if you have a convention for port assignment on your system that ensures unique ports are used by all processes, both WebSphere and non-WebSphere.

This option is only available when you create a new profile using the Advanced option or the **manageprofiles** command.

If you find that you have a port conflict, you can change the ports after the application server is created.

z/OS settings

Next, we describe the various z/OS settings:

- ▶ Long name:

The long name of an application server is the name used in scripting and the administrative console. Long names can be up to 50 characters long and include mixed-case alphabetic characters, numeric characters, and the following special characters: ! ^ () _ - . { } [, and].

- ▶ Short name:

Short names are specific to the z/OS implementation of WebSphere Application Server and are the principal names by which cells, nodes, servers, and clusters are known to z/OS.

- ▶ Specific short name (z/OS):

The short name is also used as the JOBNAME for the server. If you do not specify a value for the short name field, the short name defaults to BBOSnnn, where nnn is the first free number in the cell that can be used to create a unique short name. Make sure that you set up a RACF SERVER class profile that includes this short name.

- ▶ Generic short name (z/OS):

Nodes that have not been clustered have a server generic short name, also called a *cluster transition name*. When a cluster is created from an existing application server, the server's generic short name becomes the cluster name.

No two servers on the same z/OS system should have the same server generic short name unless they are in the same cluster.

If you do not specify a value for the generic short name field, the generic short name defaults to BBOCnnn, where nnn is the first free number in the cell that can be used to create a unique generic short name.

- Bit mode (z/OS):

The default setting is that the application server runs in 64-bit mode, but you can elect to run in 31-bit mode. Note that 31-bit mode is deprecated.

Creating an application server from the administrative console

To create an application server from the administrative console, complete the following steps:

1. Open the deployment manager administrative console.
2. Click **Servers** → **Server Types** → **WebSphere application server**.
3. Click **New**.
4. Select the node for the new server and enter a name for the new server (Figure 7-4).

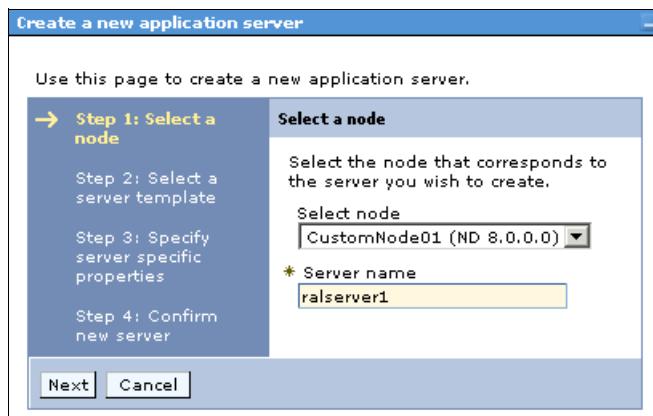


Figure 7-4 Create an application server - Select a node

Click **Next**.

5. Select a template to use by clicking the appropriate radio button (Figure 7-5).

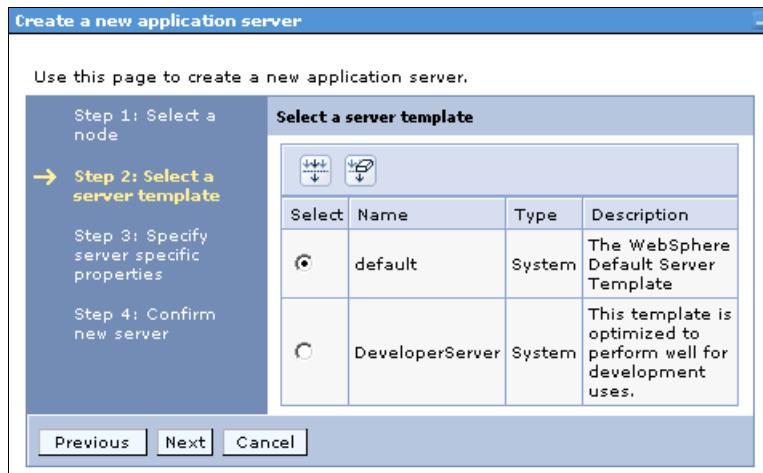


Figure 7-5 Create an application server - Select a template

On z/OS systems, there is one system defined template called defaultZOS.

Click **Next**.

6. The options you see on the next window vary depending on the platform. For distributed platforms, you see the window shown in Figure 7-6.

Select the **Generate Unique Ports** box to have unique ports generated for this server. Clearing this option will generate the default set of ports.

If you have multiple core groups defined, you have the option to select the core group.

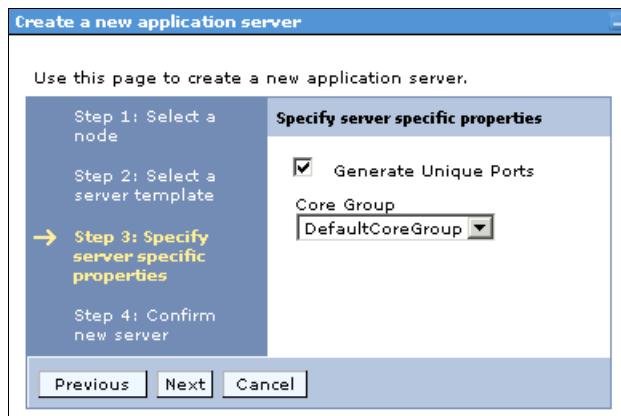


Figure 7-6 Create an application server - Generate unique ports

For z/OS systems, you see the window shown in Figure 7-7.

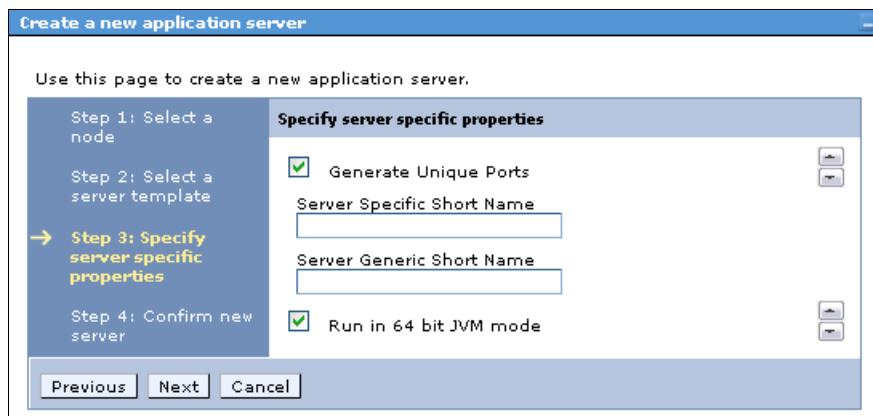


Figure 7-7 Create an application server - Generate unique ports z/OS

The server specific short name specifies the short name for the server. This item is also used as the job name (for example, BBOS002). The generic short name is the short name that is converted to a cluster short name if the server is later used in a cluster.

Click **Next**.

7. A summary window is presented with the options you chose. Click **Finish** to create the server.
8. In the messages box, click **Save** to save the changes to the master repository.
9. Review and update the virtual host settings (see "Updating the virtual host settings" on page 281 for more details).
10. Regenerate the web server plug-in and propagate it to the web server (see 12.5, "Working with the plug-in configuration file" on page 476 for more details).

Information about managing web server plug-ins is provided in the Information Center. See the topic “Communicating with web server” at the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=/com.ibm.websphere.nd.multiplatform.doc/info/ae/ae/twsv_plugin.html

Note: If you are creating an application server on a Windows operating system, this process does not give you the option of registering the new server as a Windows service. You can do this task later using the **WASService** command (see 7.9, “Enabling process restart on failure” on page 337 for more details)

Creating an application server from the job manager

To create an application server from the job manager, make the following selections as you step through the process to submit the job:

1. Start the job manager and targets. Access the job manager console.
2. Click **Jobs** → **Submit**.
3. Click the **Create application server** job type. Click **Next**. (Figure 7-8)

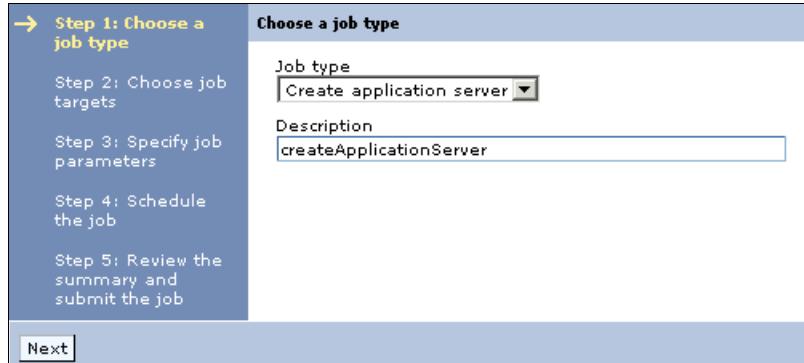


Figure 7-8 Create an application server

4. Select the job target:
 - If you are creating an application server on an unfederated node, select the application server node.
 - If you are creating a new managed server, select the deployment manager node.Add a target name by entering the name and clicking **Add**. You can also search for a target name by clicking **Find**.

Enter the user name and password with administrative authority on the target node, as shown in Figure 7-9.

The screenshot shows a configuration dialog titled "Choose job targets" for "Job type: Create application server". On the left, a vertical navigation bar lists steps: Step 1: Choose a job type, Step 2: Choose job targets (which is selected and highlighted in yellow), Step 3: Specify job parameters, Step 4: Schedule the job, and Step 5: Review the summary and submit the job. The main area contains two sections: "Target groups" (radio button selected) with a dropdown menu showing "-- No groups --" and "Target names" (radio button selected) with a list box containing "saw116-sys4Node01". Below this is a "Remove" button. A section titled "Target authentication" follows, with "User name" set to "wasadmin" and "Password authentication" selected. Under "Password authentication", fields for "Password" and "Confirm password" are filled with masked text. Other authentication options like "Public-private key authentication" and its fields are also present. At the bottom are "Previous", "Next" (highlighted in blue), and "Cancel" buttons.

Figure 7-9 Choose a job target

Click **Next**.

5. Specify the job parameters, as shown in Figure 7-10. At minimum:
- Specify a unique name of the new application server. You can use the **Find** button to get a list of existing server names on the target.
 - If the target is a deployment manager, enter the name of the node on which the server will be created.

The screenshot shows the 'Specify job parameters' dialog box. On the left, a vertical sidebar lists five steps: Step 1: Choose a job type, Step 2: Choose job targets, Step 3: Specify job parameters (which is highlighted with a yellow arrow), Step 4: Schedule the job, and Step 5: Review the summary and submit the job. The main area is titled 'Specify job parameters' and contains the following sections:

- Job type: Create application server**
- * Server name:** server2 (highlighted with a yellow background)
- Node name:** (empty input field)
- Find...** button (next to the server name field)
- Additional job parameters:** (checkbox, unchecked)
- Server template** section:
 - Template name:** (empty input field)
 - Template location:** (empty input field)
- Port control** section:
 - Generate unique ports:** (checkbox, checked)
- Platform specific** section:
 - Specific short name:** (empty input field)
 - Generic short name:** (empty input field)
 - Bit mode:** (dropdown menu, currently set to 64)

At the bottom of the dialog are three buttons: **Previous**, **Next**, and **Cancel**.

Figure 7-10 Specify the options for the new server

You can expand the Additional job parameters section where the optional settings allow you to add platform specific settings, specify a different template, and specify the setting that determines if ports unique to the installation are generated.

- The template name field will default to the default server template for the operating system on which the application server will run. You only need to specify this setting if you want to use a custom template or the DeveloperServer template.
- **Generate unique ports** is selected by default to generate unique ports for the installation.
- Platform specific information is where you can provide a short name, generic name, or bit mode for creating a server on the target. If you do not provide this information, the product generates unique names and uses the default bit mode.

Click **Next**.

6. Schedule the job.

Take the defaults for the job schedule. The defaults will execute the job once. Click **Next**, as shown in Figure 7-11.

The screenshot shows the 'Schedule the job' step of a wizard. On the left, a vertical sidebar lists steps: Step 1: Choose a job type, Step 2: Choose job targets, Step 3: Specify job parameters, Step 4: Schedule the job (highlighted in yellow), and Step 5: Review the summary and submit the job. The main panel has the following sections:

- Job type:** Create application server
- Notification:** Email addresses (text input field)
- Initial Availability:** Make the job available now (radio button selected). Date and time inputs are present.
- Expiration:** Use default expiration - 1 days (radio button selected). Options for date/time or duration are available.
- Job Availability Interval:** Run once (dropdown menu)

At the bottom are buttons: Previous, Next (highlighted in yellow), and Cancel.

Figure 7-11 Schedule the job

7. Review the settings and click **Finish**, as shown in Figure 7-12.

The screenshot shows the 'Review the summary and submit the job' step of a wizard. On the left, a vertical sidebar lists steps: Step 1: Choose a job type, Step 2: Choose job targets, Step 3: Specify job parameters, Step 4: Schedule the job, and Step 5: Review the summary and submit the job (highlighted in yellow). The main panel displays a summary table:

Summary of actions:	
Options	Values
Job type	Create application server
Description	createApplicationServer
Target names	saw116-sys4Node01
Initial availability	Make the job available now.
Expiration	Use the default expiration.
User name	wasadmin
Server name	server2
Generate unique ports	TRUE
Bit mode	64

At the bottom are buttons: Previous, Finish (highlighted in yellow), and Cancel.

Figure 7-12 Summary review

8. Verify the server was created. Click **Jobs** → **Target resources** to see the new server in the list of resources.

Updating the virtual host settings

When you install applications, you associate a virtual host with each web module.

When you create a new application server, the default_host virtual host is set as the default virtual host for web modules installed on the server. You can change this default in the web container settings for the application server or simply select a new virtual host when you install the applications.

If the application will only be accessed through a web server, and the virtual host that you will use has been set up with the web server port in the list of host aliases, then no action is necessary.

However, if application clients will access the web container directly, or if you will be installing SIP applications on this server, you need to ensure that the relevant ports generated for this application server are added to the host alias list. See 7.7, “Working with virtual hosts” on page 321 for more information.

Creating an application server template (optional)

WebSphere Application Server provides the ability to create a customized server template that is based on an existing server configuration. Then, you can use that server template to create new servers. If you are going to need more than one application server, for example, for a cluster, and if the characteristics of the server are different from the default server template, it is more efficient to create a custom template and use that template to create your server.

To create an application server template based on an existing server, complete the following steps:

1. Click **Servers** → **Server Types** → **WebSphere application servers**.
2. Click **Templates...** at the top of the server list.
3. Click **New**.
4. Select a server from the list to build the template from and click **OK**.
5. Enter a name and description for the template and click **OK**.
6. Save your configuration.

The new template will be in the list of templates and will be available to select, the next time you create an application server, as shown in Figure 7-13.

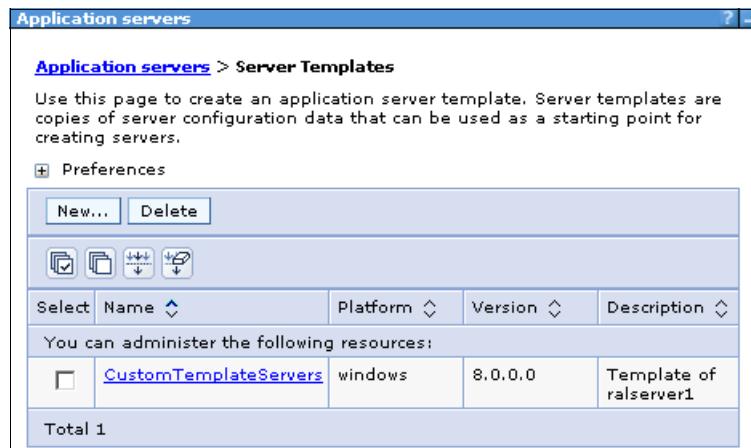


Figure 7-13 Server template listing

7.4.2 Viewing the status of an application server

There are multiple ways to check the status of an application server:

- ▶ Use the **serverStatus** command on the system where the application server is running.
- ▶ In a distributed environment, you can view the status from the administrative console. The node for the application server must be active for the deployment manager to know the status of a server on that node.
- ▶ From the job manager console.
- ▶ If the server is registered as a Windows service, you can check the status of the service.

Using the administrative console

To check the status of a managed server using the deployment manager's administrative console, the node agent must be started. To use the administrative console, complete the following steps:

1. Click **Servers** → **Server Types** → **WebSphere application servers**.
2. The servers are listed. The last column to the right contains an icon to indicate the status of each server, as shown in Figure 7-14.

The screenshot shows the 'Application servers' page in the WebSphere Administrative Console. At the top, there is a message: 'Use this page to view a list of the application servers in your environment and the status of each of these servers. You can also use this page to change the status of a specific application server.' Below this are several buttons: New..., Delete, Templates..., Start, Stop, Restart, ImmediateStop, and Terminate. There are also icons for creating, deleting, and managing resources. A table follows, with columns for Select, Name, Node, Host Name, Version, Cluster Name, and Status. The 'Status' column header is highlighted with a red box. Two rows of data are shown, each with a checkbox, a server name ('ralserver1' and 'ralserver2'), a node name ('CustomNode01'), a host name ('saw116-sys4.itso.ral.ibm.com'), a version ('ND 8.0.0.0'), and a cluster name (''). The 'Status' column for both rows contains a red X icon, which is also highlighted with a red box. At the bottom of the table, it says 'Total 2'.

Figure 7-14 Administrative console server status

Figure 7-15 shows the icons and their corresponding status.

	Started	The server is running.
	Partially started	The server is in the process of changing from a stopped state to a started state.
	Partially stopped	The server is in the process of changing from a started state to a stopped state.
	Stopped	The server is not running.
	Unknown	The server status cannot be determined.

Figure 7-15 Status icons

Note: If the server status is Unknown, the node agent on the node in which the application server is installed is not active. The server cannot be managed from the administrative console unless its node agent is active.

Using the serverStatus command

The syntax of the **serverStatus** command is as follows:

```
serverStatus.bat(sh) [options]
```

The options are shown Example 7-5.

Example 7-5 serverStatus options

```
Usage: serverStatus <server name | -all>
[-logfile <filename>]
[-replacetool]
[-trace]
[-username <username>]
[-password <password>]
[-profileName <profile>]
[-help]
```

The first argument is mandatory. The argument is either the name of the server for which status is desired, or the **-all** keyword, which requests status for all servers defined on the node.

If you have administrative security enabled, you will need to enter a user ID and password of an administrator ID. If you do not include it in the command, you will be prompted for it.

For example, to view the status of a server, run the following command:

```
cd profile_home/bin
serverStatus.sh server_name -username adminID -password adminpw
```

To check the status of all servers on the node, run the following command:

```
cd profile_home/bin
serverStatus.sh -all -username adminID -password adminpw
```

For more information about the **serverStatus** command, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=/com.ibm.websphere.nd.multiplatform.doc/info/ae/ae/rxml_serverstatus.html

Example 7-6 shows an example of using the **serverStatus** command.

Example 7-6 serverStatus example - Windows operating system

```
/opt/IBM/WebSphere/AppServer/profiles/Custom01/bin>./serverstatus.sh -all
-username admin -password adminpw
ADMU0116I: Tool information is being logged in file
/opt/IBM/WebSphere/AppServer/profiles/Custom01/logs/serverStatus.log
ADMU0128I: Starting tool with the Custom01 profile
ADMU0503I: Retrieving server status for all servers
ADMU0505I: Servers found in configuration:
ADMU0506I: Server name: nodeagent
ADMU0506I: Server name: ralserver1
ADMU0508I: The Node Agent "nodeagent" is STARTED
ADMU0509I: The Application Server "ralserver1" cannot be reached. It appears to be
stopped.
```

From the job manager console

To display the servers and their status from the job manager console, complete the following steps:

1. Click **Jobs → Targets**.
2. Select the box to the left of the node name. In the **Display resources** drop-down menu, click **Server**, as shown in Figure 7-16.

The screenshot shows a table with columns for 'Select', 'Target name', and 'Version'. There are two rows: one for 'TestDmgr01' (unchecked) and one for 'saw116-sys4Node01' (checked). A dropdown menu at the top right labeled 'Display Resources' has 'Server' selected. Buttons for 'New Host...', 'Delete Host', and other actions are also visible.

Select	Target name	Version
<input type="checkbox"/>	TestDmgr01	ND 8.0.0.0
<input checked="" type="checkbox"/>	saw116-sys4Node01	ND 8.0.0.0
Total 2		

Figure 7-16 Display the servers on a node

3. This action displays the list of servers in the node. Click the name of the server, as shown in Figure 7-17.

The screenshot shows a table with columns for 'Resources', 'Quantity', and 'Target name'. There are two rows: one for 'server/server1' (Quantity 1, Target name saw116-sys4Node01) and one for 'server/server2' (Quantity 1, Target name saw116-sys4Node01). A dropdown menu at the top right has 'Resources' selected. Buttons for 'New Host...', 'Delete Host', and other actions are also visible.

Resources	Quantity	Target name
server/server1	1	saw116-sys4Node01
server/server2	1	saw116-sys4Node01
Total 2		

Figure 7-17 Display the servers on a node

The status of the server is displayed, as shown in Figure 7-18.

The screenshot shows a table with columns for 'Resource ID', 'Target name', and 'Status'. There is one row for 'saw116-sys4Node01/server/server2' (Target name saw116-sys4Node01, Status Stopped). A header bar indicates the path: Targets > Target resources > Target Resource. A note says 'The following resources belong to the target being viewed.' A '+' button for Preferences is visible.

Resource ID	Target name	Status
saw116-sys4Node01/server/server2	saw116-sys4Node01	Stopped
Total 1		

Figure 7-18 Display the servers on a node

7.4.3 Starting an application server

How you start an application server depends largely on personal preference and on whether the application server is stand-alone or managed. This section provides information about how to start individual application servers. You can also start all application servers in a cluster by starting the cluster (see 7.6.3, “Managing clusters” on page 320 for more details).

Using the administrative console to start a managed server

Tip: Before managing a server in a distributed server environment using the administrative console, the node agent for the server's node must be running. To check the status of the node, click **System administration** → **Node Agents**. The status of the node agent is in the far right column. If it is not started, you must start it manually (see 7.5.1, “Starting and stopping nodes” on page 301 for more details).

From the administrative console, complete the following steps:

1. Click **Servers** → **Server Types** → **WebSphere application servers**.
2. Select the box to the left of each server you want to start.
3. Click **Start**.
4. Verify the results in the Server status feedback window.

If there are any errors, check the log files for the application server process. The default location for the logs is:

- ▶ *profile_home/logs/server_name/SystemOut.log*
- ▶ *profile_home/logs/server_name/startServer.log*

On z/OS, check the output in the application server job log.

Tip: By default, all the applications on a server start when the application server starts. To prevent an application from starting, see 7.8.6, “Preventing an enterprise application from starting on a server” on page 331 for more details.

Using the startServer command

The syntax of the **startServer** command is shown in Example 7-7.

Example 7-7 startServer options

```
Usage: startServer <server> [options]
      options: -nowait
                -quiet
                -logfile <filename>
                -replaceLog
                -trace
                -script [<script filename>] [-background]
                -timeout <seconds>
                -statusPort <portnumber>
                -profileName <profile>
                -recovery
                -help
```

<server> is the name of the server to be started. The first argument is mandatory and case sensitive.

For more information about the **startServer** command, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=/com.ibm.websphere.nd.multiplatform.doc/info/ae/ae/rxml_startserver.html

startServer example

Example 7-8 shows an example of using the **startServer** command. Note that the user ID and password are not required to start the server.

Example 7-8 startServer example

```
/opt/IBM/WebSphere/AppServer/profiles/Custom01/bin>./startserver.sh ralserver1
ADMU0116I: Tool information is being logged in file
/opt/IBM/WebSphere/AppServer/profiles/Custom01/logs/server40a1/startServer
.log
ADMU0128I: Starting tool with the Custom01 profile
ADMU3100I: Reading configuration for server: ralserver1
ADMU3200I: Server launched. Waiting for initialization status.
ADMU3000I: Server ralserver1 open for e-business; process id is 3928
```

Starting a server from the job manager

To start an application server from the job manager, complete the following steps:

1. Access the job manager console.
2. Click **Jobs** → **Submit**.
3. Select the **Start server** job type. Click **Next**, as shown in Figure 7-19.

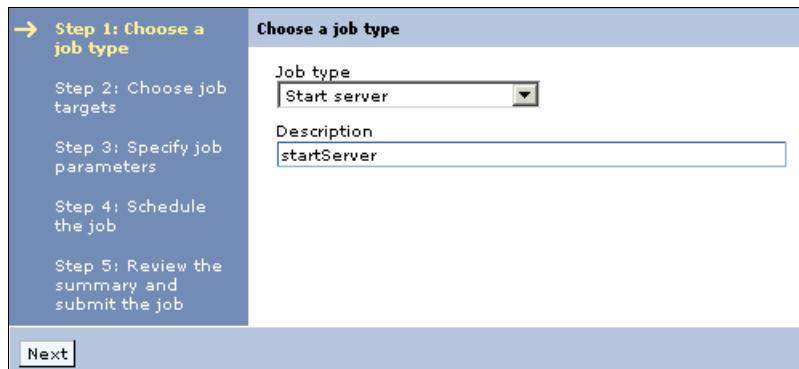


Figure 7-19 Select a job type

4. Select the job target:
 - If you are starting an application server on an unfederated node, select the application server node.
 - If you are starting a new managed server, select the deployment manager node.Add a target name by entering the name and clicking **Add**. You can also search for a target name by clicking **Find**.

Enter the user ID and password with administrative authority on the target node, as shown in Figure 7-20.

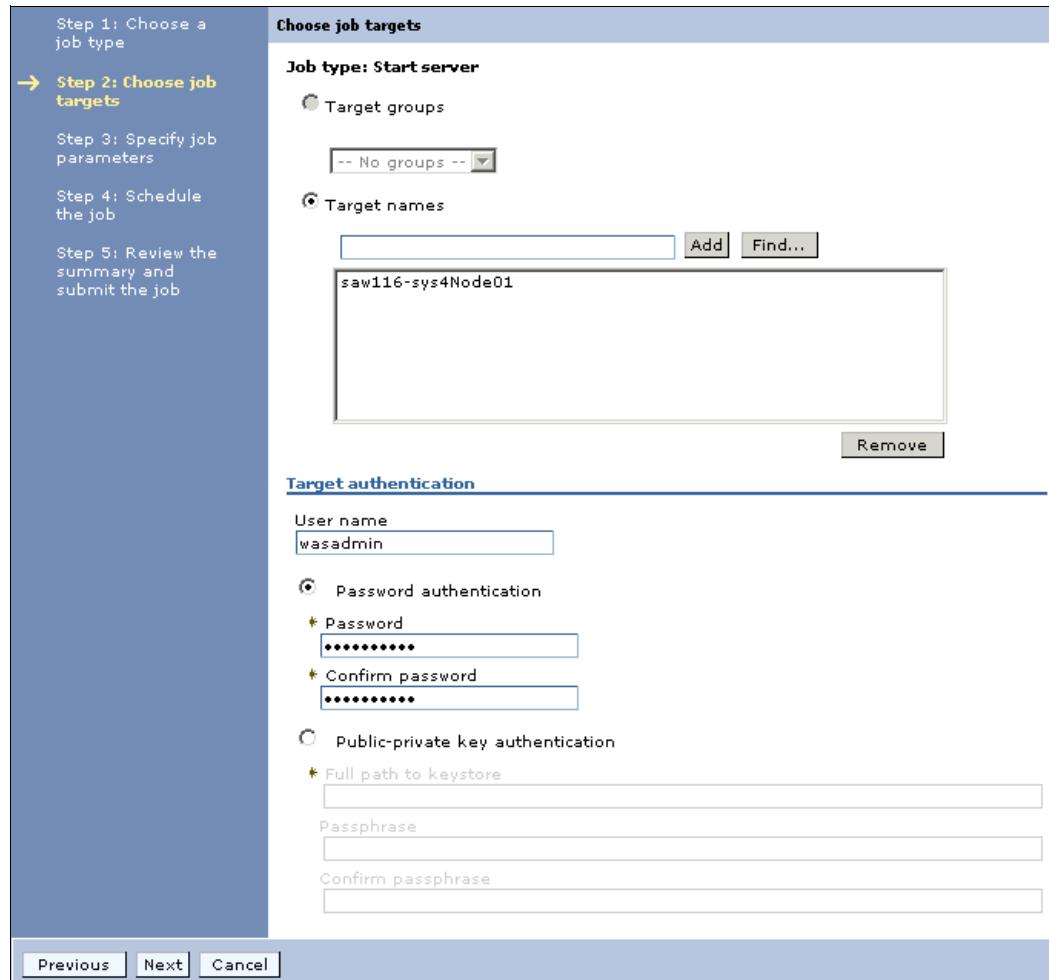


Figure 7-20 Select a target

Click **Next**.

5. Specify the job parameters.
 - Specify the name of the application server. You can use the **Find** button to get a list of existing server names on the target.
 - If the target is a deployment manager, enter the name of the node on which the server will be created.

Click **Next**, as shown in Figure 7-21.

The screenshot shows the 'Specify job parameters' step of a five-step wizard. On the left, a vertical sidebar lists steps: Step 1: Choose a job type; Step 2: Choose job targets; Step 3: Specify job parameters (highlighted in yellow); Step 4: Schedule the job; Step 5: Review the summary and submit the job. The main panel has a title 'Specify job parameters' and a sub-section 'Job type: Start server'. It contains two input fields: 'Server name' (containing 'server1') and 'Node name'. A 'Find...' button is next to the server name field. At the bottom are 'Previous', 'Next', and 'Cancel' buttons.

Figure 7-21 Select the job parameters

6. Schedule the job.

Take the defaults for the job schedule. The defaults will execute the job once. Click **Next**, as shown in Figure 7-22.

The screenshot shows the 'Schedule the job' step of the wizard. The sidebar on the left shows steps 1-5. The main panel has a title 'Schedule the job' and a sub-section 'Job type: Start server'. It includes sections for 'Notification' (Email addresses), 'Initial Availability' (radio buttons for 'Make the job available now.' or 'Schedule availability' with date/time inputs), 'Expiration' (radio buttons for 'Use default expiration - 1 days.', 'Expire the job based on a date' with date/time inputs, or 'Expire the job based on a duration' with an 'Expire after' time input), and 'Job Availability Interval' (a dropdown menu set to 'Run once'). At the bottom are 'Previous', 'Next', and 'Cancel' buttons.

Figure 7-22 Schedule the job

7. Review the summary and click **Finish**, as shown in Figure 7-23. Monitor the status of the job and ensure it completes successfully.

Options	Values
Job type	Start server
Description	startServer
Target names	saw116-sys4Node01
Initial availability	Make the job available now.
Expiration	Use the default expiration.
User name	wasadmin
Server name	server1

Figure 7-23 Summary review

7.4.4 Stopping an application server

This section shows multiple methods for stopping a server.

Using the administrative console to stop a managed server

Note: These directions assume that the node agent for the application server is running.

From the administrative console, you have the following options to stop an application server (Figure 7-24):

- ▶ The Stop button quiesces the application server and stops it. In-flight requests are allowed to complete.
- ▶ The Restart button stops and then starts the server.
- ▶ The Immediate Stop button stops the server, but bypasses the normal server quiesce process, which enables in-flight requests to complete before shutting down the entire server process. This shutdown mode is faster than the normal server stop processing, but some application clients can receive exceptions.
- ▶ The **Terminate** button deletes the application server process. Use this if immediate stop fails to stop the server.

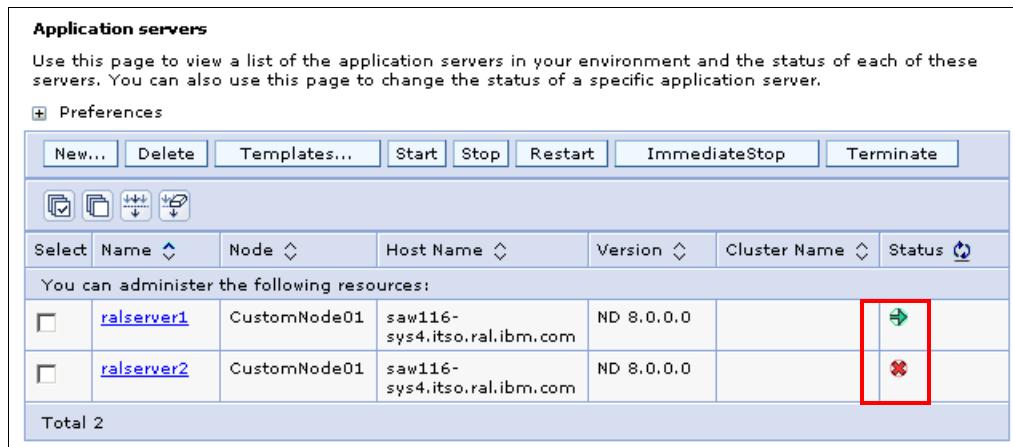


Figure 7-24 Administrative console buttons

From the administrative console, complete the following steps to stop an application server:

1. Click **Servers** → **Server Types** → **WebSphere application servers**.
2. Select the box to the left of each server you want to stop.
3. Click the appropriate stop option.

If there are any errors, check the log files for the application server process:

- ▶ *profile_home/logs/server_name/SystemOut.log*
- ▶ *profile_home/logs/server_name/stopServer.log*

On z/OS, check the output in the application server job log.

Restarting all servers on a node

If you want to stop, and then restart, all the application servers on a node, complete the following steps from the administrative console:

1. Click **System administration** → **Node agents**.
2. Select the box to the left of the node agent.
3. Click **Restart all Servers on Node**.

Stopping all servers in a cluster

If you want to stop all the servers in a cluster, complete the following steps from the administrative console:

1. Click **Servers** → **Clusters** → **WebSphere application server clusters**.
2. Select the box to the left of the cluster.
3. Click **Stop** or **Immediate Stop**.

Restarting all servers in a cluster

If you want to stop, and then restart, all the servers in a cluster, complete the following steps from the administrative console:

1. Click **Servers** → **Clusters** → **WebSphere application server clusters**.
2. Select the box to the left of the cluster.
3. Click **Ripplestart**.

Using the **stopServer** command

The syntax of the **stopServer** command is:

```
stopServer.bat(sh) [options]
```

The options are shown in Example 7-9.

Example 7-9 stopServer command

```
Usage: stopServer <server> [options]
      options: -nowait
                -quiet
                -logfile <filename>
                -replacelog
                -trace
                -timeout <seconds>
                -statusport <portnumber>
                -conntype <connector type>
                -port <portnumber>
                -username <username>
                -password <password>
                -profileName <profile>
                -help
```

<server> is the name of the server to be started. The first argument is mandatory and is case sensitive. If administrative security is enabled, a user name and password is also required.

For more information about the **stopServer** command, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=/com.ibm.websphere.nd.multiplatform.doc/info/ae/ae/rxml_stopserver.html

Example 7-10 shows an example of the **stopServer** command.

Example 7-10 stopServer command example

```
/opt/IBM/WebSphere/AppServer/profiles/Custom01/bin>./stopserver.sh ralserver1
-username admin -password adminpw
ADMU0116I: Tool information is being logged in file
/opt/IBM/WebSphere/AppServer/profiles/Custom01/logs/ralserver1/stopServer.
log
```

ADMU0128I: Starting tool with the Custom01 profile
ADMU3100I: Reading configuration for server: ralserver1
ADMU3201I: Server stop request issued. Waiting for stop status.
ADMU4000I: Server ralserver1 stop completed.

Note: If you attempt to stop a server and for some reason it does not stop, you can use the operating system commands to stop the Java process for the server.

7.4.5 Viewing runtime attributes of an application server

To view runtime attributes using the administrative console, complete the following steps:

1. Click **Servers** → **Server Types** → **WebSphere application servers** to display the list of servers.
2. Click the server name to access the detail page.
3. If the server is running, you will see both a Configuration tab and Runtime tab. If the servers is not running, you will see only a Configuration tab. Click the **Runtime** tab. Figure 7-25 shows the Runtime tab and the information that it provides.

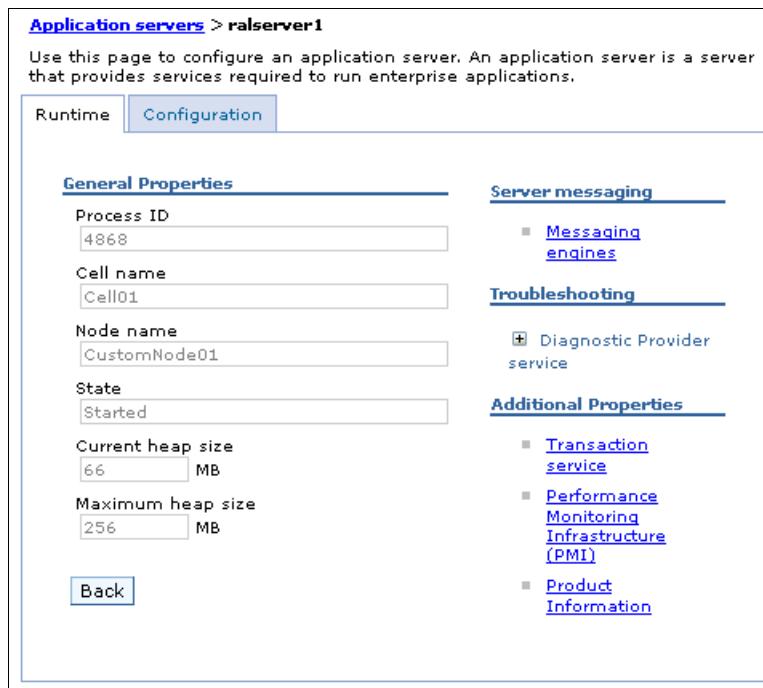


Figure 7-25 Application server Runtime tab

On the Runtime tab, you have:

- General information about the server process, including the process ID, cell name, node name, state, and the current and maximum heap size.
- Access to a list of messaging engines that run on this application server. There will be one messaging engine for each bus of which the server is a member. You can start and stop the messaging engine from this window.
- Access to the Diagnostic Provider service, allowing you to query current configuration data, state, and to initiate diagnostic tests.

- Access to the Transaction Service properties settings. You can change the timeout settings while the server is running, but not the transaction log directory setting. Changes made in this window are active until the server is stopped.

To make these settings permanent and to configure additional Transaction Service settings, click the **Configuration** tab while you have this page open. This action takes you directly to the Transaction Service settings.

You can also view or act on transactions in the following states by clicking **Review** to the right of the state. This action is not normally necessary, but in an exceptional situation, it might be useful.

- Manual transactions:

These transactions await administrative completion. For each transaction, the local or global ID is displayed. You can display each transaction resource and its associated resource manager. You can choose also to commit or rollback transactions in this state.

- Retry transactions:

These are transactions with some resources being retried. For each transaction, the local or global ID is displayed, and whether the transaction is committing or rolling back. You can display each transaction resource and its associated resource manager. You can choose also to finish, or abandon retrying, transactions in this state.

- Heuristic transactions:

These are transactions that have completed heuristically. For each transaction, the local or global ID and the heuristic outcome is displayed. You can display each transaction resource and its associated resource manager. You can also choose to clear the transaction from the list.

- Imported prepared transactions:

Transactions that have been imported and prepared but not yet committed. For each transaction, the local or global ID is displayed. You can display each transaction resource and its associated resource manager. You can also choose to commit or roll back transactions in this state.

- Performance Monitoring Service settings allow you to change the instrumentation levels while the server is running or make permanent changes to the configuration for that server.
- Product Information gives you access to extensive information about the product installation and Fix Pack information.

7.4.6 Customizing application servers

When you create a new application server, it inherits most of its configuration settings from the specified template server. To view or modify these settings, click **Servers** → **Server Types** → **WebSphere application servers**. A list of application servers defined in the cell appears in the workspace. Click the name of the application server to make a modification.

This section gives you a quick overview of the types of settings that you can customize. See Figure 7-26 for a list of most of the settings (not all settings are shown due to the size of the configuration window).

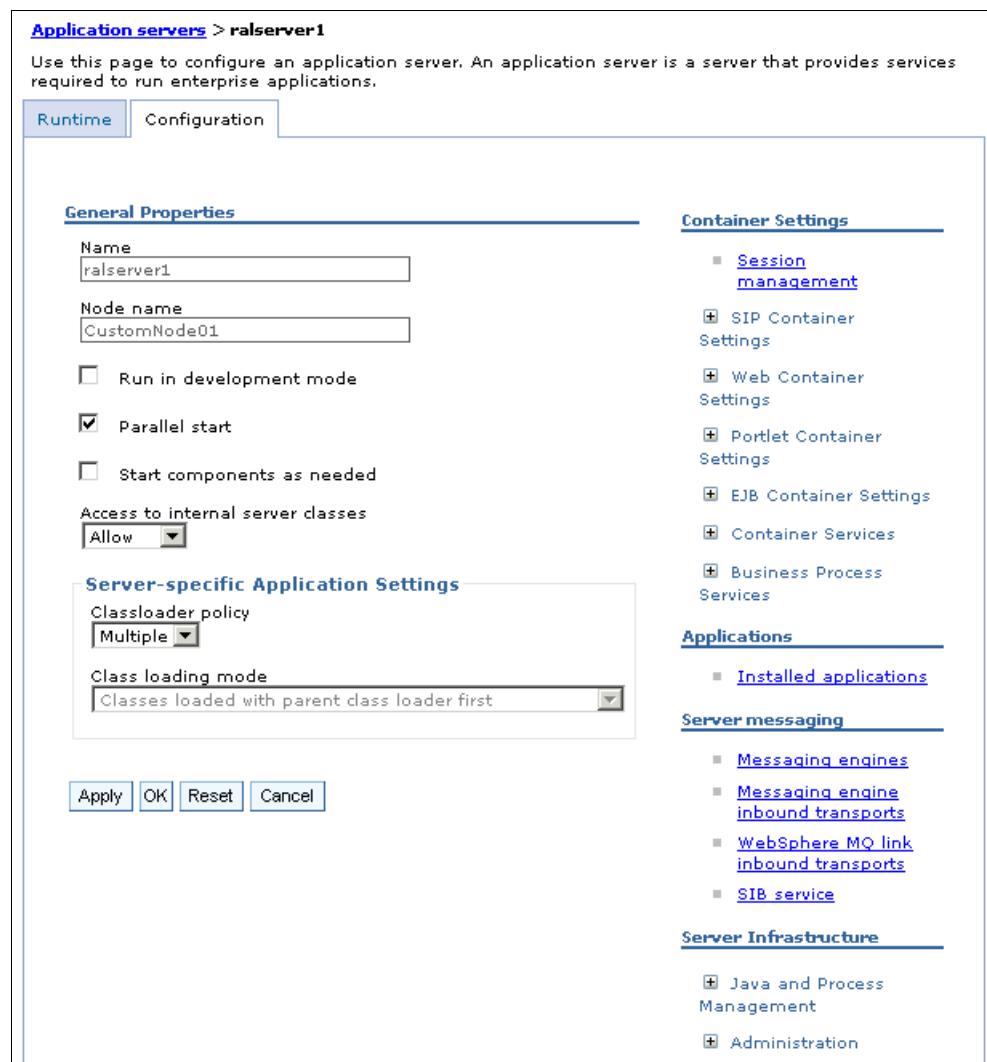


Figure 7-26 Application server configuration

General properties

The general properties consist of a few items that you can see immediately:

- Server name and node name is defined.
You also have short name and unique ID for servers on z/OS.
- Run in development mode: Enable this option to streamline the startup time of an application server. Do not enable this setting on production servers.
- Parallel start: Select this field to start the server components, services, and applications on multiple threads. This action might shorten the startup time.

The order in which the applications start depends on the weights you assigned to each of them. Applications that have the same weight are started in parallel.

To set the weight of an application, in the administrative console, click **Applications** → **Application Types** → **WebSphere enterprise applications** → **application_name** → **Startup behavior**, and then specify an appropriate value in the **Startup order** field.

- ▶ Start components as needed: Select this field to start the application server components as they are needed. This action might shorten the start time.
- ▶ Access to internal server classes: Specifies whether the applications can access many of the server implementation classes.
- ▶ Application classloader policy and class loading mode: These settings allow you to define an application server-specific classloader policy and class loading mode. Information about class loaders is provided in Chapter 20, “Understanding class loaders” on page 747.

Note: You also have the run in 64 bit JVM mode setting on z/OS, which provides additional virtual storage for user applications in 64-bit mode. Removing the check from this selection enables your server to start in 31-bit mode.

There is no interdependence between the modes in which you are running different servers. You can run some of your servers in 64-bit mode and some of your servers in 31-bit mode. However, you should eventually convert all of your servers to run in 64-bit mode because support for running servers in 31-bit mode is deprecated.

Container settings

Each application server has containers that run specific application components. This section in the configuration page for the server provides links to pages where you can modify the settings for the containers.

Tip: Modifying container settings is not something you would normally do on a daily basis. Information about the most commonly used settings in these sections is provided throughout this book in the appropriate topics. For now, it is enough to know that each container has settings that allow you to modify its configuration and how to find those settings.

Session management

The session management settings allow you to manage HTTP session support, which includes specifying a session tracking mechanism, setting maximum in-memory session count, controlling overview, and configuring session timeout. Information about session management is provided in Chapter 25, “Session management” on page 889.

SIP container settings

Session Initiation Protocol (SIP) support extends the application server to allow it to run SIP applications written to the JSR 116 specification. SIP is used to establish, modify, and terminate multimedia IP sessions, including IP telephony, presence, and instant messaging. If you have SIP applications, review these settings.

Web container settings

The web container serves application requests for servlets and JSPs. The web container settings allow you to specify the default virtual host, enable servlet caching, disable servlet request and response caching, number of thread timeouts and the default timeout, settings for using thread pools or a work manager to start runnable objects, specify session manager settings such as persistence and tuning parameters, and HTTP transport properties.

The settings are shown in Figure 7-27.

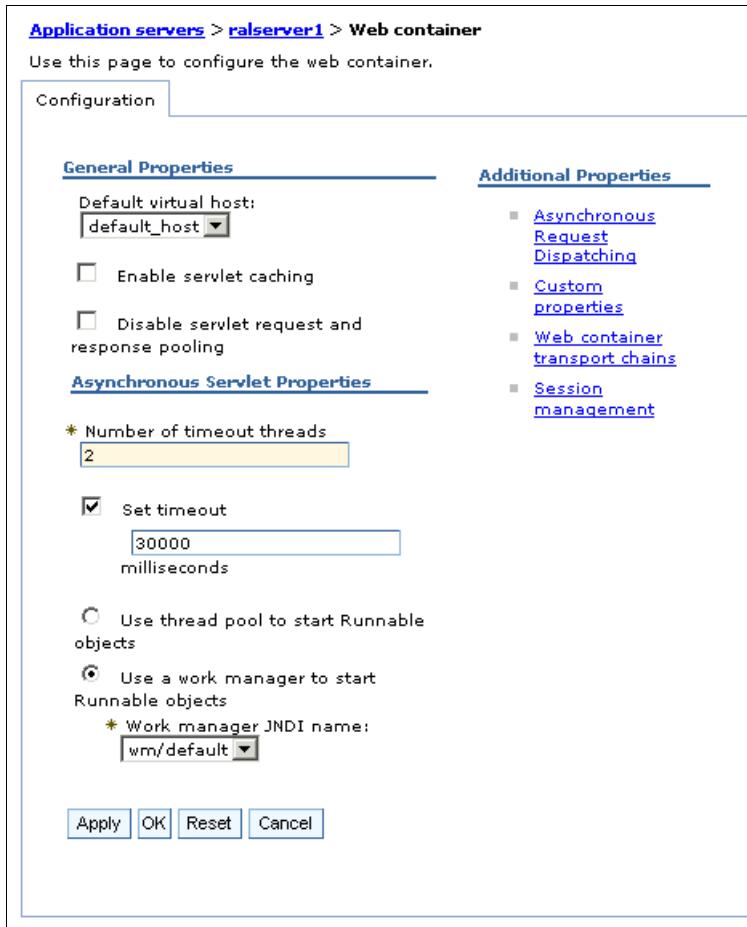


Figure 7-27 Web container settings

The Asynchronous Request Dispatcher (ARD) enables servlets and JSPs to make standard include calls concurrently on separate threads. Selecting this link allows you to enable ARD and configure related settings.

Portlet container services

The portlet container is the runtime environment for portlets using the JSR 168 Portlet Specification. Portlets based on this JSR 168 Portlet Specification are referred to as standard portlets. You can use these settings to enable portlet fragment caching to save the output of portlets to the dynamic cache.

EJB container properties

These properties allow you configure the services provided by the EJB container, which includes setting the passivation directory path, EJB cache and timer service settings, pool cleanup interval, a default data source JNDI name, and to enable stateful session bean failover using memory-to-memory replication.

Figure 7-28 shows the EJB container properties window.

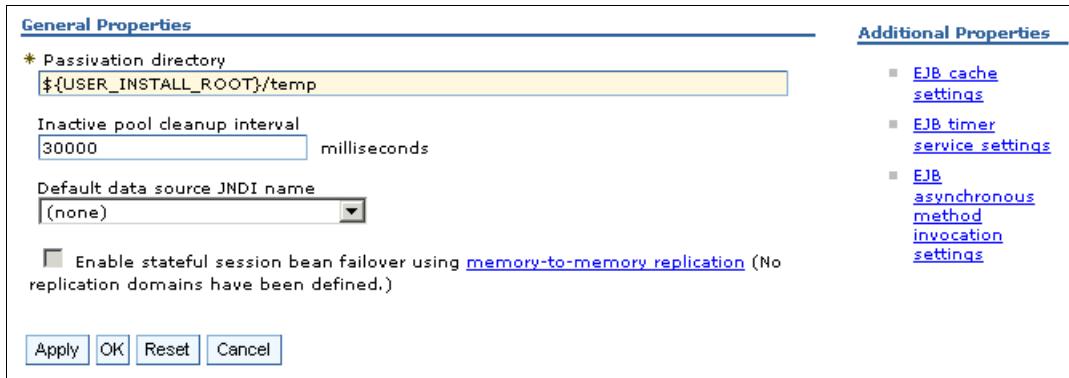


Figure 7-28 EJB container settings

Container services

The following settings are available under the container services section:

- ▶ Application profiling service: Application profiling is a WebSphere extension that, when used along with access intents, allows you to define strategies to dynamically control concurrency, prefetch, and read-ahead. The container services settings allows you to enable this service and to set the compatibility mode for J2EE 1.3 applications.
- ▶ Transaction service: The transaction service properties allow you to specify settings for the transaction service, as well as manage active transaction locks. The settings include the directory location for the transaction service on the application server to store log files for recovery, the total transaction lifetime timeout, and client inactivity timeout.
- When the application server is running, a Runtime tab is available in the Transaction Service properties workspace. From here, you can manage running transactions and modify timeout settings at run time.
- ▶ Dynamic cache service: This page allows you to specify settings for the dynamic cache service of this server.
- ▶ Compensation service: The compensation service supports server-level configuration for compensation enablement and logging. This service is not started automatically. If you will be running applications that require this service, you must enable it here.
- ▶ Internationalization service: This service enables you to configure and manage an internationalization context for an application for which components are distributed across the enterprise. This section of the configuration window allows you to enable this service. It is not enabled by default.
- ▶ The Default Java Persistence API (JPA) default settings: JPA provides a mechanism for managing persistence and object-relational mapping and functions for the EJB 3.0 specifications. This page allows you to configure default settings for JPA. JPA settings in an application will override these settings.
- ▶ Object pool service: The object pool service manages object pool resources that are used by the application server. This section of the configuration window allows you to disable this service (it is enabled by default).
- ▶ ORB service settings: These settings allow you to specify settings for the Object Request Broker service. These include request timeout, thread settings, and connection cache minimum and maximum.

- ▶ Startup beans service: Startup beans are session beans that run business logic through the invocation of start and stop methods when applications start and stop. This section of the configuration window allows you to enable this service (it is disabled by default).

Business process services

The business process settings allow you to manage the following features:

- ▶ Activity session service
- ▶ Work area partition service
- ▶ Work area service

Applications

Use the **Installed Applications** link to view the applications installed on this server. This link will display the collection of applications as links to the configuration page for each application.

Server messaging

The server messaging settings provide configuration settings and information for the messaging services.

Server infrastructure

The server infrastructure settings include settings for Java and process management and administration services:

- ▶ Java and Process Management:
 - Class loader: Creates and configures class loader instances. Information about class loaders is provided in Chapter 20, “Understanding class loaders” on page 747.
 - Process definition: Defines runtime properties, such as the program to run, arguments to run the program, and the working directory. Within the process definitions, you will find the JVM definitions, such as the initial and maximum heap sizes, debug options, the process class path, or different runtime options, such as profiler support and heap size.
 - Process execution: Includes settings such as the process priority, or the user and group that should be used to run the process. These settings are not applicable on the Windows platform.
 - Monitoring policy: Determines how the node agent will monitor the application server. It includes ping intervals, timeouts, and an initial state setting. These settings can be used to ensure that the server is started when the node starts and will be restarted in the event of a failure.
- ▶ Administration:
 - Custom properties: Specifies additional custom properties for this component.
 - Administration services: This group of settings allows you to specify various settings for administration facility for this server, such as administrative communication protocol settings and timeouts. (These settings are not something with which you would normally be concerned.)
 - Server components: Creates additional runtime components that are configurable.
 - Custom services: Creates custom service classes that run within this server and their configuration properties.

If you plan to extend the administration services by adding custom MBeans, refer to the topic “Extending WebSphere Application Server Administrative System with custom MBeans” in the Information Center at the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.nd.doc/info/ae/ae/tjmx_extend.html

Communications

The following communications settings are available:

- ▶ Ports:

These settings contain the basic port definitions for the server, which are shown in Figure 7-29.

Tip: You might not ever need to manually change these ports. It is likely, however, that you will want to view them.

For example, if you use the **dumpNameSpace** command, you can specify the bootstrap port of the process from which to dump the name space. When you federate a node, you will need to know the SOAP connector port of the node or deployment manager. The inbound communications ports are essential for accessing applications and the administrative console.

Clicking **Details** takes you to a page that has links for the configurable port settings.

Some port settings will be defined to use the channel framework. These settings will have an associated transport chain that represents the network protocol stack. A transport chain consists of one or more types of channels, each of which supports a different type of I/O protocol, such as TCP or HTTP. Network ports can be shared among all of the channels within a chain. The Channel Framework function automatically distributes a request arriving on that port to the correct I/O protocol channel for processing.

Ports		
Port Name	Port	Details
BOOTSTRAP_ADDRESS	9813	
SOAP_CONNECTOR_ADDRESS	8880	
ORB_LISTENER_ADDRESS	9100	
SAS_SSL_SERVERAUTH_LISTENER_ADDRESS	9401	
CSIV2_SSL_SERVERAUTH_LISTENER_ADDRESS	9403	
CSIV2_SSL_MUTUALAUTH_LISTENER_ADDRESS	9402	
WC_adminhost	9060	
WC_defaulthost	9080	
DCS_UNICAST_ADDRESS	9353	
WC_adminhost_secure	9043	
WC_defaulthost_secure	9443	
SIP_DEFAULTHOST	5060	
SIP_DEFAULTHOST_SECURE	5061	
SIB_ENDPOINT_ADDRESS	7276	
SIB_ENDPOINT_SECURE_ADDRESS	7286	
SIB_MQ_ENDPOINT_ADDRESS	5558	
SIB_MQ_ENDPOINT_SECURE_ADDRESS	5578	
IPC_CONNECTOR_ADDRESS	9637	

Figure 7-29 Viewing application server ports

- ▶ Message listener service:

The message listener service provides support for WebSphere Application Server V5 message-driven beans applications.

- ▶ Communications Enabled Applications (CEA):

The Communication Enabled Applications setting provides the ability to add dynamic web communications to any application or business process. You can enable this setting and then configure the Representational State Transfer (REST) interface and the computer-telephony interaction (CTI) gateway.

Performance

These settings allow you to specify settings for the Performance Monitoring Infrastructure (PMI) and the Performance and Diagnostic Advisor Configuration framework. These performance monitoring settings are covered in Chapter 15, “Monitoring distributed systems” on page 541.

Security

Security settings for the application server allow you to set specific settings at the server level. Security settings are covered in *WebSphere Application Server V7.0 Security Guide*, SG24-7660.

Troubleshooting

These settings include the ones for logging and tracing. For information about troubleshooting and using these settings, see *WebSphere Application Server V6: Diagnostic Data*, REDP-4085.

Additional properties

The following settings are defined under the additional properties section:

- ▶ Class loader viewer service: This service is used to enable or disable the service that keeps track of classes that are loaded.
- ▶ Core group service: These settings are related to high availability.
- ▶ Endpoint listeners: An endpoint listener receives requests from service requester applications within a specific application server or cluster.
- ▶ Debugging service: On this page, you can specify settings for the debugging service, to be used in conjunction with a workspace debugging client application, for example, the Application Server Toolkit.
- ▶ Thread pools: The thread pool specifies the possible maximum number of concurrently running threads in the web container. As one thread is needed for every client request, this setting directly relates to the number of active clients that can possibly access the web container on this application server at any given time. A timeout value can be specified for the application server to remove threads from the pool based on a timed period of inactivity.

Finally, an option for creating threads beyond the maximum pool size is available. Be careful when using this option. It can have the unexpected effect of allowing the web container to create more threads than the JVM might be able to process, creating a resource shortage and bringing the application server to a halt.

- ▶ Reliable messaging state: This link allows you to view and manage the WS-ReliableMessaging runtime state.
- ▶ Web server plug-in properties: This setting is used to change the HTTP plug-in configuration without having to stop the server and start it again.

7.5 Working with nodes in a distributed environment

Managing nodes is a concept specific to a Network Deployment environment. Nodes are managed by the deployment manager through a process known as a *node agent* that resides on each node. To manage a node in a Network Deployment environment, the node must be defined and the node agent on each WebSphere Application Server node must be started.

Nodes are created when you create a profile. Nodes are added to a cell through federation (see 3.3.7, “Federating nodes to a cell” on page 112 for more details).

7.5.1 Starting and stopping nodes

A node consists of the node agent and the servers. There are several ways to start and stop a node and node agent, or stop them individually. Before using any of these methods, be sure to note whether it affects the entire node, including servers, or just the node agent.

Starting a node agent

When a node agent is stopped, the deployment manager has no way to communicate with it. Therefore, the node agent has to be started with the **startNode** command run from the profile node system.

startNode command

The syntax of the **startNode** command is:

```
startNode.bat(sh) [options]
```

The options are shown in Example 7-11.

Example 7-11 startNode command

```
Usage: startNode [options]
      options: -nowait
                -quiet
                -logfile <filename>
                -replacelog
                -trace
                -script [<script filename>] [-background]
                -timeout <seconds>
                -statusport <portnumber>
                -profileName <profile>
                -recovery
                -help
```

For information about the use of the **startNode** command, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=/com.ibm.websphere.nd.multiplatform.doc/info/ae/ae/rxml_startnode.html

See Example 7-12 for an example of the **startNode** command. Note that a user ID and password is not required.

Example 7-12 startNode command

```
/opt/IBM/WebSphere/AppServer/profiles/Custom01/bin>./startnode.sh
ADMU0116I: Tool information is being logged in file
/opt/IBM/WebSphere/AppServer/profiles/Custom01/logs/nodeagent/startServer.log
```

```
ADMU0128I: Starting tool with the Custom01 profile
ADMU3100I: Reading configuration for server: nodeagent
ADMU3200I: Server launched. Waiting for initialization status.
ADMU3000I: Server nodeagent open for e-business; process id is 7480
```

Starting a node on z/OS using the START command

To start a node agent on z/OS using the START command, use the following format:

```
START nodeagent_procname,JOBNAME=server_shortname,
ENV=cell_shortname.node_shortname.server_shortname
```

For example:

```
START WPACRA,JOBNAME=WPAGNTA,ENV=WPCELL.WPNODEA.WPAGNTA
```

Stopping a node agent

To stop the node agent and leave the servers running, complete the following actions, depending on your preferred method.

From the administrative console, complete the following steps:

1. From the administrative console, click **System administration** → **Node agents**.
2. Select the box beside the node agent for the server and click **Stop**.

Or, from a command prompt, use the **stopNode** command.

Note: After you stop the node agent, the deployment manager has no way to communicate with the servers on that node. The servers might be up and running, but the administrative console is not able to determine their status.

stopNode command

The syntax of the **stopNode** command is:

```
stopNode.bat(sh) [options]
```

The options are shown in Example 7-13.

Example 7-13 The stopNode command

```
Usage: stopNode [options]
      options: -nowait
                -stopservers [-saveNodeState]
                -quiet
                -logfile <filename>
                -replacelog
                -trace
                -timeout <seconds>
                -statusport <portnumber>
                -conntype <connector type>
                -port <portnumber>
                -username <username>
                -password <password>
                -profileName <profile>
                -help
```

For more information about the **stopNode** command, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=/com.ibm.websphere.nd.multiplatform.doc/info/ae/ae/rxml_stopnode.html

See Example 7-14 for an example and sample output of the **stopNode** command.

Example 7-14 stopNode command

```
/opt/IBM/WebSphere/AppServer/profiles/Custom01/bin>./stopnode.sh -username admin  
-password adminpw  
ADMU0116I: Tool information is being logged in file  
/opt/IBM/WebSphere/AppServer/profiles/Custom01/logs/nodeagent/stopServer.log  
ADMU0128I: Starting tool with the Custom01 profile  
ADMU3100I: Reading configuration for server: nodeagent  
ADMU3201I: Server stop request issued. Waiting for stop status.  
ADMU4000I: Server nodeagent stop completed.
```

Stopping a node on z/OS using the STOP command

To stop a node agent on z/OS, use the following command:

```
STOP nodeagent_JOBNAME
```

For example:

```
STOP WPAGNTA
```

Stopping a node (the node agent and servers)

You can use the administrative console to stop a node and its servers with one action.

Complete the following steps:

1. From the administrative console, click **System administration** → **Nodes**.
2. Select the box beside the node and click **Stop**.

Restarting a node agent

You can restart a running node agent from the administrative console by completing the following steps from the administrative console:

1. Click **System administration** → **Node agents**.
2. Select the box beside the node agent for the server and click **Restart**.

7.5.2 Node agent synchronization

During a synchronization operation, a node agent checks with the deployment manager to see if any configuration documents that apply to the node have been updated. New or updated documents are copied to the node repository, and deleted documents are removed from the node repository.

Automatic synchronization

Automatic configuration synchronization between the node and the deployment manager is enabled by default. You can configure the interval between synchronizations in the administrative console by completing the following steps:

1. Expand **System administration** → **Node agents** in the administrative console.
2. Select the node agent process on the appropriate server to open the Properties page.
3. In the Additional Properties section, click **File synchronization service**.

4. Configure the synchronization interval. By default, the synchronization interval is set to one minute.

The default synchronization interval on z/OS is five minutes.

Tip: Increase the synchronization interval in a production environment to reduce the impact.

Note that a separate setting exists as an administrative console preference. The Synchronize changes with Nodes option, when selected, indicates that any time a change is saved to the console, it is automatically synchronized out to the running nodes. To set this preference, click **System administration** → **Console Preferences** and select the Synchronize changes with Nodes check box, as shown in Figure 7-30.

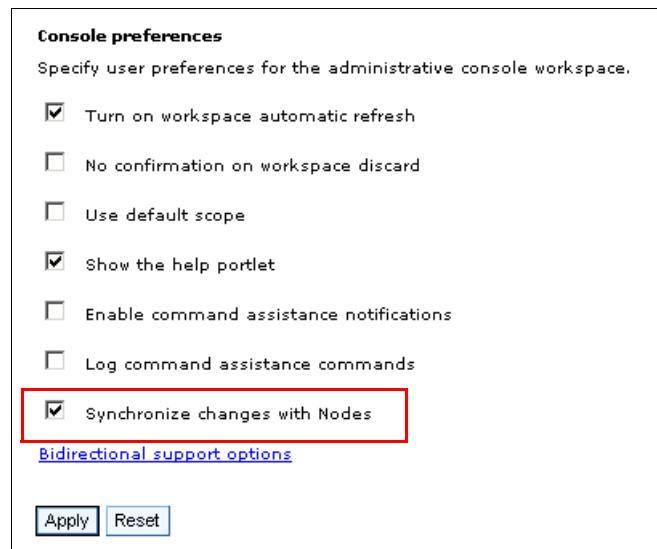


Figure 7-30 Administrative console preferences

Forced synchronization

Synchronization can be forced by clicking **System administration** → **Nodes**. Select the box beside a node and click **Synchronize** or **Full Synchronization**.

- Synchronize performs an immediate synchronization on the selected node. This type of synchronization is optimized for performance and only synchronizes changed files. If there are issues with manually edited files, this action might not result in a complete synchronization.
- The Full Synchronization option disregards optimization and ensures that the node and cell configuration are identical.

Using the syncNode command

The **syncNode** command can be used from the node to force the synchronization of a node's local configuration repository with the master repository on the deployment manager node.

Tip: The **syncNode** command is normally only used in exception situations. To use the **syncNode** command, the node agent must be stopped. You can use the **-stopservers** and **-restart** options on the **syncNode** command to stop the node agent and application servers, and then restart the node agent.

The syntax of the **syncNode** command is:

```
syncNode.bat(sh) [options]
```

The options are shown in Example 7-15.

Example 7-15 syncNode command

```
Usage: syncNode dmgr_host [dmgr_port] [-conntype <type>] [-stopservers]
      [-restart] [-quiet] [-nowait] [-logfile <filename>] [-replacelog]
      [-trace] [-username <username>] [-password <password>]
      [-localusername <localusername>] [-localpassword <localpassword>]
      [-profileName <profile>] [-help]
```

For more information about the **stopNode** command, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=/com.ibm.websphere.nd.multiplatform.doc/info/ae/ae/rxml_syncnode.html

Example 7-16 shows an example of using the **syncNode** command. The command is executed from the node. The **-stopservers** and **-restart** options are used to stop all the servers on the node, including the node agent, and then restart the node agent after the synchronization.

Example 7-16 syncNode usage examples

```
/opt/IBM/WebSphere/AppServer/profiles/Custom01/bin>./syncNode.sh T60 8882
-stopservers -re start -username admin -password adminpw
ADMU0116I: Tool information is being logged in file
/opt/IBM/WebSphere/AppServer/profiles/Custom01/logs/syncNode.log
ADMU0128I: Starting tool with the Custom01 profile
ADMU0401I: Begin syncNode operation for node Custom01 with Deployment Manager T60:
8882
ADMU0505I: Servers found in configuration:
ADMU0506I: Server name: nodeagent
ADMU0506I: Server name: ralserver1
ADMU0506I: Server name: ralserver2
ADMU2010I: Stopping all server processes for node Custom01
ADMU0512I: Server ralserver1 is now STOPPED.
ADMU0512I: Server ralserver2 cannot be reached. It appears to be stopped.
ADMU0512I: Server nodeagent cannot be reached. It appears to be stopped.
ADMU0016I: Synchronizing configuration between node and cell.
ADMU0018I: Launching Node Agent process for node: Custom01
ADMU0020I: Reading configuration for Node Agent process: nodeagent
ADMU0022I: Node Agent launched. Waiting for initialization status.
ADMU0030I: Node Agent initialization completed successfully. Process id is: 4924
ADMU0402I: The configuration for node Custom01 has been synchronized with
Deployment Manager T60: 8882
```

7.5.3 Removing a node from a cell

There are two ways of removing a node from a network distributed administration cell.

Note: When a node is removed, it is restored to its original configuration.

Using the administrative console

From the administrative console, complete the following steps:

1. Click **System administration** → **Nodes**.
2. Select the check box beside the node you want to remove and click **Remove Node**.

This method runs the **removeNode** command in the background.

Using the removeNode command

The **removeNode** command detaches a node from a cell and returns it to a stand-alone configuration.

The syntax of the **removeNode** command is:

```
removeNode.bat(sh) [options]
```

The options are shown in Example 7-17.

Example 7-17 removeNode command

```
Usage: removeNode [-force] [-quiet] [-nowait] [-statusport <port>] [-logfile
<filename>]
[-replacelog] [-trace] [-username <username>] [-password <password>]
[-profileName <profile>] [-help]
```

For more information about the **removeNode** command, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=/com.ibm.websphere.nd.multiplatform.doc/info/ae/ae/rxml_removenode.html

To use the command:

1. Change the directory to the *profile_root/bin* directory.
2. Run **removeNode**. All parameters are optional for this command.

On z/OS, the **removeNode.sh** command is in the *install_root/bin* directory. You need to specify the **-profileName** parameter to specify the profile for the node you want to remove.

The command performs the following operations:

1. Connects to the deployment manager process to read the configuration data.
2. Stops all of the running server processes of the node, including the node agent process.
3. Removes servers in the node from clusters.
4. Restores the original stand-alone node configuration. This original configuration was backed up when the node was originally added to the cell.
5. Removes the node's configuration from the master repository of the cell. The local copy of the repository held on each node will be updated at the next synchronization point for each node agent. Although the complete set of configuration files are not pushed out to other nodes, some directories and files are pushed out to all nodes.
6. Removes installed applications from application servers in the cell that are part of the node being removed.
7. Copies the original application server cell configuration into the active configuration.

The command provides the **-force** option to force the local node's configuration to be decoupled from the cell even if the deployment manager cannot be contacted. However, if this situation occurs, the cell's master repository will then have to be separately updated to reflect the node's removal, for example, through manual editing of the master repository configuration files.

Example

Example 7-18 shows an example of using the **removeNode** command.

Example 7-18 removeNode example

```
/opt/IBM/WebSphere/AppServer/bin>/removeNode.sh -profileName Custom02 -username
admin -password admin
ADMU0116I: Tool information is being logged in file
    /opt/IBM/WebSphere/AppServer/profiles/Custom02/logs/removeNode.log
ADMU0128I: Starting tool with the Custom02 profile
ADMU2001I: Begin removal of node: Custom02
ADMU0009I: Successfully connected to Deployment Manager Server: t60:8882
ADMU0505I: Servers found in configuration:
ADMU0506I: Server name: nodeagent
ADMU0506I: Server name: server1
ADMU2010I: Stopping all server processes for node Custom02
ADMU0512I: Server Custom02 cannot be reached. It appears to be stopped.
ADMU0510I: Server nodeagent is now STOPPED
ADMU2021I: Removing all servers on this node from all clusters in the cell.
ADMU2014I: Restoring original configuration.
ADMU2017I: The local original configuration has been restored.

ADMU0306I: Note:
ADMU2031I: Any applications that were uploaded to the Cell01 cell configuration
during addNode using the -includeapps option are not uninstalled by removeNode.
ADMU0307I: You might want to:
ADMU2032I: Use wsadmin or the Administrative Console to uninstall any such
applications from the Deployment Manager.

ADMU0306I: Note:
ADMU2033I: Any buses that were uploaded to the Cell01 cell configuration during
addNode using the -includebuses option are not uninstalled by removeNode.

ADMU0307I: You might want to:
ADMU2034I: Use wsadmin or the Administrative Console to uninstall any such buses
from the Deployment Manager.
ADMU2024I: Removal of node Custom02 is complete.
```

7.5.4 Renaming a node

The **renameNode** command allows you to modify the node name of a federated server.

renameNode command

The syntax of the **renameNode** command is:

```
renameNode.bat(sh) [options]
```

The options are shown in Example 7-19.

Example 7-19 renameNode command syntax

```
Usage: renameNode dmgr_host dmgr_port node_name [-nodeshortname <name>]
      [-conntype <type>] [-logfile <filename>] [-trace]
      [-username <username>] [-password <password>] [-help]
```

For more information about the **renameNode** command, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=/com.ibm.websphere.nd.multiplatform.doc/info/ae/ae/rxml_renamenode.html

To run the command, complete the following steps:

1. Change to the *profile_root/bin* directory of the deployment manager.
2. Run the **renameNode** command.

The command:

1. Connects to the deployment manager.
2. Stops all servers.
3. Changes the node configuration on the deployment manager.
4. Synchronizes the node.

7.5.5 Node groups

In a distributed environment, you can have nodes in a cell with different capabilities. However, there are restrictions on how the nodes can coexist.

Node groups are created to group nodes of similar capability together to allow validation during system administration processes. Effectively, this situation means that a node group establishes a boundary from which servers can be selected for a cluster. Nodes on distributed platforms and nodes on the IBM i platform can be members of the same node group, but they cannot be members of a node group that contains a node on a z/OS platform.

Note: Do not confuse node groups with “groups of nodes” in the job manager. These are two different concepts.

A default node group called **DefaultNodeGroup** is automatically created for you when the deployment manager is created, based on the deployment manager platform. New nodes on similar platforms are automatically added to the default group. A node must belong to at least one node group, but can belong to more than one.

As long as you have nodes in a cell with similar platforms, you do not need to do anything with node groups. New nodes are automatically added to the node group. However, before adding a node on a platform that does not have the same capabilities as the deployment manager platform, you will need to create the new node group.

Working with node groups

You can display the default node group and its members by clicking **System Administration** → **Node Groups**, as shown in Figure 7-31.

The screenshot shows a web-based administration interface for managing node groups. At the top, there is a brief description of what a node group is: "Use this page to manage node groups. A node group is a collection of application server nodes. A node group establishes a boundary for cluster creation. All cluster members must be on nodes that are members of the same node group. Nodes that are organized into a node group need enough capabilities in common to ensure that clusters formed across those nodes can host the same application in each cluster member. A node must be a member of at least one node group and can be a member of more than one node group. To delete a node group, it must be empty." Below this is a navigation bar with a "Preferences" link and buttons for "New..." and "Delete". There are also icons for creating, deleting, and modifying resources. A table lists the node groups, with columns for Select, Name, Members, and Description. The table shows one entry: "DefaultNodeGroup" with 2 members, described as "WebSphere Default Node Group.". A footer indicates "Total 1" node group.

Figure 7-31 Display a list of node groups

On this window, you can perform a number of actions:

- ▶ To create a new node group, click **New**. The only thing that you need to enter is the name of the new node group. Click **OK**.
- ▶ To delete a node group, select the box to the left of the node group name and click **Delete**.
- ▶ To display a node group, click the node group name. Figure 7-32 shows the DefaultNodeGroup.

This screenshot shows the "Node group properties" dialog box. It has two main sections: "General Properties" on the left and "Additional Properties" on the right. In the "General Properties" section, there are fields for "Name" (set to "DefaultNodeGroup"), "Members" (set to "2"), and "Description" (set to "WebSphere Default Node Group."). In the "Additional Properties" section, there are two expandable categories: "Custom properties" and "Node group members". At the bottom of the dialog are buttons for "Apply", "OK", "Reset", and "Cancel".

Figure 7-32 Node group properties

- To add a node to a node group, display the node group and click **Node group members** in the Additional Properties section (Figure 7-32 on page 309). When the list appears, click **Add**. You will be able to select from a list of nodes (Figure 7-33).

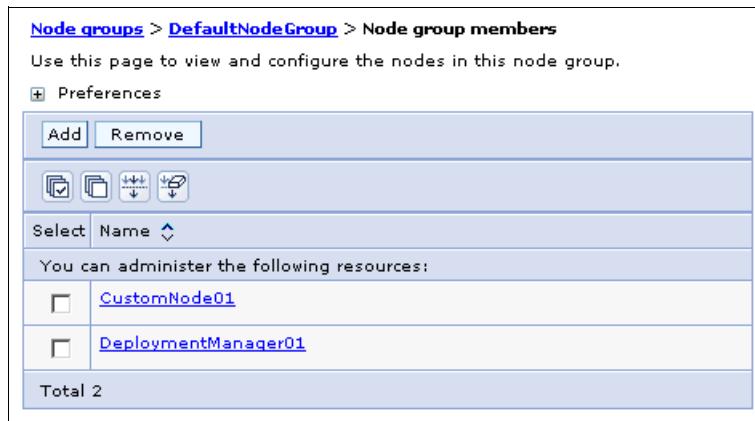


Figure 7-33 Displaying node group members

7.6 Working with clusters

This section provides information about creating, configuring, and managing application server clusters using the administrative console.

Clusters consist of one or more application servers that run the same applications. The configuration of each server can be unique.

Before creating a cluster, consider the number of servers you want to add to the cluster, the nodes on which they will be created, and how the workload should be distributed across the servers.

Work is distributed across the servers in the cluster based on weights assigned to each application server. If all cluster members have identical weights, work is distributed among the cluster members equally. Servers with higher weight values are given more work. An example formula for determining routing preference is as follows:

$$\% \text{ routed to Server1} = \text{weight}_1 / (\text{weight}_1 + \text{weight}_2 + \dots + \text{weight}_n)$$

In the formula, n represents the number of cluster members in the cluster. Consider the capacity of the system that hosts the application server.

7.6.1 Creating application server clusters

When you create a cluster, you have the option to create an empty cluster (no servers) or to create the cluster with one or more servers. The first application server added to the cluster acts as a template for subsequent servers. You can create the first server during the cluster creation process or you can convert an existing application server. The rest of the servers must be new and can be created when you create the cluster or added later.

Tip: When creating a cluster, it is possible to select the template of an existing application server for the cluster without adding that application server into the new cluster. If you need to change the attributes of the servers in your cluster after the cluster has been created, you must change each server individually. For this reason, consider creating an application server with the server properties that you want as a standard in the cluster first, then use that server as a template or as the first server in the cluster.

Cluster and cluster member options

When you create a new cluster, you have the following options to consider:

- ▶ Prefer local:

This setting indicates that a request to an EJB should be routed to an EJB on the local node if available. This is the default setting and generally will result in better performance.

- ▶ Configure HTTP session memory-to-memory replication (create a replication domain):

WebSphere Application Server supports session replication to another WebSphere Application Server instance. In this mode, sessions can replicate to one or more WebSphere Application Server instances to address HTTP Session single point of failure.

When you create a cluster, you can elect whether to create a replication domain for the cluster. The replication domain is given the same name as the cluster and is configured with the default settings for a replication domain. When the default settings are in effect, a single replica is created for each piece of data and encryption is disabled. Also, the web container for each cluster member is configured for memory-to-memory replication.

For more information about replication domains, refer to 25.8, “Persistent session management” on page 903.

When you create a new cluster member, you have the following options to consider:

- ▶ Basis for first cluster member:

You can add application servers to the cluster when you create the cluster or later.

The first cluster member can be a new application server or you can convert an existing application server so that it becomes the first cluster member.

Subsequent application servers in the cluster must be created new. The first application server in the cluster acts as a template for the subsequent servers.

The options you have depend on the how you create the cluster.

When you use the job manager, you have the option to convert an existing server to use as the first cluster member, or create an empty cluster and run additional jobs to add cluster members.

When you use the deployment manager, you can convert an existing server, create one or more new servers, or create an empty cluster.

Note: The option to use an existing application server does not appear in the deployment manager administrative console if you create an empty cluster and then add a member later. If you want to convert an existing application server as the first server, specify that option when you create the cluster or use the job manager to create the cluster member.

Tip: To remove a server from a cluster, you must delete the server. Take this situation into consideration when you are determining whether to convert an existing server to a cluster.

- ▶ Server weight for each cluster member:

The weight value controls the amount of work that is directed to the application server. If the weight value for this server is greater than the weight values that are assigned to other servers in the cluster, then this server receives a larger share of the workload. The weight value represents a relative proportion of the workload that is assigned to a particular application server. The value can range from 0 to 20.

Member weight: Specify the relative weight of this server in the cluster. Values are from 0 to 20. A 0 indicates that work is to be routed to this server only in the event that no other servers are available.

On z/OS, weight is used to balance some of the workload types, but others are balanced by the z/OS system:

- For HTTP requests, weights are used to distribute HTTP traffic between the web server plug-in and the controller handling the clustered application server. Assign a higher weight value to the application server that should receive the HTTP traffic.
- For web services calls, information is transferred from a servant in one application server to a controller in another application server. The application server that receives the call has the highest weight value.
- Weight has no affect on Internet Inter-ORB Protocol (IIOP) requests. IIOP requests are distributed to the correct application server using the sysplex distributor.

Using the deployment manager administrative console

To create a new cluster, complete the following steps:

1. Click **Servers** → **Clusters** → **WebSphere application server clusters**.
2. Click **New**.
3. Enter the information for the new cluster (see Figure 7-34):
 - Enter a cluster name of your choice.
 - On z/OS, you will also be asked for the short name for the cluster.

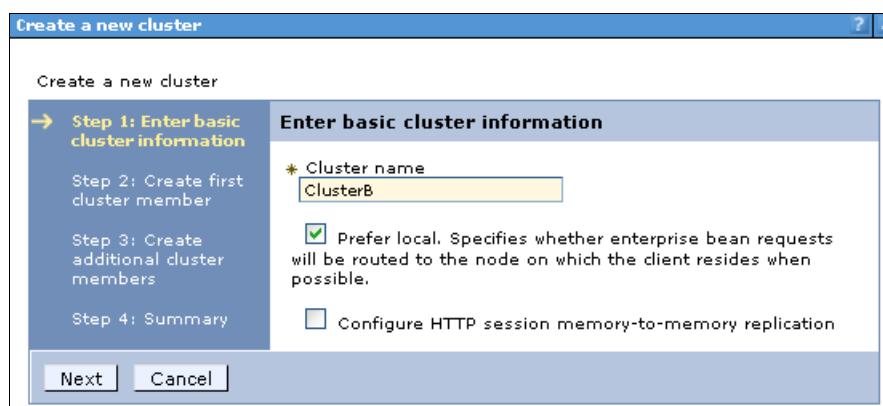


Figure 7-34 Creating a new cluster

4. Create first cluster member: The first cluster member determines the server settings for the cluster members (Figure 7-35).

Create first cluster member

The first cluster member determines the server settings for the cluster members. A server configuration template is created from the first member and stored as part of the cluster data. Additional cluster members are copied from this template.

* Member name
server1b1

Select node
CustomNode01(ND 8.0.0.0)

* Weight
2 (0..20)

Generate unique HTTP ports

Core Group
DefaultCoreGroup

Select how the server resources are promoted in the cluster.
Cluster

Select basis for first cluster member:

- Create the member using an application server template.
default
- Create the member using an existing application server as a template.
Cell01/CustomNode01(ND 8.0.0.0)/ralserver1
- Create the member by converting an existing application server.
Cell01/CustomNode01(ND 8.0.0.0)/ralserver1
- None. Create an empty cluster.

Previous Next Cancel

Figure 7-35 First cluster member

The fields are:

- Member Name: Enter the name of the new server to be added to the cluster. On z/OS, you will also be asked for the short name for the server.
- Select Node: Specifies the node on which this new cluster member is created.
- Server weight: Assign the weight for this server.
- Generate unique HTTP ports: Generates unique port numbers for every transport that is defined in the source server, so that the resulting server that is created will not have transports that conflict with the original server or any other servers defined on the same node.
- Core Group: Because multiple core groups exist, you must select the core group for the cluster members to join. This field only displays if you have multiple core groups defined.

- Select how the server resources are promoted in the cluster: Specifies how resources, such as data sources, are initially created in the cluster. This option is only available for the first cluster member. All other cluster members are based on the cluster member template, which is created from the first cluster member. You can select from the following options:
 - Cluster, which indicates the resources defined can be used across all cluster members. This setting reduces the amount of configuration and management of resources. The cluster option is the default setting.
 - Server, which indicates that the resources are defined at the cluster member level. This setting is useful if you want to have different configuration settings for resources defined on each cluster member.
 - Both, which copies the resources of the cluster member (server) level to the cluster level.
 - Select basis for first cluster member:
 - If you click **Create the member using an application server template**, the settings for the new application server are identical to the settings of the application server template you select from the list of available templates.
 - If you click **Create the member using an existing application server as a template**, the settings for the new application server are identical to the settings of the application server you select from the list of existing application servers. However, applications that are installed on the template server are not installed on the new servers.
 - If you click **Create the member by converting an existing application server**, the application server you select from the list of available application servers becomes a member of this cluster.
- Applications that are installed on the existing server are automatically installed on new members of the cluster.
- Note that the only way to remove a server from a cluster is to delete it, and when you delete the cluster, all servers in the cluster are deleted.
- If you click **None. Create an empty cluster**, a new cluster is created, but it does not contain any cluster members.

Click **Next**.

5. Create additional cluster members: Use this window to create additional members for a cluster. You can add a member to a cluster when you create the cluster or after you create the cluster. A copy of the first cluster member that you create is stored as part of the cluster data and becomes the template for all additional cluster members that you create.

To add a member, enter a new server name, select the node, and click **Add Member**. The new member will be added to the list, as shown in Figure 7-36.

Create additional cluster members

Enter information about this new cluster member, and click Add Member to add this cluster member to the member list. A server configuration template is created from the first member, and stored as part of the cluster data. Additional cluster members are copied from this template.

* Member name
server2b1

Select node
CustomNode01(ND 8.0.0.0)

* Weight
2 (0..20)

Generate unique HTTP ports

Add Member

Use the Edit function to modify the properties of a cluster member in this list. Use the Delete function to remove a cluster member from this list. You are not allowed to edit or remove the first cluster member.

Select	Member name	Nodes	Version	Weight
	server1b1	CustomNode01	ND 8.0.0.0	2
Total 1				

Edit **Delete**

Previous Next Cancel

Figure 7-36 Additional cluster members

6. When all the servers have been entered, click **Next**.
7. A summary window shows you what will be created.
8. Click **Finish** to create the cluster and new servers.
9. Save the configuration.

Adding additional servers to the cluster

To add a server using the administrative console, complete the following steps:

1. Click **Servers** → **Clusters** → **WebSphere application server clusters**.
2. Click the cluster name.
3. Under the Additional Properties sections, click **Cluster members**.
4. Click **New**. This action opens the same configuration window that was used when you created the cluster (Figure 7-36).
5. Enter the name of the new server to create, select the node, and select the options to use. Click **Add Member**.
6. Click **Next**, and then click **Finish**.

Using cluster member templates

When you created your cluster's servers, a server template was created for the cluster by copying the first cluster member's configuration. This template is then used when you create additional servers for that cluster. This situation is important to understand, because you might not get the results you expect when working with clusters.

To view a cluster's member templates using the administrative console, complete the following steps:

1. Click **Servers** → **Clusters** → **WebSphere application server clusters**.
2. Click the cluster name.
3. Under the Additional Properties sections, click **Cluster members**.
4. Click **Templates....**

These steps will open a configuration window that lists the templates in this cluster. Typically, you will only have one template, but you will have additional templates if the cluster includes servers that are at different versions of WebSphere Application Server (for example, versions 8 and 7).

From this configuration window, you can view and modify the server attributes of the template. If you modify the attributes, it is important to understand that existing cluster members will not be affected. The template is only used for creating new cluster members.

To modify the attributes of a cluster's member using the administrative console, complete the following steps:

1. Click **Servers** → **Clusters** → **WebSphere application server clusters**.
2. Click the cluster name.
3. Under the Additional Properties sections, click **Cluster members**.
4. Click the cluster member. This action opens the server configuration window where you can make your change.

If you want to make the same change to multiple cluster members, you must repeat these steps. You should also modify the same attributes in the cluster member template, because new cluster members will be created based on the template. If you do not change the cluster's template(s), additional cluster members will not match the existing members.

If you want to modify the same server attribute(s) across all of a cluster's members and you have several cluster members, one way to accomplish this task is by using the administrative console and completing the following steps:

1. Navigate to the cluster's templates.
2. Click the cluster template and make your desired changes.
3. Save your changes.
4. Delete all of the members in the cluster.
5. Recreate the members.

The new members will be created from the updated cluster member template, and all of the cluster members will have the same configuration.

Using the job manager

When you create an application server cluster from a job manager, you can either create an empty cluster and run subsequent jobs to add cluster members, or you can convert an existing application server as your first cluster member. If you want to create a cluster with one or more new application servers in one step, use the administrative console instead of the job manager.

Building a cluster using the job manager is done in two major steps:

1. Create the cluster.
2. Create the cluster members.

Creating the cluster

To create an application server cluster from the job manager, complete the following steps:

1. Click **Jobs** → **Submit**.
2. Click the **Create cluster** job type.
3. Select the deployment manager as the job target.
Enter the user ID and password with administrative authority on the deployment manager.
4. Specify the job parameters, as shown in Figure 7-37.
 - Specify the name of the new cluster.

The screenshot shows a 'Specify job parameters' dialog box. On the left, a vertical sidebar lists steps: Step 1: Choose a job type, Step 2: Choose job targets, Step 3: Specify job parameters (which is highlighted in yellow), Step 4: Schedule the job, and Step 5: Review the summary and submit the job. The main area contains the following fields:

- Job type: Create cluster**
- * Cluster name:** ClusterC
- Prefer local
- Cluster type:** Application Server
- Short name:** (empty input field)
- Additional job parameters
- Replication domain**
- Create domain
- Convert server**
- Server node:** (empty input field)
- Server name:** (empty input field)
- Member weight:** (empty input field)
- Node group:** (empty input field)
- Replication entry:** (empty input field)

At the bottom are buttons: Previous, Next, and Cancel.

Figure 7-37 Specify the options for the new server

Optionally:

- Prefer local: Selected, which is true (the default), or clear the check for false
- Cluster type: The options are:
 - APPLICATION_SERVER (the default)
 - PROXY_SERVER
 - ONDEMAND_ROUTER

Leave this field blank to create an application server cluster.
- Short name: Cluster short name on z/OS platforms
- Create domain: For creating true or false (the default)
- Convert server settings: If you want to use an existing server as the first member of the cluster, complete the server node and server name fields. The other fields are optional.

If you specify the cluster name, and take all the other defaults, you will create an empty cluster. When you create an empty cluster, it does not appear in the deployment manager console until you submit a job to add a member to it.

5. Schedule the job.

Take the defaults for the job schedule. The defaults will execute the job once. Click **Next**.

6. Review the summary and click **Finish**. Monitor the status of the job and ensure it completes successfully.

Creating the cluster members

To create new application server cluster members from the job manager, complete the following steps:

1. Click **Jobs → Submit**.
2. Click the **Create cluster member** job type.
3. Select the deployment manager as the job target.

Enter the user ID and password with administrative authority on the deployment manager.

4. Specify the job parameters, as shown in Figure 7-38:
 - a. Specify the name of the cluster.
 - b. Specify the node where the cluster member will be created.
 - c. Specify the name for the new cluster member.

The screenshot shows a software interface for specifying job parameters. On the left, a vertical navigation bar lists steps: Step 1: Choose a job type, Step 2: Choose job targets, Step 3: Specify job parameters (which is highlighted in blue), Step 4: Schedule the job, and Step 5: Review the summary and submit the job. The main panel is titled 'Specify job parameters' and has a sub-section 'Job type: Create cluster member'. It contains the following fields:

- * Cluster name: ClusterC
- * Member node: Custom02
- * Member name: server1c1
- Member weight: (empty input field)
- Member UUID: (empty input field)
- Generate unique ports
- Replicator entry
- Short name: (empty input field)
- + Additional job parameters...

At the bottom of the dialog are buttons for Previous, Next, and Cancel.

Figure 7-38 Specify the options for the new server

Optionally:

- Member weight: Specify the relative weight of this server in the cluster. Values are from 0 to 20. A 0 indicates that work is to be routed to this server only in the event that no other servers are available.
- Member UUID.
- Generate unique ports: The default is the check box is selected for this option, which generates unique ports.
- Replicator entry: Selecting the check box adds a replicator entry for this server in the cluster replication domain. The default is not selected, which is a false value.
- Short name: The short name for the server on z/OS.

If this is the first server in the cluster, or if you want to specify a different node or core group, expand the additional job parameters section. It contains settings to specify the template information and the option to specify a node and core group other than the default.

5. Schedule the job.

Use the defaults for the job schedule. The defaults will execute the job once. Click **Next**.

6. Review the summary and click **Finish**. Monitor the status of the job and ensure it completes successfully.

7.6.2 Viewing the cluster topology

The deployment manager administrative console provides a graphical view of the existing clusters and their members. To see the view, complete the following steps:

1. Click **Servers** → **Clusters** → **Cluster Topology**.
2. Expand each category, as shown in Figure 7-39.

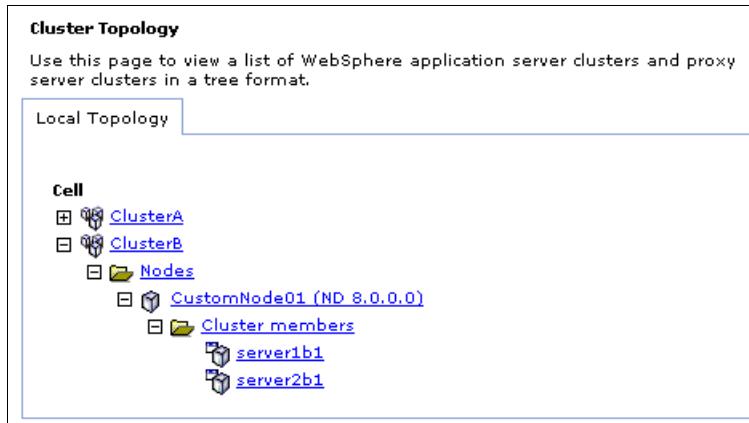


Figure 7-39 Cluster topology view

3. Selecting a server will take you to the configuration window for the application server.

7.6.3 Managing clusters

Application servers within a cluster can be managed as independent servers. A second option is to manage all the servers in the cluster as a single entity.

Using the administrative console

To display and manage the application server clusters, complete the following steps:

1. Click **Servers** → **Clusters** → **WebSphere application server clusters**.
2. Select each cluster you want to work with and select one of the following options:
 - Start: Use this option to start all servers in the cluster.
 - Stop: Use this option to stops all servers in the cluster. This option allows the server to finish existing requests and allows failover to another member of the cluster.
 - Ripplestart: Use this option to stop and then start all servers in the cluster one at a time.
 - ImmediateStop: Stop all servers immediately.
 - Delete: Deletes the cluster and all servers in the cluster.

Using the job manager

The job manager provides several job types that help you manage clusters:

- ▶ Delete cluster
- ▶ Delete cluster member
- ▶ Start cluster: Use this option to start all servers in a cluster.
 - You can specify that a ripplestart be used (specify true or false). The default is that a ripplestart is not used. A ripplestart will stop and then restart each server.

- You can also specify a timeout value. If the timeout expires and all servers have not started, the state of the cluster will be reported without waiting any longer for the servers to start.
- ▶ Stop cluster:
You can also specify a timeout value. If the timeout expires and all servers have not started, the state of the cluster will be reported without waiting any longer for the servers to start.

7.7 Working with virtual hosts

Note: For many users, creating virtual hosts is unnecessary because the `default_host` that is provided is sufficient.

For an example of defining and using a new virtual host, see 21.8, “Deploying the application” on page 818.

A *virtual host* is a configuration enabling a single host machine to resemble multiple host machines. It consists of a host alias or aliases, which consist of a host name and a port number. If you specify an asterisk (*) as a host name, all host names and IP addresses that the web server can receive will be mapped to that virtual host.

The following virtual hosts are defined during installation:

- ▶ The `default_host` virtual host is intended for access to user applications, either through the HTTP transport or through a web server.
Host aliases in this virtual host generally include the ports required to access applications from the web server and directly to the application server. Examples are the `wc_defaulthost`, `wc_defaulthost_secure`, `sip_defaulthost`, and `sip_defaulthost_secure` ports for application servers and ports 80 and 443 for requests through the web server.
- ▶ The `admin_host` virtual host is used for access to the WebSphere administrative console.
At installation time, the host is configured to match requests on the `wc_adminhost` and `wc_adminhost_secure` ports for the stand-alone server or deployment manager.
- ▶ The `proxy_host` virtual host includes default port definitions, port 80 and 443, which are typically initialized as part of the proxy server initialization. Use this proxy host as appropriate with routing rules associated with the proxy server.

When you install an application, you associate a virtual host with each web module in the application. By associating a virtual host with a web module, requests that match the host aliases for the virtual host should be processed by servlets in this web module. The web server plug-in also checks the URI of the request against the URIs for the web module to determine whether the web module can handle them or not. You can view or modify the virtual host to which a web module is assigned by clicking **Applications** → **Application Types** → **WebSphere enterprise applications** → `app_name` → **[Web Module Properties]** **Virtual hosts**.

A single virtual host can be associated with multiple web modules as long as each application has unique URIs. If there are duplicate URIs among applications, different virtual hosts must be created and associated with each of the applications.

A default virtual host is associated with a web container when you create the application server. To find the default virtual host, click **Servers** → **Server Types** → **WebSphere application servers** and click the server name to open the configuration page. In the Container settings section, expand **Web Container Settings** and click **Web container**.

7.7.1 Creating and updating virtual hosts

By default, default_host is associated with all user application requests. The following examples show cases in which multiple virtual hosts should be created:

- ▶ Applications with conflicting URIs
- ▶ Special support for extra ports
- ▶ Providing independence of each virtual host for applications and servers

To create a new virtual host, complete the following steps:

1. Click **Environment** → **Virtual hosts** and then click **New**.
2. Enter a name for the virtual host and click **Apply**. Note that two links become active: Host Aliases and MIME Types.
3. Click **Host Aliases** in the Additional Properties pane.
4. Click **New**.
5. Enter values for the Host Name and Port fields and click **OK**.

The host aliases are not necessarily the same as the host name and port number of the WebSphere Application Server servers. They are the host names and port numbers that the web server plug-in is expecting to receive from the browser. The web server plug-in will send the request to the application server using the host name and port number in the transport setting for that server. If the web server is running on a separate machine from WebSphere, then the host aliases are for web server machines.

Mapping HTTP requests to host aliases is case sensitive and the match must be alphabetically exact. Also, different port numbers are treated as different aliases.

For example, the request `http://www.myhost.com/myservlet` does *not* map to any of the following steps:

- `http://myhost/myservlet`
- `http://www.myhost.com/MyServlet`
- `http://www.myhost.com:9876/myservlet`

If the web server plug-in receives a request that does not match one of the virtual hosts, it passes the request to the web server. The web server looks in the `web_server_root/htdocs` directory for the content. If it finds the content, it serves the page to the client. If it does not find the content, an HTTP 404 response is returned to the client.

Simple wild cards can be used on the host aliases. A * can be used for the host name, the port, or both. It means that any request will match this rule.

Note: If the virtual host is used in a cluster environment, all host aliases used by servers in the cluster should be registered in the virtual host.

6. Save your changes.

Host aliases can also be updated for virtual hosts through the administrative console. To update, complete the following steps:

1. Click **Environment** → **Virtual hosts**.
2. Click the virtual host name to open the configuration page.

3. Click **Host Aliases** in the Additional Properties pane.
4. Click **New**.
5. Enter values for the Host Name and Port fields and click **OK**.

Important: If you create, delete, or update virtual hosts, you need to regenerate the web server plug-in.

7.8 Managing applications

WebSphere Application Server V8 supports J2EE 1.3, J2EE 1.4, Java EE 5, and Java EE 6, which we refer to as *enterprise applications*. WebSphere Application Server V8 can run the following types of applications:

- ▶ Java EE applications
- ▶ Portlet applications
- ▶ Session Initiation Protocol applications
- ▶ Business-level applications
- ▶ OSGi applications (New in Version 8)

A new concept that was introduced with Version 7 allows you to manage applications at a business level. A *business-level application* does not contain binary application files, but rather, identifies WebSphere and non-WebSphere artifacts that logically belong together to form a business-level application. Deployable EAR files and modules are assets for the business-level application. After being deployed, an application becomes a composition unit to the business-level application. Administration of the artifacts that belong to the business-level application is separate from the administration of the business level definition.

OSGi applications are built on an architecture for developing and deploying modular applications and libraries. OSGi applications are built using the OSGi API and deploying it into an OSGi container. WebSphere Application Server provides an OSGi container as part of its basic architecture. For more information about OSGi applications, see Chapter 24, “Working with OSGi applications” on page 871.

Applications can be configured and managed using the following methods:

- ▶ Using the **wsadmin** tool:

The **wsadmin** scripting tool can be used to manage both business level and enterprise applications.

Using scripts to manage applications is more complicated than using the other methods. It requires skill in at least one of the supported scripting languages and a complete understanding of the WebSphere Application Server configuration. However, scripting can offer a greater degree of control and can be quite useful in situations where you are performing the same administrative tasks multiple times, or when the tasks are to be done by multiple administrators. With introduction of script libraries, many of the scripting tasks have become easier to perform.

Information about using **wsadmin** scripts is found in Chapter 8, “Administration with scripting” on page 343.

- ▶ Using the administrative console:

Using the administrative console is an easy way to install or update business level and enterprise applications. Wizards take you through the process and provide help information at each step. This is the method provided in this section at a high level.

- ▶ Using the job manager:

The job manager can be used to install, update, uninstall, stop, and start enterprise applications. Currently, business-level applications must be managed by submitting **wsadmin** jobs.

- ▶ Using a monitored directory:

(New in Version 8) Using the new drag and drop monitored directory deployment feature allows you to install and update an application simply by dragging or copying the file to a monitored directory. Monitored Directory support is available in all packages and supports deployment of Java EE 5 and later modules. You can also use application properties files to install and update an application through the monitored directory support. By default, monitored directory deployment is not enabled. For information about enabling and using this feature, see 22.4, “Using a monitored directory” on page 844.

WebSphere Rapid Deployment tools provide a shortcut to installing, uninstalling, and updating applications. WebSphere Rapid Deployment is still present in WebSphere Application Server V8 and supported only in the Base package. However, these tools do not support JEE 5 nor J2EE 1.2 specification levels. For information about using this feature, refer to the topic “Rapid deployment of J2EE applications” in the Information Center.

7.8.1 Managing enterprise applications: Administrative console

To view and manage applications using the administrative console, click **Applications** → **Enterprise Applications**.

In the window, you see the list of installed applications and options for performing application management tasks. Select one or more applications by selecting the box to the left of the application name, and then click an action to perform. The exception to this action is the Install option, which installs a new application, and requires no existing application to be selected, as shown in Figure 7-40.

Select	Name	Application Status
<input type="checkbox"/>	DefaultApplication.ear	✗
<input type="checkbox"/>	Dynamic Cache Monitor	✗

Figure 7-40 Working with enterprise applications

The following list describes the actions you can choose on this window:

- ▶ Start:

Applications normally start when the server to which they are mapped starts. Exceptions to this situation include when the application has just been installed, and when the application has been stopped manually.

- ▶ Stop:
You can stop an application manually without affecting the rest of the application server processes. This situation is common when you are updating an application or want to make it unavailable to users.
- ▶ Install:
The install option takes you through the process of installing a new enterprise application EAR file.
- ▶ Uninstall:
Use this option to uninstall an application. This action removes it from the application servers and from the configuration repository.
- ▶ Update or Rollout Update:
Applications can be updated in several ways. The update options include full application, single module, single file, and partial application.
- ▶ Remove file:
With this option, you can remove a single file from an application.
- ▶ Export:
Use this option to export an EAR file of the application.
- ▶ Export DDL:
Use this option to export DDL files found in the application.
- ▶ Export File:
Use this option to export individual files found in the application.

7.8.2 Deploying an enterprise application

To install an enterprise application into a WebSphere configuration, you must install its modules onto one or more application servers.

Business-level applications (BLAs): Enterprise applications are considered a composite unit of a business-level application (BLA) and will be can either be added to the BLA at installation time, or can be installed when the undeployed application assets are added to the BLA.

Adding a new cluster member: When an application server is added as a member to a server cluster, the modules installed on other members are also installed on the new member. You do not need to re-install or upgrade the application.

In this section, we give you an overview of application installation. Later in this book, we provide information about application installation in detail:

- ▶ An example of installing an enterprise application is given in 21.8, “Deploying the application” on page 818.
- ▶ An example of installing a business-level application can be found in 21.9, “Deploying business-level applications” on page 822.

Using business-level applications

When you install an application, it is added to a business-level application (BLA). This BLA can be an existing BLA, or you can create a new BLA as part of the application installation process. If you do not plan to use BLAs to manage your applications, allowing the BLA to be created during the application install process is fine.

However, if you plan to manage your application as part of a BLA, you should start the installation process by importing the application EAR file as an asset, creating a business-level application to hold the assets, and then installing the assets.

Complete the following steps:

1. Import the application EAR file as an asset:

- a. Open the WebSphere administrative console and click **Applications** → **Application Types**. Click the **Assets** link.
- b. Click the **Import** button and complete the steps in the wizard.

The asset will be imported to the asset repository. The default location is *profile_root/installedAssets/asset_name/BASE/*.

Repeat this step for any additional assets that will belong to the BLA.

Click **Save to master configuration** when done.

2. Create a business-level application:

- a. Click **Applications** → **Application Types** → **Business-level applications**. Click the **New** button.
- b. Enter the name for the BLA and click **Apply**.
- c. On the Business-level applications page, click the **Add** button under the Deployed assets section. You can use this option to add an asset (the EAR file) or a shared library.
Select the **Add Asset** option.
- d. On the next window, select the EAR file and click **Continue**. You will now configure (install) this asset with the necessary deployment options. This installation follows the normal steps for installing an application to WebSphere Application Server (see “Installing an application using the administrative console” on page 326).
- e. Proceed through the installation windows until the Summary window is shown, and click **Finish**. WebSphere now installs the application and it is now a composition unit, an asset which has been configured.

Note: At Step 1: Select installation options, WebSphere generates a unique application name, such as app1143018803114601914, for the application. You might want to change this name to something more descriptive.

3. Click **Save to the master configuration** when done.

Installing an application using the administrative console

There are two approaches to installing an enterprise application. The first approach is to install the application as part of the process of adding assets to a BLA. The second approach is to install the application and select a BLA, or create a new BLA as part of the process. This section assumes that you are taking the latter approach.

In either case, the process is basically the same after you get into the application installation wizard.

Complete the following steps:

1. Click **Applications** → **New Application** → **New Enterprise Application**.
2. Specify the location of the EAR file to install.

The EAR file that you are installing can be either on the client machine running the web browser, or on any of the nodes in the cell (Figure 7-41).

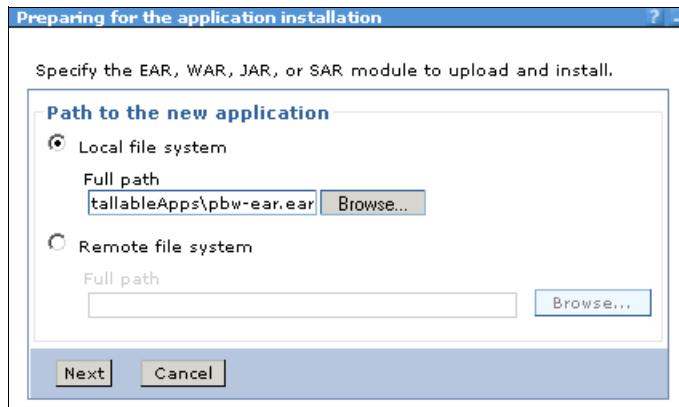


Figure 7-41 *Installing an enterprise application*

Click **Next**.

3. Select whether to use the fast path through the wizard or to display all the installation steps in the wizard.

The fast path will only display the steps in the wizard where decisions cannot be made through defaults.

If you expand **Choose to generate default bindings and mapping**, you can alter the bindings for the application you are deploying. If you select the **Generate Default Bindings** option, WebSphere Application Server completes any incomplete bindings in the application with default values, but it does not alter any existing bindings. Selecting **Override existing bindings** allows you to specify a bindings file, which contains new bindings (Figure 7-42).

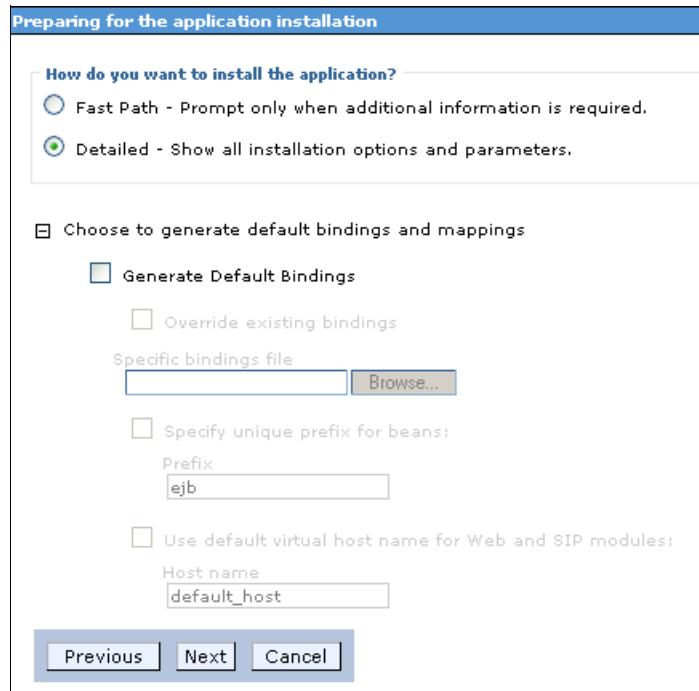


Figure 7-42 Select the path through the wizard and default binding options

Click **Next**.

4. The rest of the installation process is done in steps. The steps can vary, depending on the contents of the EAR file and whether you selected the fast path or detailed path through the wizard. The following steps are a typical sequence for the detailed path:
 - a. Provide options to perform the installation.
This window also provides you the option of selecting a BLA or creating a new one. If you are installing the application as a result of adding an asset to a BLA, the correct BLA should be selected.
If you create a new BLA, it will have the same name as the application.
 - b. Map modules to servers.
 - c. Provide JSP reloading options for web modules.
 - d. Map shared libraries.
 - e. Map shared library relationships.
 - f. Provide JNDI names for beans.
 - g. Bind EJB business interfaces to a JNDI name.
 - h. Map EJB references to beans.
 - i. Map resource references to resources.
 - j. Map virtual hosts for web modules.

- k. Map context roots for web modules.
 - l. Map security roles to users or groups.
 - m. Map the JASPI provider.
 - n. Specify metadata options. The settings on this page can be used to instruct a Java EE 5 enterprise bean (EJB) or web module deployment descriptor to ignore annotations that specify deployment information.
5. Click **Finish** to install the application.
 6. Save the configuration and synchronize the changes to the nodes.
 7. Update the web server plug-in and propagate it to the web server if you have a web server configured in your environment.

Using the job manager

You can install enterprise applications from the job manager by selecting the following options to submit a job.

Note: When you use the job manager to install an application, the only options you can specify from the job manager console is the location to install; all the other options that you see in the administrative console will be the defaults.

The first step is to make the application available to the system where the installation job will run. Complete the following steps:

1. Before installing an application, you must transfer the application binaries (EAR file) from the job manager to the node where the job will run using the distributeFile job.

In Rational Application Developer, export the EAR file to the *jmgr_profile_root/config/temp/JobManager* directory.

2. In the Job manager console, click **Jobs** → **Submit**. This action launches the Job properties wizard.
 - a. Select the **Distribute file** job type and click **Next**.
 - b. Select the deployment manager or administrative agent as the target node.
 - c. Enter the EAR file location on the job manager and the location to store the EAR file on the node. The arguments are entered as shown in Figure 7-43.

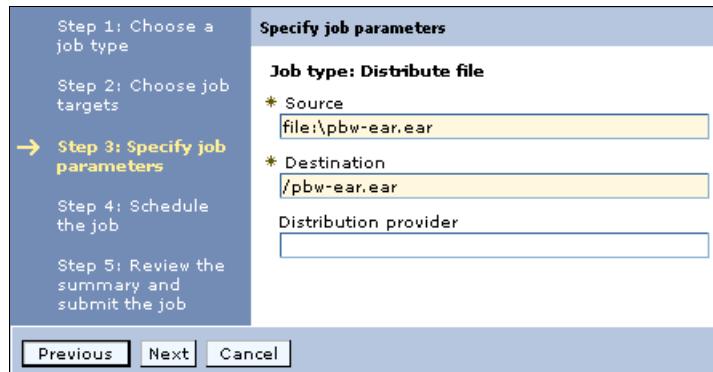


Figure 7-43 Specify the location of the source and target file

Using these arguments, the script file will be distributed to the following location:

dmgr_profile_root/downloadedContent/app_name

Click **Next**.

- d. Use the defaults for the job schedule. The defaults will execute the distribute file job once. Click **Next**.
- e. Review the summary and click **Finish**. Monitor the status of the job and ensure it completes successfully.

To install the application from the job manager, complete the following steps:

1. Click **Jobs** → **Submit**.
2. Select the **Install application** job type and click **Next**.
3. Select the node where the job will run.
Enter the user ID and password with administrative authority on the target node.
4. Specify the job parameters.

At minimum:

- Specify the name of the new application.
- If the target is a deployment manager, enter the name of the node and server or cluster on which the application will be installed (Figure 7-44).

The screenshot shows a 'Specify job parameters' dialog box. On the left, a sidebar lists steps: Step 1: Choose a job type, Step 2: Choose job targets, Step 3: Specify job parameters (which is highlighted in blue), Step 4: Schedule the job, and Step 5: Review the summary and submit the job. The main area is titled 'Job type: Install application'. It contains the following fields:

- * Application name: PlantsByWebSphere
- Application location: (empty input field)
- Server name: (empty input field) with a 'Find...' button to its right
- Node name: (empty input field)
- Cluster name: ClusterC with a 'Find...' button to its right

At the bottom of the dialog are 'Previous', 'Next', and 'Cancel' buttons.

Figure 7-44 Specify the options for application install

- f. Use the defaults for the job schedule. The defaults will execute the job once. Click **Next**.
- g. Review the summary and click **Finish**. Monitor the status of the job and ensure it completes successfully.

7.8.3 Uninstalling an enterprise application

To uninstall an enterprise application from the administrative console, use one of the following methods:

1. Click **Applications** → **Application Types** → **WebSphere enterprise applications**.
2. Select the application you want to uninstall and click **Uninstall**.

Or, from the job manager, use the **Uninstall application** job type.

Note: When you uninstall an application and it is the only composite unit in a business-level application, the BLA is also deleted.

7.8.4 Starting an enterprise application

An application starts automatically when the application server to which it is mapped starts. You only need to start an application immediately after installing it, or if you have manually stopped it.

Application startup: Starting an application server starts the applications mapped to that server. The order in which the applications start depends on the startup order you assigned to each of them. The application with the lowest startup order value is started first. Applications that have the same order designation are started in no particular order. Enabling the parallel start option for the application server means to start applications with the same weight in parallel.

To view or change the application startup order, click **Applications** → **Application Types** → **WebSphere enterprise applications**. Open the configuration window for the application by clicking the application name and click **Startup behavior**.

An application can be started by following these steps from the administrative console:

1. Click **Applications** → **Application Types** → **WebSphere enterprise applications**.
2. Select the application you want and click **Start**.

An application can be started from the job manager console by clicking the **Start application** job type.

Note: To start an application, the application server that contains the application has to be started. If not, the application displays in the administrative console as unavailable and you are not able to start it.

7.8.5 Stopping an enterprise application

An application can be stopped using the administrative console.

From the administrative console, complete the following steps:

1. Click **Applications** → **Application Types** → **WebSphere enterprise applications**.
2. Select the application you want to stop and click **Stop**.

Or, from the job manager console, use the **Stop application** job type.

7.8.6 Preventing an enterprise application from starting on a server

By default, an application will start when the server starts. The only way to prevent this occurrence is to disable the application from running on the server.

From the administrative console, complete the following steps:

1. Click **Applications** → **Application Types** → **WebSphere enterprise applications**.
2. Click the application name to open the configuration.
3. Select **Target specific application status** in the Detail Properties section.

4. Select the server for which you want to disable the application.
5. Click the **Disable Auto Start** button.
6. Save the configuration.

7.8.7 Viewing application details

The administrative console does not display the deployed servlets, JSPs, or EJBs directly on the administrative console. However, you can use the administrative console to display XML deployment descriptors for the enterprise application, web modules, and EJB modules.

To view the application deployment descriptor for an application, complete the following steps:

1. Click **Applications** → **Application Types** → **WebSphere enterprise applications**.
2. Click the application name in which you are interested.
3. Under the Configuration tab, click **View Deployment Descriptor** under the Detail Properties section.

Figure 7-45 shows the deployment descriptor window for the DefaultApplication enterprise application. The Configuration tab shows you the structure defined by the deployment descriptor:

- ▶ The name of the enterprise application
- ▶ The web modules and their context roots
- ▶ The EJB modules and their associated JAR files
- ▶ The security roles associated with the enterprise application

```

<application id="Application_ID" >
  <display-name> DefaultApplication.ear </display-name>
  <description> This is the IBM WebSphere Application Server Default Application. </description>
  <module id="WebModule_1" >
    <web>
      <web-uri> DefaultWebApplication.war </web-uri>
      <context-root> / </context-root>
    </web>
  </module>
  <module id="EjbModule_1" >
    <ejb> Increment.jar </ejb>
  </module>
  <security-role id="SecurityRole_1310585783759" >
    <description> All Authenticated users role. </description>
    <role-name> All Role </role-name>
  </security-role>
</application>

```

Figure 7-45 Enterprise application deployment descriptor

Viewing EJB modules

To see the EJBs that are part of an enterprise application, complete the following steps:

1. Click **Applications** → **Application Types** → **WebSphere enterprise applications**.
2. Click the application name in which you are interested.
3. Click **Manage Modules** under the Modules Items section.

4. Click the EJB module name that you want to view (Figure 7-46).

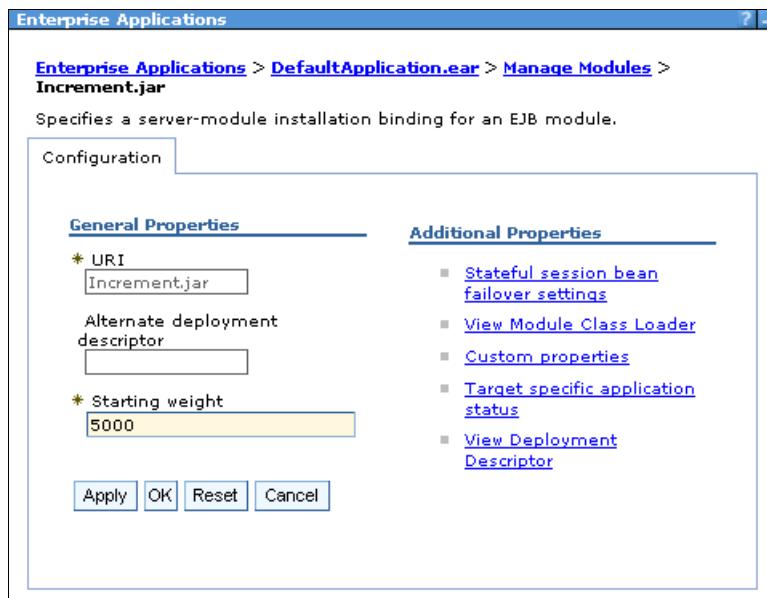


Figure 7-46 Viewing an EJB module configuration

5. Click **View Deployment Descriptor** under Additional Properties to see the EJB deployment descriptor.

Viewing web modules

To see the servlets and JSPs that are part of an enterprise application, complete the following steps:

1. Click **Applications** → **Application Types** → **WebSphere enterprise applications**.
2. Click the application name in which you are interested.
3. Click **Manage Modules** under the Modules Items section.

- Click the web module name you want to view (Figure 7-47).

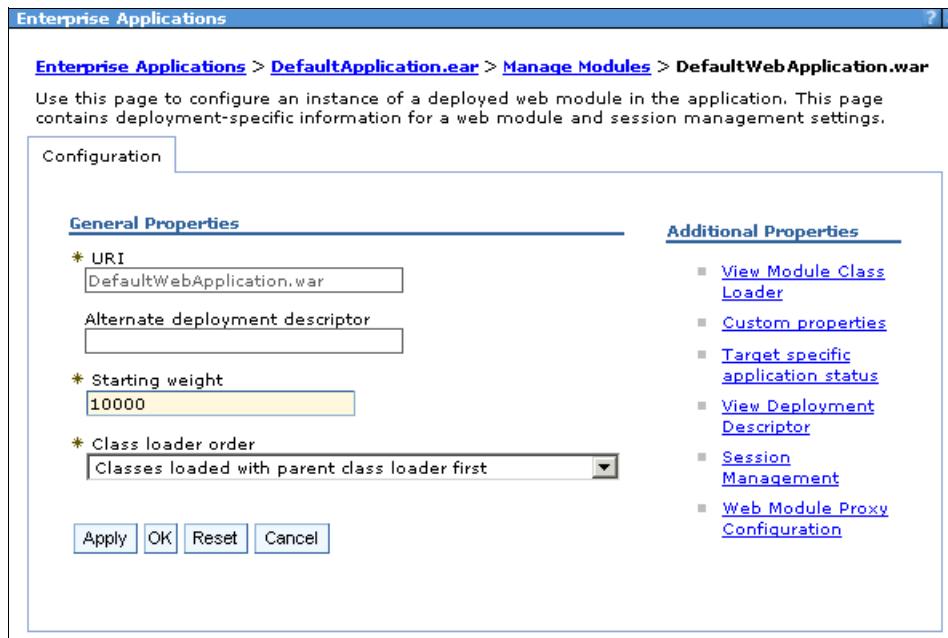


Figure 7-47 View a web module

- Click **View Deployment Descriptor** to see the details of the web module content.

7.8.8 Finding a URL for a servlet or JSP

The URL for a servlet or JSP is the path used to access it from a browser. The URL is partly defined in the deployment descriptor provided in the EAR file and partly defined in the deployment descriptor for the web module containing the servlet or JSP.

To find the URL for a servlet or JSP, complete the following steps:

- Find the context root of the web module containing the servlet.
- Find the URL for the servlet.
- Find the virtual host where the web module is installed.
- Find the aliases by which the virtual host is known.
- Combine the virtual host alias, context root, and URL pattern to form the URL request of the servlet/JSP.

For example, to look up the URL for the snoop servlet, complete the following steps:

- Find the context root of the web module DefaultWebApplication of the DefaultApplication enterprise application. This web module contains the snoop servlet.
 - Click **Applications** → **Application Types** → **WebSphere enterprise applications**.
 - Click the application in which you are interested, in our case, **DefaultApplication.ear**.

- c. On the Configuration tab, click **Context Root for Web Modules** (Figure 7-48) in the Web Module Properties section. You can see that:
- There is only one web module in this application, that is, the Default Web Application.
 - The context root is “/”. We will use this later.

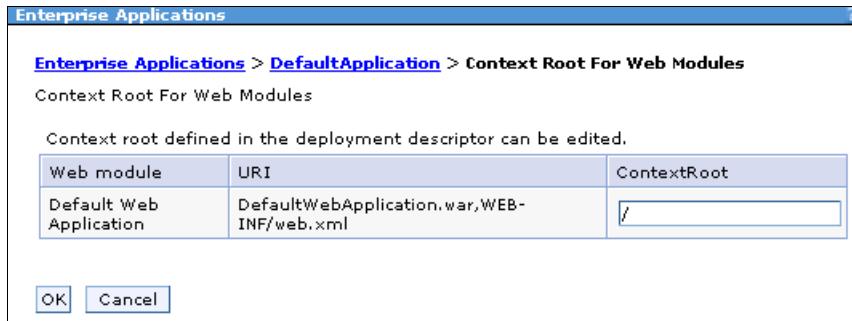


Figure 7-48 Context root for the web modules in DefaultApplication

- Click **OK** to return to the DefaultApplication configuration.
2. Find the URL for the snoop servlet:
- In the DefaultApplication configuration window, select **Manage Modules** in the Modules section.
 - Click the **Default Web Application** web module to see the general properties.
 - Click **View Deployment Descriptor** under Additional Properties. This action displays the web module properties window, as shown in Figure 7-49. Note that the URL pattern for the snoop servlet starting from the web module context root is “/snoop/*”. The web module context root is “/”.



Figure 7-49 DefaultWebApplication web module deployment descriptor

- d. Note that as you navigate through the windows, a navigation path is displayed below the Messages area, which is the bread-crumb trail. Click **DefaultApplication.ear** to return to the application configuration page.
3. Find the virtual host where the web module is installed:
- In the DefaultApplication configuration window, click **Virtual hosts** under the Web Module Properties section. This action will display all of the web modules contained in the enterprise application, and the virtual hosts in which they have been installed, as shown in Figure 7-50. Note that the Default Web Application Web module has been installed on the default_host virtual host.

The screenshot shows the 'Virtual hosts' configuration window. At the top, there is a breadcrumb navigation: Enterprise Applications > DefaultApplication > Virtual hosts. Below the title, a descriptive text states: 'Specify the virtual host where you want to install the Web modules that are contained in your application. You can install Web modules on the same virtual host or disperse them among several hosts.' There is a checkbox labeled 'Apply Multiple Mappings'. A table lists the web modules and their assigned virtual hosts:

Select	Web module	Virtual host
<input type="checkbox"/>	Default Web Application	default_host

Figure 7-50 List of virtual hosts

- From the administration console navigation tree, click **Environment** → **Virtual Hosts**.
 - Click **default_host**.
5. Click **Host Aliases** under Additional Properties. This action shows the list of aliases by which the default_host virtual host is known, as shown in Figure 7-51.

The screenshot shows the 'Host Aliases' configuration window for the default_host virtual host. At the top, there is a breadcrumb navigation: Virtual Hosts > default_host > Host Aliases. Below the title, a descriptive text states: 'Use this page to edit, create, or delete a domain name system (DNS) alias by which the virtual host is known.' There is a checkbox labeled 'Preferences'. A table lists the host aliases and their corresponding ports:

Select	Host Name	Port
<input type="checkbox"/>	*	9080
<input type="checkbox"/>	*	80
<input type="checkbox"/>	*	9443
<input type="checkbox"/>	*	5060
<input type="checkbox"/>	*	5061
<input type="checkbox"/>	*	443

Figure 7-51 Default_host virtual host aliases

Note that the aliases are composed of a DNS host name and a port number. The host aliases for the default_host virtual host are *:80, *:9080, and *:9443, where "*" stands for any host name.

6. Combine the virtual host alias, context root, and URL pattern to form the URL request of the snoop servlet. Requests for the servlet with any of the following URLs will map to the default_host virtual host:
 - `http://hostname:80/snoop`
 - `http://hostname:9080/snoop`
 - `https://hostname:9443/snoop`

7.9 Enabling process restart on failure

WebSphere Application Server does not have either:

- ▶ A nanny process to monitor whether the AdminServer process is running, and restart it if the process has failed
- ▶ An AdminServer process to monitor whether each application server process is running, and restart it if the process has failed

Instead, WebSphere Application Server uses the native operating system functionality to restart a failed process. Refer to the following sections that provide information about your operating system.

7.9.1 Windows

The administrator can choose to register one or more of the WebSphere Application Server processes on a machine as a Windows service during profile creation, or after by using the **WASService** command. With this command, Windows then automatically attempt to restart the service if it fails.

Syntax

Enter **WASService.exe** with no arguments to get a list the valid formats, as shown in Example 7-20.

Example 7-20 WASService command format

```
Usage: WASService.exe (with no arguments displays this help)
      || -add <service name>
          -serverName <Server>
          -profilePath <Server's Profile Directory>
              [-wasHome <WebSphere Install Directory>]
              [-configRoot <Config Repository Directory>]
              [-startArgs <additional start arguments>]
              [-stopArgs <additional stop arguments>]
              [-userid <execution id> -password <password>]
              [-logFile <service log file>]
              [-logRoot <server's log directory>]
              [-encodeParams]
              [-restart <true | false>]
              [-startType <automatic | manual | disabled>]
      || -remove <service name>
      || -start <service name> [optional startServer.bat parameters]
      || -stop <service name> [optional stopServer.bat parameters]
      || -status <service name>
      || -encodeParams <service name>
```

Be aware of the following considerations when using the **WASService** command:

- ▶ When adding a new service, the **-serverName** argument is mandatory. The serverName is the process name. If in doubt, use the **serverstatus -all** command to display the processes. For a deployment manager, the serverName is dmgr, for a node agent it is nodeagent, and for a server, it is the server name.
- ▶ The **-profilePath** argument is mandatory. It specifies the home directory for the profile.
- ▶ Use unique service names. The services are listed in the Windows Services control window as:

IBM WebSphere Application Server V8.0 - <service name>

The convention used by the Profile Management Tool is to use the node name as the service name for a node agent. For a deployment manager, it uses the node name of the deployment manager node concatenated with dmgr as the service name.

Examples

Example 7-21 shows using the **WASService** command to add a node agent as a Windows service.

Example 7-21 Registering a deployment manager as a Windows service

```
C:\WebSphere\AppServer\bin>WASService -add "DeploymentManager01" -serverName dmgr
-profilePath "C:\websphere\appserver\profiles\DeploymentManager01" -restart true
Adding Service: DeploymentManager01
    Config Root: C:\WebSphere\AppServer\profiles\DeploymentManager01\config
    Server Name: nodeagent
    Profile Path: C:\WebSphere\AppServer\profiles\DeploymentManager01
    Was Home: C:\WebSphere\AppServer\
    Start Args:
    Restart: 1
IBM WebSphere Application Server V8.0 - DeploymentManager01 service successfully
added.
```

Note that the service name added in Figure 7-52 will be IBM WebSphere Application Server V8.0, concatenated with the name you specified for the service name.

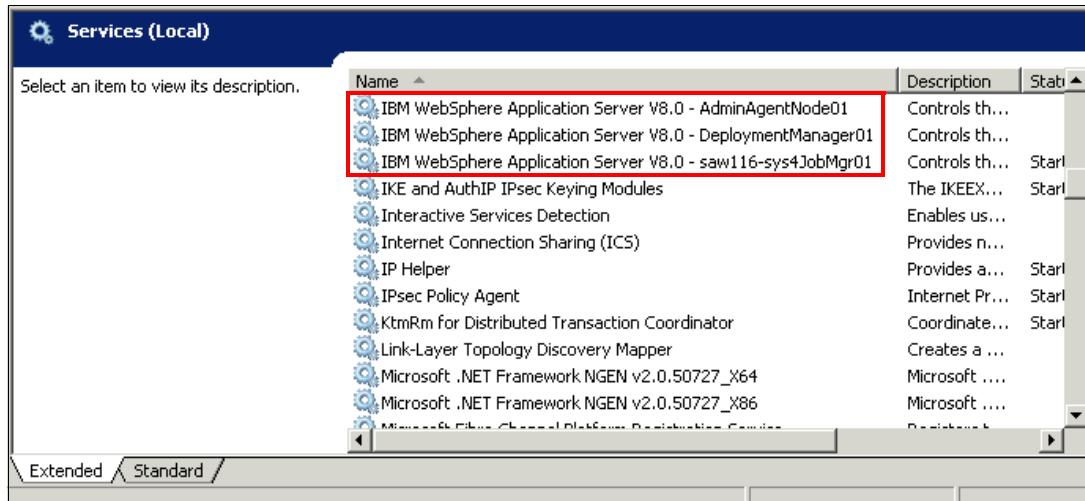


Figure 7-52 New service

If you remove the service using the **WASService -remove** command, specify only the latter portion of the name, as shown in Example 7-22.

Example 7-22 Removing a service

```
C:\WebSphere\AppServer\bin>WASService -remove "DeploymentManager01"
Remove Service: DeploymentManager01
Successfully removed service
```

7.9.2 UNIX and Linux

The administrator can choose to include entries in inittab for one or more of the WebSphere Application Server processes on a machine, as shown in Example 7-23. Each such process will then be automatically restarted if it has failed.

Example 7-23 Inittab contents for process restart

On deployment manager machine:

```
ws1:23:respawn:/usr/WebSphere/DeploymentManager/bin/startManager.sh
```

On node machine:

```
ws1:23:respawn:/usr/WebSphere/AppServer/bin/startNode.sh
ws2:23:respawn:/usr/WebSphere/AppServer/bin/startServer.sh nodename_server1
ws3:23:respawn:/usr/WebSphere/AppServer/bin/startServer.sh nodename_server2
ws4:23:respawn:/usr/WebSphere/AppServer/bin/startServer.sh nodename_server2
```

Note: When setting the action for **startServer.sh** to respawn in /etc/inittab, be aware that **init** will always restart the process, even if you intended for it to remain stopped. As an alternative, you can use the **rc.was** script located in \${WAS_HOME}/bin, which allows you to limit the number of retries.

The best solution is to use a monitoring product that implements notification of outages and logic for automatic restart.

7.9.3 z/OS

WebSphere for z/OS takes advantage of the z/OS Automatic Restart Management (ARM) to recover application servers. Each application server running on a z/OS system (including servers you create for your business applications) are automatically registered with an ARM group. Each registration uses a special element type called SYSCB, which ARM treats as restart level 3, ensuring that RRS (It is a z/OS facility that provides two-phase sync point support across participating resource managers) restarts before any application server.

Note: If you have an application that is critical for your business, you need facilities to manage failures. z/OS provides rich automation interfaces, such as automatic restart management, that you can use to detect and recover from failures. The automatic restart management handles the restarting of servers when failures occur.

Some important things to consider when using automatic restart management:

- ▶ If you have automatic restart management (ARM) enabled on your system, you might want to disable ARM for the WebSphere Application Server for z/OS address spaces before you install and customize WebSphere Application Server for z/OS. During customization, job errors might cause unnecessary restarts of the WebSphere Application Server for z/OS address spaces. After installation and customization, consider enabling ARM.
- ▶ If you are ARM-enabled and you cancel or stop a server, it will restart in place using the **armrestart** command.
- ▶ It is a good idea to set up an ARM policy for your deployment manager and node agents. For more information about how to change the ARM policies, refer to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.installation.zseries.doc/info/zseries/ae/cins_changearm.html

- ▶ If you start the location service daemon on a system that already has one, it will terminate.
- ▶ Every other server will come up on a dynamic port unless the configuration has a fixed port. Therefore, the fixed ports must be unique in a sysplex.
- ▶ If you issue STOP, CANCEL, or MODIFY commands against server instances, be aware of how automatic restart management behaves regarding WebSphere Application Server for z/OS server instances; Table 7-1 depicts the behavior of ARM regarding WebSphere Application Server for z/OS server instances.

Table 7-1 Behavior of ARM and WebSphere Application Server for z/OS server instances

When you issue	ARM behavior
STOP address_space	It will not restart the address space.
CANCEL address_space	It will not restart the address space.
CANCEL address_space, ARMRESTART	It will restart the address space.
MODIFY address_space, CANCEL	It will not restart the address space.
MODIFY address_space, CANCEL,ARMRESTART	It will restart the address space.

For more information about how to activate the ARM, refer to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=/com.ibm.websphere.installation.zseries.doc/info/zseries/ae/tins_activearm.html

Let us say you have activated ARM and want to check the status of address spaces registered for automatic restart management; to get this information, you need to:

1. Initialize all servers.

2. Issue one or both of the commands shown in Example 7-24.

Example 7-24 Displaying the status of address spaces registered for automatic restart management

To display all registered address spaces (including the address spaces of server instances), issue the command:

```
d xcf,armstatus,detail
```

To display the status of a particular server instance, use the display command and identify the job name. For example, to display the status of the Daemon server instance (job BBODMN), issue the following command:

```
d xcf,armstatus,jobname=bbodmn,detail
```

For more information about how to use the **display** command, refer to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=/com.ibm.websphere.nd.multiplatform.doc/info/ae/ae/rxml_mvsdisplay.html



Administration with scripting

The administrative console is sufficient for tasks that are non-repetitive, have a minimal number of administrative steps, and are relatively simple. For administration that requires many steps, which can be repetitive, and time consuming to configure, `wsadmin` combined with scripts is an ideal tool.

In this chapter, we introduce the `wsadmin` scripting solution and describe how you can use it to perform basic tasks.

This chapter contains the following topics:

- ▶ Overview of WebSphere scripting
- ▶ Launching `wsadmin`
- ▶ Command and script invocation
- ▶ The `wsadmin` tool management objects
- ▶ Managing WebSphere using script libraries
- ▶ Assistance with scripting
- ▶ Example: Using scripts with the job manager
- ▶ Online resources

8.1 Overview of WebSphere scripting

WebSphere Application Server provides a scripting interface based on the *Bean Scripting Framework (BSF)*. This interface is called **wsadmin**. BSF is an open source project that is used to implement an architecture for incorporating scripting into Java applications and applets. The BSF architecture works as an interface between Java applications and scripting languages. Using this framework allows scripting languages to complete the following tasks:

- ▶ Look up a preregistered bean and access a predeclared bean.
- ▶ Register a newly created bean.
- ▶ Perform all bean operations.
- ▶ Bind events to scripts in the scripting language.

Because **wsadmin** uses BSF, it can make various Java objects available through language-specific interfaces to scripts. Figure 8-1 shows the major components that are involved in the **wsadmin** scripting solution.

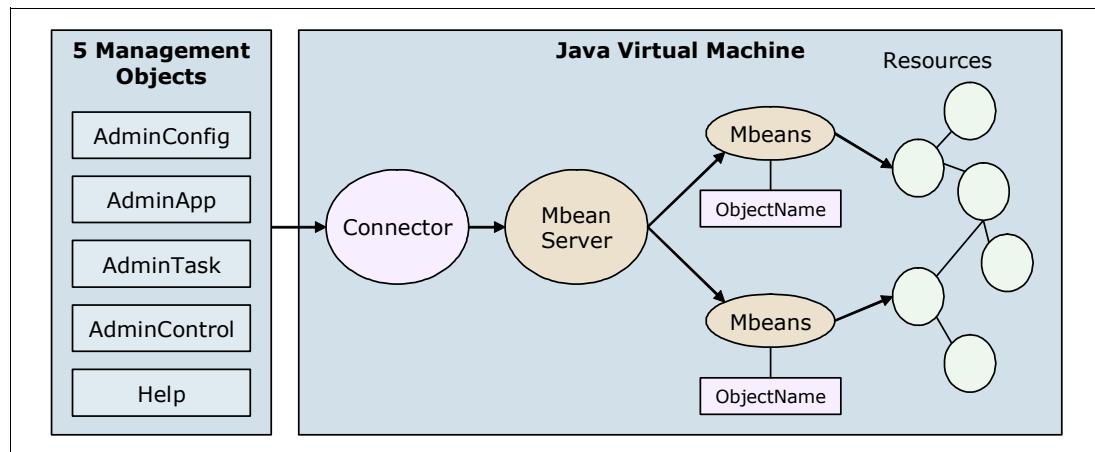


Figure 8-1 The wsadmin scripting

You can use the following programming language to write **wsadmin** scripts:

- ▶ Jython
- ▶ Jacl

With WebSphere Application Server V7, the deprecation process for the Jacl syntax begins. The script library and the command assistance on the administrative console support only Jython.

If you have existing Jacl scripts and want to start migrating to Jython, you can use the Jacl-to-Jython conversion utility to convert the scripts. This conversion assistant typically achieves 95-98% of a preliminary conversion. In most cases, the resulting conversion is syntactically and runtime equivalent.

However, you should verify each line to ensure that the code functions as you originally intended. When Jacl and Jython language differences result in lines of code that are difficult to convert automatically, the converted lines are flagged with a #?PROBLEM? tag. This tag provides assistance in finding areas that are most likely in need of manual conversion.

8.2 Launching wsadmin

The **wsadmin** command file resides in the **bin** directory of every profile. Start **wsadmin** from a command prompt with the following command (as appropriate):

- ▶ UNIX: *profile_root/bin/wsadmin.sh*
- ▶ Windows: *profile_root\bin\wsadmin*

Note that the **wsadmin** command also exists in the **bin** directory of the *install_root* directory. If you start **wsadmin** from this location, you must be careful to specify the profile to work within the command. If you do not specify the profile (or forget to specify it), the default profile is chosen.

Example 8-1 shows how to start **wsadmin**. In this example, the **wsadmin** command is used to connect to the job manager. It is issued from the **bin** directory of the job manager profile. The profile does not need to be specified. The **-lang** argument indicates Jython is used (Jacl is the default).

Example 8-1 Flexible management - The wsadmin command line

```
C:\WebSphereV8\AppServer\profiles\jmgr40\bin>wsadmin -lang jython
WASX7209I: Connected to process "jobmgr" on node jmgr40node using SOAP connector
; The type of process is: JobManager
WASX7031I: For help, enter: "print Help.help()"
wsadmin>
```

To get syntax-related help, use **wsadmin -?** or **-help** (see Example 8-2).

Example 8-2 The wsadmin syntax

```
wsadmin
      [ -h(elp) ]
      [ -? ]
      [ -c <command> ]
      [ -p <properties_file_name>]
      [ -profile <profile_script_name>]
      [ -f <script_file_name>]
      [ -javaoption java_option]
      [ -lang language]
      [ -wsadmin_classpath class_path]
      [ -profileName profile]
      [ -conntype
          SOAP
              [-host host_name]
              [-port port_number]
              [-user userid]
              [-password password] |
          RMI
              [-host host_name]
              [-port port_number]
              [-user userid]
              [-password password] |
          JSR160RMI
              [-host host_name]
              [-port port_number]
              [-user userid]
              [-password password] |
```

```

IPC
    [-ipchost host_name]
    [-port port_number]
    [-user userid]
    [-password password] |
    NONE
]
[ -jobid <jobid_string>]
[ -tracefile <trace_file>]
[ -appendtrace <true/false>]
[ script parameters ]

```

8.2.1 Scripting environment properties file

You can set the properties that determine the scripting environment for **wsadmin** using either the command line or a properties file. Modifying the properties file can be useful when you want to change a default setting, for example, changing the language from Jacl to Jython.

You can set properties in the following locations:

- ▶ The installation default properties file for the profile, which is located in the following directory:
profile_root/properties/wsadmin.properties
- ▶ A user default properties file, which is located in the Java `user.home` property.
- ▶ A customized properties file placed in the location that is pointed to by the `WSADMIN_PROPERTIES` environment variable.
- ▶ A customized properties file, which is pointed to using the `-p` argument to the **wsadmin** command.

When **wsadmin** is started, properties are loaded from these files in the order listed here. The properties file that is loaded last overrides any property files loaded earlier.

Table 8-1 lists the properties.

Table 8-1 The wsadmin properties

Property	Value
<code>com.ibm.ws.scripting.connectionType</code>	SOAP, RMI or NONE
<code>com.ibm.scripting.port</code>	TCP port of target system
<code>com.ibm.scripting.host</code>	Host name of target system
<code>com.ibm.ws.scripting.defaultLang</code>	Jython or Jacl
<code>com.ibm.ws.scripting.echoparams</code>	Determines whether parameters or arguments are output to STDOUT or to the wsadmin trace file
<code>com.ibm.ws.scripting.traceFile</code>	File for trace information
<code>com.ibm.ws.scripting.validationOutput</code>	Location of validation reports
<code>com.ibm.ws.scripting.traceString</code>	=com.ibm.*=all=enabled
<code>com.ibm.ws.scripting.appendTrace</code>	Appends to the end of the existing log file
<code>com.ibm.ws.scripting.profiles</code>	List of profiles to be run before running user commands, scripts, or an interactive shell

Property	Value
com.ibm.ws.scripting.emitWarningForCustomSecurityPolicy	Controls whether message WASX7207W is emitted when custom permissions are found
com.ibm.ws.scripting.tempdir	Stores temporary files when installing applications
com.ibm.ws.scripting.validationLevel	Level of validation to use when configuration changes are made from the scripting interface
com.ibm.ws.scripting.crossDocumentValidationEnabled	Determines whether the validation mechanism examines other documents when changes are made to one document
com.ibm.ws.scripting.classpath	List of paths to search for classes and resources

Some of the listed properties in the wsadmin.properties file are commented out by default. An example is com.ibm.ws.scripting.traceString. If you want to trace **wsadmin** execution, remove the comment sign (#) from the properties file.

Some of the properties contain default values. For example, com.ibm.ws.scripting.connectionType has a default value of SOAP. Thus, when a scripting process is invoked, a SOAP connector is used to communicate with the server. The com.ibm.ws.scripting.defaultLang property is set to Jacl.

Use the **-p** option to specify a customized properties file. Example 8-3 shows sample coding for invoking **wsadmin** to execute a script file using a specific properties file.

Example 8-3 Specifying properties file on the command line

```
C:\WebSphereV8\AppServer\profiles\dmgr40\bin>wsadmin -p C:\Webspherrev8\Appserver
\profiles\dmgr40\properties\wsadmin_custom.properties
WASX7209I: Connected to process "dmgr" on node dmgr40node using SOAP connector;
The type of process is: DeploymentManager
WASX7031I: For help, enter: "print Help.help()"
```

8.2.2 Script profile file

A *script profile* is a script that is invoked before the main script or before invoking **wsadmin** in interactive mode. The purpose of the script profile is to customize the environment in which the script runs. For example, a script profile can be set for the Jacl scripting language that makes Jacl-specific variables or procedures available to the interactive session or main script.

Use the **-profile** command-line option to specify a profile script. Several **-profile** options can be used on the command line and are invoked in the order given.

8.2.3 Connected versus local mode

The **wsadmin** command can operate in either connected or local mode. In connected mode, all operations are performed by method invocations on running JMX MBeans. In local mode, the application server (MBeans server) is not started and the **wsadmin** objects are limited to configuring the server by means of directly manipulating XML configuration documents.

When operating in local mode, be sure that you are operating on the correct profile, either using the **-profileName** argument or starting **wsadmin** from the profile/bin directory.

When performing configuration changes in local mode in a distributed server environment, care should be taken to make configuration changes at the deployment manager level. Changes made directly to the node configuration will be lost at server startup or at configuration replication.

Use the `-conntype NONE` option to run in local mode.

8.3 Command and script invocation

The `wsadmin` commands can be invoked in three different ways. This section describes how to invoke the command.

Note: For simplicity, the examples in this chapter assume the following facts:

- ▶ `wsadmin` is executed from the `profile_root/bin` directory. It is not necessary to specify the profile name, host, and port.
- ▶ Administrative security is disabled. In reality, you need to specify the user name and password when you invoke `wsadmin`.

8.3.1 Invoking a single command (-c)

You can use the `-c` option to execute a single command using `wsadmin`, as shown in Example 8-4. In this example, we use the `AdminControl` object to query the node name of the WebSphere server process.

Example 8-4 Running a single command in wsadmin

```
C:\WebSphereV8\AppServer\profiles\jmgr40\bin>wsadmin -lang jython -c
AdminControl.getNode()
WASX7209I: Connected to process "jobmgr" on node jmgr40node using SOAP connector
; The type of process is: JobManager
'jmgr40node'

C:\WebSphereV8\AppServer\profiles\jmgr40\bin>
```

8.3.2 Running script files (-f)

You can use the `-f` option to execute a script file. Example 8-5 shows a two-line Jython script named `myScript.py`. The script has a `.py` extension to reflect the Jython language syntax of the script. The extension plays no significance in `wsadmin`; the `com.ibm.ws.scripting.defaultLang` property or `-lang` parameter is used to determine the language used. If the property setting is not correct, use the `-lang` option to identify the scripting language, because the default is Jacl.

Example 8-5 Jython script

```
print "This is an example Jython script"
print "+ AdminControl.getNode()""
```

Example 8-6 shows how to execute the script.

Example 8-6 Running a Jython script in wsadmin

```
C:\WebSphereV8\AppServer\profiles\dmgr40\bin>wsadmin -f myScript.py -lang jython
```

```
WASX7209I: Connected to process "dmgr" on node dmgr40node using SOAP connector;
The type of process is: DeploymentManager
This is an example Jython script
dmgr40node
```

8.3.3 Invoking commands interactively

You can run the command execution environment in interactive mode, so that you can invoke multiple commands without incurring the impact of starting and stopping the **wsadmin** environment for every single command. Run the **wsadmin** command without the command (-c) or script file (-f) options to start the interactive command execution environment, as shown in Example 8-7.

Example 8-7 Starting the wsadmin interactive command execution environment

```
C:\WebSphereV8\AppServer\profiles\dmgr40\bin>wsadmin -lang jython
WASX7209I: Connected to process "dmgr" on node dmgr40node using SOAP connector;
The type of process is: DeploymentManager
WASX7031I: For help, enter: "print Help.help()"
wsadmin>
```

From the **wsadmin>** prompt, you can invoke the WebSphere administrative objects and built-in language objects, as shown in Example 8-8. Enter the commands at the **wsadmin>** prompt.

Example 8-8 Interactive command invocation

```
wsadmin>AdminControl.getNode()
'dmgr40node'
wsadmin>
```

End the interactive execution environment by typing **quit**, and then pressing the Enter key.

8.4 The wsadmin tool management objects

The **wsadmin** tool has the following administrative objects that provide server configuration and management capabilities:

- ▶ Help
- ▶ AdminControl
- ▶ AdminConfig
- ▶ AdminApp
- ▶ AdminTask

This section provides information about how to use the script libraries and command assist. For information about how to use management objects, refer to the Information Center found at the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=%2Fcom.ibm.websphere.nd.multiplatform.doc%2Finfo%2Fae%2Fxml_launchscript.html&resultof=%22wsadmin%22

8.4.1 Help

The Help object provides a quick way to get information about methods, operations, and attributes while using scripting. For example, to get a list of the public methods that are available for the AdminControl object, enter the following command:

```
wsadmin>print Help.AdminConfig()
```

To get a detailed description of a specific object method and the parameters that it requires, invoke the help method of the target object with the method name as the option to the help method, as shown in Example 8-9.

Example 8-9 AdminConfig.help scripting

```
wsadmin>print AdminConfig.help("createClusterMember")
WASX7284I: Method: createClusterMember
    Arguments: cluster id, node id, member attributes

    Description: Creates a new Server object on the node specified
    by "node id." This Server is created as a new member of the existing
    cluster specified by "cluster id," and has attributes specified in
    "member attributes." One attribute is required: "memberName."
    The Server is created using the default template for
    Server objects, and has the name and specified by the
    "memberName" attribute.

Method: createClusterMember

Arguments: cluster id, node id, member attributes, template id

Description: Creates a new Server object on the node specified
by "node id." This Server is created as a new member of the existing
cluster specified by "cluster id," and has attributes specified in
"member attributes." One attribute is required: "memberName."
The Server is created using the Server template
specified by "template id," and has the name specified by
the "memberName" attribute.
```

The AdminTask object also supports searching for the specific command by using a wildcard character under help. For example, Example 8-10 shows the command to get a list of the command names that start with the term *create*.

Example 8-10 AdminTask.help scripting

```
wsadmin>print AdminTask.help("-commands", "create*")
WASX8004I: Available admin commands:

createAllActivePolicy - Create a policy that automatically activates all group members.
```

```
createApplicationServer - Command that creates a server  
createApplicationServerTemplate - creates a server Template based on a server configuration  
createAuditEncryptionConfig - Configures audit record encryption.  
createAuditEventFactory - Creates an entry in the audit.xml to reference the configuration of a Factory interface.  
createAuditFilter - Creates an entry in the audit.xml to reference an Audit Specification. Enab  
createAuditKeyStore - Creates a new Key Store.  
createAuditNotification - Configures an audit notification.  
createAuditNotificationMonitor - Configures an audit notification monitor.  
createAuditSelfSignedCertificate - Create a new self-signed certificate and store it in a keys  
....
```

8.4.2 AdminControl

The AdminControl object is used for operational control. It communicates with MBeans that represent live objects running a WebSphere server process. It includes commands to query existing running objects and their attributes and to invoke operations on the objects. In addition to the operational commands, the AdminControl object supports commands to query information about the connected server, to trace clients, to reconnect to a server, and to start and stop a server.

Note that because the AdminControl object operates on live MBeans, you cannot use it to start a deployment manager, node agent, or stand-alone application server.

8.4.3 AdminConfig

The AdminConfig object is used to manage the configuration information that is stored in the repository. This object communicates with the WebSphere Application Server configuration service component to make configuration inquiries and changes. You can use it to query existing configuration objects, create configuration objects, modify existing objects, and remove configuration objects. In a distributed server environment, the AdminConfig commands are available only if a scripting client is connected to the deployment manager. When connected to a node agent or a managed application server, the AdminConfig commands will not be available because the configuration for these server processes are copies of the master configuration that resides in the deployment manager.

8.4.4 AdminApp

The AdminApp object can update application metadata, map virtual hosts to web modules, and map servers to modules for applications already installed. Changes to an application, such as specifying a library for the application to use or setting session management configuration properties, are performed using the AdminConfig object.

8.4.5 AdminTask

The AdminTask object is used to access a set of task-oriented administrative commands that provide an alternative way to access the configuration commands and the running object management commands. The administrative commands run simple and complex commands. The administrative commands are discovered dynamically when the scripting client is started. The set of available administrative commands depends on the edition of WebSphere Application Server you install. You can use the AdminTask object commands to access these commands.

Two run modes are always available for each administrative command, namely the batch and interactive mode. When you use an administrative command in interactive mode, you go through a series of steps to collect your input interactively. This process provides users a text-based wizard and a similar user experience to the wizard in the administrative console. You can also use the help command to obtain help for any of the administrative commands and the AdminTask object.

8.4.6 Properties file based configuration

Using complex scripts requires knowledge of the Jacl or Jython scripting languages and the public MBean interfaces. The use of a properties file based configuration provides a way to simplify administrative tasks using `wsadmin`.

The following commands in the PropertiesBasedConfiguration command group implement this type of configuration:

- ▶ `extractConfigProperties`
- ▶ `validateConfigProperties`
- ▶ `applyConfigProperties`
- ▶ `deleteConfigProperties`
- ▶ `createPropertiesFileTemplates`

Properties file based configuration extracts configuration data to one file that is easy to read using any editor tool. The configuration attributes are provided as key/value pairs, as shown in Figure 8-2.

```
#  
# SubSection 1.0 # JDBCProvider attributes  
#  
ResourceType=JDBCProvider  
ImplementingResourceType=JDBCProvider  
ResourceId=Cell={!{cellName}}:JDBCProvider=ID#builtin_jdbcprovider  
#  
  
#  
#Properties  
#  
classpath=${DERBY_JDBC_DRIVER_PATH}/derby.jar  
name=Derby JDBC Provider (XA)  
implementationClassName=org.apache.derby.jdbc.EmbeddedXADataSource  
nativepath={}  
description=Built-in Derby JDBC Provider (XA)  
providerType=Derby JDBC Provider (XA) #readonly  
xa=true #boolean
```

Figure 8-2 Example - Properties for a JDBC provider

1. Resource type and Identifier

2. Configuration Information

After the properties are extracted, you can make any necessary changes to the attributes and validate the changes before applying them to the server. You can also create or delete configuration objects.

Example 8-11 shows samples of these commands.

Example 8-11 The wsadmin commands for properties based configuration

```
AdminTask.extractConfigProperties('-configData Node=myNode -propertiesFileName  
myNodeProperties.props')
```

```

AdminTask.validateConfigProperties('-propertiesFileName myNodeProperties.props
-reportFile report.txt')

AdminTask.applyConfigProperties('-propertiesFileName myPropFile.props -validate
true')

AdminTask.deleteConfigProperties('-propertiesFileName myPropFile.props')

AdminTask.createPropertiesFileTemplates('-propertiesFileName serverTemplate.props
-configType Server')

```

Because it is not possible to modify every configuration using properties file based configuration, you should *not* use this tool for backup and recovery. Use the **BackupConfig** and **RestoreConfig** commands found in the <was_home>/bin directory as the main backup and recovery tools.

8.5 Managing WebSphere using script libraries

You can use script libraries to perform a higher level of **wsadmin** functions than when using a single **wsadmin** command. Only a single line from a library function is needed to perform complex functions. Each script is written in Jython and is often referred to as *the Jython script*. The script libraries are categorized into six types, and the types are subdivided, as listed in Table 8-2.

Python script files are supplied for each Jython class file. The Python files can be read as a text files. Table 8-2 shows the scripts and the type of resource they manage. Each script has a set of procedures that perform specific functions.

Table 8-2 The types of script libraries

TYPE	Python (Jython) script file
Application	AdminApplication AdminBLA
Resources	AdminJ2C AdminJDBC AdminJMS AdminResources
Security	AdminAuthorizations
Servers	AdminClusterManagement AdminServerManagement
System	AdminNodeGroupManagement AdminNodeManagement
Utilities	AdminLibHelp AdminUtilities

Script libraries, their procedures, and syntax: You can find more information about these script libraries in the WebSphere Application Server Information Center at the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.nd.multiplatform.doc/info/ae/ae/we1c_ref_adm_jython.html

8.5.1 Invoking script libraries

The script libraries are located in the *install_root* /scriptlibraries directory. These libraries are loaded when **wsadmin** starts and are readily available from the **wsadmin** command prompt or to be used from the customized scripts. You can invoke the scripts in interactive mode or script mode.

Interactive mode

Interactive mode is suitable for simple tasks and testing. Using this mode allows you to get the results directly. To invoke a script interactively, start a **wsadmin** session and enter the script name, procedure, and arguments at the **wsadmin>** prompt (see Example 8-12).

Example 8-12 Using the Jython scripts in interactive mode

```
C:\WebSphere\AppServer\bin>wsadmin -lang jython
WASX7209I: Connected to process "dmgr" on node DmgrNode02 using SOAP connector;
The type of process is: DeploymentManager
WASX7031I: For help, enter: "print Help.help()"
wsadmin>AdminServerManagement.checkIfServerExists("sys1Node01", "Amember01")
-----
AdminServerManagement: Check if server exists
Node name: sys1Node01
Server name: Amember01
Usage: AdminServerManagement.checkIfServerExists("sys1Node01", "Amember01")
-----
'true'
wsadmin>
```

Script file mode (-f)

Using a script file with **wsadmin** is useful when you want to have daily tasks performed automatically or if you need to manage multiple servers. To run in script mode, you select the script libraries to use and merge them into your own script file. Save the custom script as a Python file, and run it from the command line.

Example 8-13 shows a Python file containing two script library commands.

Example 8-13 The test.py script

```
# Writting by Jython
# Script file name : test.py

AdminServerManagement.checkIfServerExists("sys1Node01", "Amember21")
AdminServerManagement.createApplicationServer("sys1Node01", "Amember21")
```

Example 8-14 shows how to invoke the script.

Example 8-14 The invoke test.py script

```
C:\WebSphere\AppServer\bin>  
C:\WebSphere\AppServer\bin>wsadmin.bat -lang jython -f C:\temp\test.py
```

Customizing scripts

Customizing scripts is an advanced use of the script mode. You can add customized code written in Python or Jython to your script file (the one that calls the script libraries).

Note: Do *not* modify the script libraries. If you need to customize the scripts, you can copy parts of the library code to other files and modify the copied code to improve it or to better suit your needs.

When customizing a script, it is best to use the following steps:

1. Run each Jython script in interactive mode, and then verify the syntax and the result.
2. Create the script file that will call the script libraries by combining all Jython scripts. Verify that the results are as you expect.
3. Add your additional **wsadmin** commands, written in Python, and then verify that the customized script does the work that you intended.

8.5.2 Displaying help for script libraries

You can use the AdminLibHelp script to display each script within a script library. For example, the following command displays each script in the AdminApplication script library:

```
print AdminLibHelp.help("AdminApplication")
```

You can use the help script to display detailed descriptions, arguments, and usage information for a specific script. For example, the following command displays detailed script information for the listApplications script in the AdminApplication script library:

```
print AdminApplication.help('listApplications')
```

Example 8-15 shows sample code for displaying help information for the createApplicationServer procedure in the AdminServerManagement script.

Example 8-15 Help information for a procedure in createApplicationServer

```
wsadmin>print AdminServerManagement.help('createApplicationServer')  
WASL2004I: Procedure: createApplicationServer
```

Arguments: nodeName, serverName, (Optional) templateName

Description: Create a new application server

Usage: AdminServerManagement.createApplicationServer(nodeName,
serverName, templateName)
wsadmin>

8.5.3 Application script library

The application scripts provide a set of procedures to manage and configure enterprise applications and business level applications. You can use these scripts individually, or you can combine them in a custom script file to automate application installation, configuration, and management tasks.

The library that is located in the `install_root/scriptlibraries/application/V70` directory contains the following scripts:

- ▶ AdminApplication, which provides procedures to manage enterprise applications
- ▶ AdminBLA, which provides procedures to manage business level applications

We provide information about the AdminApplication script in the next section. For information about the AdminBLA script, see the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.nd.multiplatform.doc/info/ae/ae/rxml_7libbla.html

AdminApplication script

The AdminApplication script contains procedures that allow you to manage enterprise applications. The AdminApplication script uses the following syntax:

`AdminApplication.procedure_name(arguments)`

The following list summarizes the script procedures that are available with this script. The name of the procedure suggests the function that the procedure provides.

- ▶ Install and uninstall applications:
 - `installAppWithAppNameOption`
 - `installAppWithDefaultBindingOption`
 - `installAppWithNodeAndServerOptions`
 - `installAppWithClusterOption`
 - `installAppModulesToSameServerWithMapModulesToServersOption`
 - `installAppModulesToDiffServersWithMapModulesToServersOption`
 - `installAppModulesToSameServerWithPatternMatching`
 - `installAppModulesToDiffServersWithPatternMatching`
 - `installAppModulesToMultiServersWithPatternMatching`
 - `installAppWithTargetOption`
 - `installAppWithDeployEjbOptions`
 - `installAppWithVariousTasksAndNonTasksOptions`
 - `installWarFile`
 - `uninstallApplication`
- ▶ Update applications:
 - `addSingleFileToAnAppWithUpdateCommand`
 - `addSingleModuleFileToAnAppWithUpdateCommand`
 - `addUpdateSingleModuleFileToAnAppWithUpdateCommand`
 - `addPartialAppToAnAppWithUpdateCommand`
 - `deleteSingleFileToAnAppWithUpdateCommand`
 - `deleteSingleModuleFileToAnAppWithUpdateCommand`
 - `deletePartialAppToAnAppWithUpdateCommand`
 - `updateApplicationUsingDefaultMerge`
 - `updateApplicationWithUpdateIgnoreNewOption`
 - `updateApplicationWithUpdateIgnoreOldOption`
 - `updateEntireAppToAnAppWithUpdateCommand`
 - `updatePartialAppToAnAppWithUpdateCommand`

- updateSingleFileToAnAppWithUpdateCommand
- updateSingleModuleFileToAnAppWithUpdateCommand
- ▶ Export applications:
 - exportAnAppToFile
 - exportAllApplicationsToDir
 - exportAnAppDDLToDir
- ▶ Configure application deployment characteristics:
 - configureStartingWeightForAnApplication
 - configureClassLoaderPolicyForAnApplication
 - configureClassLoaderLoadingModeForAnApplication
 - configureSessionManagementForAnApplication
 - configureApplicationLoading
 - configureLibraryReferenceForAnApplication
 - configureEJBModulesOfAnApplication
 - configureWebModulesOfAnApplication
 - configureConnectorModulesOfAnApplication
- ▶ Start and stop enterprise and business level applications:
 - startApplicationOnSingleServer
 - startApplicationOnAllDeployedTargets
 - startApplicationOnCluster
 - stopApplicationOnSingleServer
 - stopApplicationOnAllDeployedTargets
 - stopApplicationOnCluster
- ▶ Query applications:
 - checkIfAppExists
 - getAppDeployedNodes
 - getAppDeploymentTarget
 - getTaskInfoForAnApp
 - listApplications
 - listApplicationsWithTarget
 - listModulesInAnApp

Example: Installing an application

You can use multiple procedures to install applications. When preparing to install an application, determine the options that you need and choose the procedure accordingly.

The most basic procedure is `installAppwithNodeAndServerOptions`. If the installation is successful, the `installed` application successfully message returns.

This procedure uses the following syntax:

```
AdminApplication.installAppWithNodeAndServerOptions(app_name, app_location,
node_name, app_server_Name)
```

Example 8-16 illustrates this procedure.

Example 8-16 Installing the application script library

```
wsadmin>AdminApplication.installAppWithNodeAndServerOptions("IBMUTC",
"C:/IBMUTC.ear", "sys1Node01", "Amember01")
```

AdminApplication:	Install application with -node and -server options
Application name:	IBMUTC

```

Ear file to deploy:      C:/IBMUTC.ear
Node name:              sys1Node01
Server name:             Amember01
Usage: AdminApplication.installAppWithNodeAndServerOptions("IBMUTC", "C:/IBMUTC
.ear", "sys1Node01", "Amember01")
-----
WASX7327I: Contents of was.policy file:grant codeBase "file:${application}"
{permission java.security.AllPermission;
};

ADMA5016I: Installation of IBMUTC started.
ADMA5058I: Application and module versions are validated with versions of deployment targets.

CWSAD0040I: The application IBMUTC is configured in the Application Server repository.
ADMA5113I: Activation plan created successfully.
ADMA5011I: The cleanup of the temp directory for application IBMUTC is complete.

ADMA5013I: Application IBMUTC installed successfully.
OK: installAppWithNodeAndServerOptions('IBMUTC', 'C:/IBMUTC.ear', 'sys1Node01',
'Amember01', 'false'):

```

Example: Starting an application

Multiple procedures are also available to start an application. To start the application, choose the most suitable script.

The startApplicationSingleServer procedure starts a single application on a single application server. This procedure uses the following syntax:

```
AdminApplication.startApplicationOnSingleServer(app_name, node_name,
app_server_name)
```

Example 8-17 illustrates this procedure.

Example 8-17 Starting an application script library

```

wsadmin>AdminApplication.startApplicationOnSingleServer("IBMUTC","sys1Node01","Ame
mber01")
-----
AdminApplication:      Start an application on a single server
Application name:     IBMUTC
Node name:             sys1Node01
Server name:           Amember01
Usage: AdminApplication.startApplicationOnSingleServer("IBMUTC",
"sys1Node01","Amember01")
-----
OK: startApplicationOnSingleServer('IBMUTC', 'sys1Node01', 'Amember01', 'false')
:1

```

8.5.4 Resource script library

The Resource script library provides a set of scripts to manage WebSphere resources. The library provides script functions for J2C resources, JDBC providers, and JMS resources at the server scope. If you need to configure resources at the cell, node, or cluster level, you can customize the scripts for this purpose.

The script library is located in the `install_root/scriptlibraries/resources/V70` directory. It contains the following scripts and procedures:

► AdminJ2C script:

The following procedures allow you to configure J2C resources:

- `createJ2CActivationSpec`
- `createJ2CAdminObject`
- `createJ2CConnectionFactory`
- `installJ2CResourceAdapter`

The following procedures allow you to query J2C resources:

- `listAdminObjectInterfaces`
- `listConnectionFactoryInterfaces`
- `listJ2CActivationSpecs`
- `listJ2CAdminObjects`
- `listJ2CConnectionFactories`
- `listJ2CResourceAdapters`
- `listMessageListenerTypes`

► AdminJDBC script:

The following procedures allow you to configure JDBC resources:

- `createDataSource`
- `createDataSourceUsingTemplate`
- `createJDBCProvider`
- `createJDBCProviderUsingTemplate`

The following procedures allow you to query JDBC resources:

- `listDataSources`
- `listDataSourceTemplates`
- `listJDBCProviders`
- `listJDBCProviderTemplates`

► AdminJMS script:

The following procedures allow you to configure JMS resources:

- `createGenericJMSSConnectionFactory`
- `createGenericJMSSConnectionFactoryUsingTemplate`
- `createGenericJMSSDestination`
- `createGenericJMSSDestinationUsingTemplate`
- `createJMSProvider`
- `createJMSProviderUsingTemplate`
- `createWASQueue`
- `createWASQueueUsingTemplate`
- `createWASQueueConnectionFactory`
- `createWASQueueConnectionFactoryUsingTemplate`
- `createWASTopic`
- `createWASTopicUsingTemplate`
- `createWASTopicConnectionFactory`
- `createWASTopicConnectionFactoryUsingTemplate`
- `startListenerPort`

The following procedures allow you to query JMS resources:

- `listGenericJMSSConnectionFactories`
- `listGenericJMSSConnectionFactoryTemplates`
- `listGenericJMSSDestinations`
- `listGenericJMSSDestinationTemplates`

- listJMSProviders
- listJMSProviderTemplates
- listWASQueueConnectionFactoryTemplates
- listWASQueueTemplates
- listWASTopicConnectionFactoryTemplates
- listWASQueueConnectionFactories
- listWASQueues
- listWASTopicConnectionFactories
- listWASTopics
- listWASTopicTemplates

► AdminResources:

Use the following script procedures to configure your mail settings:

- createCompleteMailProvider
- createMailProvider
- createMailSession
- createProtocolProvider

Use the following script procedures to configure your resource environment settings:

- createCompleteResourceEnvProvider
- createResourceEnvEntries
- createResourceEnvProvider
- createResourceEnvProviderRef

Use the following script procedures to configure your URL provider settings:

- createCompleteURLProvider
- createURL

Use the following script procedures to configure additional Java Enterprise Edition (JEE) resources:

- createJAASAuthenticationAlias
- createLibraryRef
- createSharedLibrary
- createScheduler
- createWorkManager

Example: Listing JDBC resources

You can use the listDataSources and listJDBCProviders procedures of the AdminJDBC script to display a list of configuration IDs for the JDBC providers and data sources in your environment.

The syntax to use each procedure is:

- AdminJDBC.listDataSources(*ds_name*)
- AdminJDBC.listJDBCProviders(*jdbcn_name*)

Example 8-18 shows the use of these procedures.

Example 8-18 List of JDBC resources

```
wsadmin>AdminJDBC.listDataSources("PLANTSDB")
```

```
AdminJDBC:          listDataSources
Optional parameter:
DataSource name:    PLANTSDB
Usage: AdminJDBC.listDataSources("PLANTSDB")
```

```

['PLANTSDB(cells/Cel102/nodes/sys1Node01/servers/server1|resources.xml#DataSource_1183122165968)']
wsadmin>
wsadmin>AdminJDBC.listJDBCProviders("Derby JDBC Provider")
-----
AdminJDBC:           listJDBCProviders
Optional parameter:
JDBC provider name: Derby JDBC Provider
Usage: AdminJDBC.listJDBCProvider("Derby JDBC Provider")
-----
["Derby JDBC Provider(cells/Cel102/nodes/sys1Node01/servers/server1|resources.xml#JDBCProvider_1183122153343)"]

```

Example: Creating a J2C connection factory

The `createJ2CConnectionFactory` procedure in the `AdminJ2C` script creates a new J2C connection factory in the environment. The result is the configuration ID of the new J2C connection factory.

The arguments are the resource adapter, connection factory name, the connection factory interface, and the Java Naming and Directory Interface (JNDI) name arguments. This procedure uses the following syntax:

```
AdminJ2C.createJ2CConnectionFactory(resource_adapterID, connfactory_name,  
connFactory_interface, jndi_name, attributes)
```

Example 8-19 shows sample coding for creating a J2C connection factory.

Example 8-19 The `createJ2CConnectionFactory` procedure

```

wsadmin>AdminJ2C.createJ2CActivationSpec("WebSphere MQ Resource
Adapter(cells/Cel102/nodes/DmgrNode02/servers/dmgr|resources.xml#J2CResourceAdapte
r_1234298429000)", "WebSphere MQ Resource Adapter", "javax.jms.MessageListener",
"jdbc/PlantsByWebSphereDataSourceNONJTA")
-----
AdminJ2C:           createJ2CActivationSpec
J2CResourceAdapter configID:   WebSphere MQ Resource
Adapter(cells/Cel102/nodes/DmgrNode02/servers/dmgr|resources.xml#J2CResourceAdapte
r_1234298429000)
J2CActivationSpec name:        WebSphere MQ Resource Adapter
Message listener type:         javax.jms.MessageListener
jndi name:                   jdbc/PlantsByWebSphereDataSourceNONJTA
Optional attributes:
    otherAttributesList []
Usage: AdminJ2C.createJ2CActivationSpec("WebSphere MQ Resource
Adapter(cells/Cel102/nodes/DmgrNode02/servers/dmgr|resources.xml#J2CResourceAdapte
r_1234298429000)", "WebSphere MQ Resource Adapter", "javax.jms.MessageListener",
"jdbc/PlantsB
yWebSphereDataSourceNONJTA")
-----
'"WebSphere MQ Resource Adapter(cells/Cel102/nodes/DmgrNode02/servers/dmgr|resou
rces.xml#J2CActivationSpec_1236206121468)"'
wsadmin>
```

8.5.5 Security script library

The security script library provides a script to manage the security. This library is located in the *install_root/scriptlibraries/security/V70* directory.

The AdminAuthorizations script has the following procedures:

- ▶ Configure authorization groups:
 - addResourceToAuthorizationGroup
 - createAuthorizationGroup
 - mapGroupsToAdminRole
 - mapUsersToAdminRole
- ▶ Remove users and groups from the security authorization settings:
 - deleteAuthorizationGroup
 - removeGroupFromAllAdminRoles
 - removeGroupsFromAdminRole
 - removeResourceFromAuthorizationGroup
 - removeUserFromAllAdminRoles
 - removeUsersFromAdminRole
- ▶ Query your security authorization group configuration:
 - listAuthorizationGroups
 - listAuthorizationGroupsForUserID
 - listAuthorizationGroupsForGroupID
 - listAuthorizationGroupsOfResource
 - listUserIDsOfAuthorizationGroup
 - listGroupIDsOfAuthorizationGroup
 - listResourcesOfAuthorizationGroup
 - listResourcesForUserID

Example: Listing the authorization groups

The listAuthorizationGroups procedure displays each authorization group in the security configuration. This script does not require arguments.

```
AdminAuthorizations.listAuthorizationGroups()
```

Example 8-20 shows sample coding for listing the authorization groups.

Example 8-20 Listing authorization groups

```
wsadmin>AdminAuthorizations.listAuthorizationGroups()
```

```
AdminAuthorizations: List authorization groups
Usage: AdminAuthorizations.listAuthorizationGroups()
```

```
['sec_group1']
```

8.5.6 Server script library

The server script library provides a set of scripts and their procedures to manage the server and the cluster component.

This library is located in the `install_root/scriptlibraries/servers/V70` directory. The library contains the following scripts and procedures:

- ▶ AdminServerManagement script:

Start and stop servers:

- startAllServers
- startSingleServer
- stopAllServers
- stopSingleServer

Configure servers:

- createApplicationServer
- createAppServerTemplate
- createGenericServer
- createWebServer
- deleteServer
- deleteServerTemplate

Query the server configuration:

- checkIfServerExists
- checkIfServerTemplateExists
- getJavaHome
- getServerProcessType
- getServerPID
- help
- listJVMProperties
- listServers
- listServerTemplates
- listServerTypes
- queryingMBeans
- showServerInfo
- viewingProductInformation

Manage server settings:

- configureAdminService
- configureApplicationServerClassloader
- configureDynamicCache
- configureEJBContainer
- configureFileTransferService
- configureListenerPortForMessageListenerService
- configureMessageListenerService
- configureStateManageable
- configureCustomProperty
- configureCustomService
- configureEndPointsHost
- configureJavaVirtualMachine
- configureORBService
- configureProcessDefinition
- configureRuntimeTransactionService
- configureThreadPool
- configureTransactionService
- setJVMProperties
- setTraceSpecification
- configureCookieForServer
- configureHTTPTransportForWebContainer

- configureSessionManagerForServer
 - configureWebContainer
 - configureJavaProcessLogs
 - configurePerformanceMonitoringService
 - configurePMIRequestMetrics
 - configureRASLoggingService
 - configureServerLogs
 - configureTraceService
- AdminClusterManagement script:
- Start cluster processes:
- rippleStartAllClusters
 - rippleStartSingleCluster
 - startAllClusters
 - startSingleCluster
- Stop cluster processes:
- immediateStopAllRunningClusters
 - immediateStopSingleCluster
 - stopAllClusters
 - stopSingleCluster
- Configure clusters:
- createClusterMember
 - createClusterWithFirstMember
 - createClusterWithoutMember
 - createFirstClusterMemberWithTemplate
 - createFirstClusterMemberWithTemplateNodeServer
- Remove clusters and cluster members:
- deleteCluster
 - deleteClusterMember
- Query a cluster configuration:
- checkIfClusterExists
 - checkIfClusterMemberExists
 - help
 - listClusters
 - listClusterMembers

Example: Creating an application server

The CreateApplicationServer procedure of the AdminServerManagement script creates a new application server. The script requires the node to be running. This procedure uses the following syntax:

```
AdminServerManagement.createApplicationServer(node_name, server_name,Template)
```

Example 8-21 illustrates sample coding for creating an application server.

Example 8-21 Creating an application server

```
wsadmin>AdminServerManagement.createApplicationServer("sys1Node01", "Amember01", "default")
-----
AdminServerManagement: Create an application server on a given node
Node name:          sys1Node01
New Server name:    Amember01
Optional parameter:
```

```
Template name: default
Usage: AdminServerManagement.createApplicationServer("sys1Node01", "Amember01", "default")
-----
'Amember01(cells/Ce1102/nodes/sys1Node01/servers/Amember01|server.xml#Server_1235061945890)'
```

Example: Starting an application server

The StartAllServers procedure of the AdminServerManagement script starts all application servers on a node. StartSingleServer starts one server. These procedures use the following syntax:

- ▶ AdminServerManagement.startAllServers(*node_name*)
- ▶ AdminServerManagement.startSingleServer(*node_name*, *server_name*)

Example 8-22 shows sample coding for starting an application server.

Example 8-22 Starting the application server using a single script library

```
wsadmin>AdminServerManagement.startAllServers("sys1Node01")
-----
AdminServerManagement: Start all servers
Node name: sys1Node01
Usage: AdminServerManagement.startAllServers("sys1Node01")
-----
Start server: Amember01
Start server: server1
OK: startAllServers('sys1Node01', 'false'):1

wsadmin>AdminServerManagement.startSingleServer("sys1Node01", "Amember01")
-----
AdminServerManagement: Start single server
Node name: sys1Node01
Server name: Amember01
Usage: AdminServerManagement.startSingleServer("sys1Node01", "Amember01")
-----
Start server: Amember01
OK: startSingleServer('sys1Node01', 'Amember01', 'false'):1
```

Example: Stopping application servers

The StopAllServers procedure stops all application servers on a node. The StopSingleServer will stop one server. These procedures use the following syntax:

- ▶ AdminServerManagement.stopAllServers(*node_name*)
- ▶ AdminServerManagement.stopSingleServer(*node_name*, *server_name*)

Example 8-23 shows sample coding for stopping application servers.

Example 8-23 Stopping the application server using a single script library

```
wsadmin>AdminServerManagement.stopAllServers("sys1Node01")
-----
AdminServerManagement: Stop all servers
Node name: sys1Node01
Usage: AdminServerManagement.stopAllServers("sys1Node01")
-----
Stop server: Amember01
WASX7337I: Invoked stop for server "Amember01" Waiting for stop completion.
```

```

Stop server: server1
WASX7337I: Invoked stop for server "server1" Waiting for stop completion.
OK: stopAllServers('sys1Node01', 'false'):1

wsadmin>AdminServerManagement.stopSingleServer("sys1Node01", "Amember01")
-----
AdminServerManagement: Stop single server
Node name: sys1Node01
Server name: Amember01
Usage: AdminServerManagement.stopSingleServer("sys1Node01", "Amember01")
-----
Stop server: Amember01
WASX7337I: Invoked stop for server "Amember01" Waiting for stop completion.
OK: stopSingleServer('sys1Node01', 'Amember01', 'false'):1

```

8.5.7 System management script library

The system management script library provides a set of scripts that manage nodes and node groups. This library is located in the *install_root/scriptlibraries/system/V70* directory. It contains the following scripts and procedures:

- ▶ **AdminNodeManagement:**
 - configureDiscoveryProtocolOnNode
 - doesNodeExist
 - isNodeRunning
 - listNodes
 - restartActiveNodes
 - restartNodeAgent
 - stopNode
 - stopNodeAgent
 - syncActiveNodes
 - syncNode
- ▶ **AdminNodeGroupManagement:**
 - addNodeGroupMember
 - checkIfNodeExists
 - checkIfNodeGroupExists
 - createNodeGroup
 - createNodeGroupProperty
 - deleteNodeGroup
 - deleteNodeGroupMember
 - deleteNodeGroupProperty
 - help
 - listNodeGroups
 - listNodeGroupMembers
 - listNodeGroupProperties
 - modifyNodeGroup
 - modifyNodeGroupProperty

Example: Querying node group members

The `listNodeGroupMembers` procedure lists the name of each node that is configured within a specific node group. This procedure uses the following syntax:

```
AdminNodeGroupManagement.listNodeGroupMembers(node_group_name)
```

Example 8-24 shows sample coding for querying node group members.

Example 8-24 Listing node group members using the script library

```
wsadmin>AdminNodeGroupManagement.listNodeGroupMembers ("DefaultNodeGroup")
-----
AdminNodeGroupManagement:      List nodes for a given node group
Optional parameter:
    Node group name:      DefaultNodeGroup
Usage: AdminNodeGroupManagement.listNodeGroupMembers ("DefaultNodeGroup")
-----
['DmgrNode02', 'sys1Node01']
```

Example: Synchronizing a node

The `synActiveNodes` and `syncNode` procedures propagate a configuration change. These commands use the following syntax:

- ▶ `AdminNodeManagement.syncActiveNodes()`
- ▶ `AdminNodeManagement.syncNode(node_name)`

Example 8-25 shows sample coding for synchronizing a node.

Example 8-25 Synchronizing the node using the script library

```
wsadmin>AdminNodeManagement.syncNode ("sys1Node01")
-----
AdminNodeManagement:      syncNode
nodeName:                 sys1Node01
Usage: AdminNodeManagement.syncNode ("sys1Node01")
-----
true
1
```

8.6 Assistance with scripting

When you perform an action in the administrative console, you can use the command assistance feature to show the corresponding scripting commands. This feature allows you to capture and copy scripting commands for use in `wsadmin` scripts. You also have the option to send these as notifications to Rational Application Developer V8.0, where you can use the Jython editor to build scripts.

8.6.1 Enabling command assistance

The command assistance feature can help you view `wsadmin` scripting commands in the Jython language for the last action run in the administrative console.

When you perform an action in the administrative console, you can select the **View administrative scripting command for last action** option in the Help area of the window to display the command equivalent. You can copy and paste this command into a script or command window.

You can also enable additional features as follows:

1. Click **System administration → Console Preferences**.

2. Select the command assistance features that you want to use (see Figure 8-3):

- Enable command assistance notifications:

This option should be used in non-production environments to send notifications containing command assist data. Enablement of the notifications allows integration with Rational Application Developer.

- Log command assistance commands:

This option sends the commands to a log.

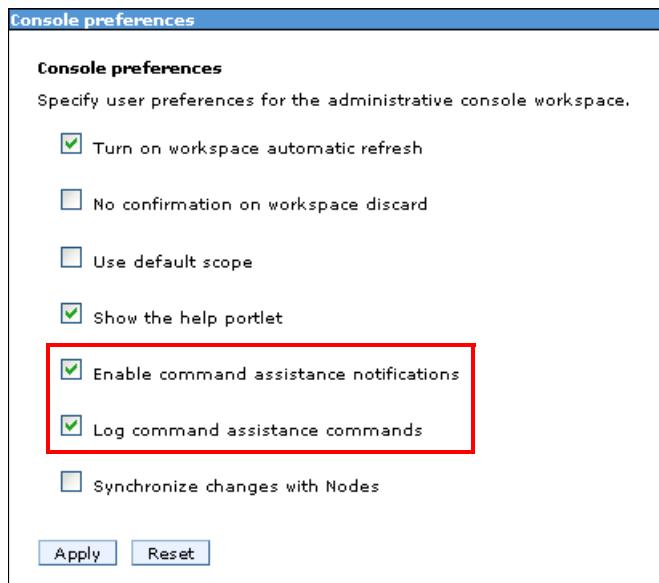


Figure 8-3 Administrative scripting command - Mapping to actions

3. Click **Apply**.

When you select the option to log commands, they are stored in the following location:

profile_root/logs/AssistanceJythonCommands_user_name.log

See Example 8-26 for some sample coding.

Example 8-26 Log location

C:\WebSphere\AppServer\profiles\dmgr01\logs\dmgr
\commandAssistanceJythonCommands_wasadmin.log

The first line of each log entry consists of a time stamp and the location within the console where the command was generated. Below the time stamp is the command information. Example 8-27 shows a sample of the log.

Example 8-27 The command assistance - Log content

```
# [2/24/09 12:15:42:890 EST] Business-level applications > New
AdminTask.createEmptyBLA(['-name sample -description Sample'])

# [2/24/09 12:15:42:906 EST] Business-level applications > New
AdminTask.listBLAs(['-blaID WebSphere:blaname=sample -includeDescription true'])

# [2/24/09 12:15:42:906 EST] Business-level applications > New
```

```

AdminTask.listCompUnits('[-blaID WebSphere:blaname=sample -includeDescription true
                        -includeType true ]')

# [2/24/09 12:15:47:500 EST] Business-level applications > sample
AdminTask.listAssets('[-includeDescription true ]')
# [2/24/09 12:15:47:531 EST] Business-level applications > sample
AdminTask.listBLAs('[-includeDescription true ]')

# [2/24/09 12:15:50:531 EST] Business-level applications > sample > Add
AdminTask.addCompUnit('[-blaID WebSphere:blaname=sample -cuSourceID
                      WebSphere:blaname=IBMUTC ]')

# [2/24/09 12:15:53:562 EST] Business-level applications > sample > Add
AdminTask.addCompUnit('[-blaID WebSphere:blaname=sample -cuSourceID
                      WebSphere:blaname=IBMUTC -CUOptions [[WebSphere:blaname=sample WebSphere:blaname=IBMUTC
                      IBMUTC_0001 "" 1]]]')

# [2/24/09 12:15:57:625 EST] Adding composition unit to the business-level application
AdminConfig.save()

# [2/24/09 12:15:57:890 EST] BLAManagement
AdminTask.listBLAs('[-includeDescription true ]')

# [2/24/09 12:16:01:421 EST] Business-level applications
AdminTask.startBLA('[-blaID WebSphere:blaname=sample ]')

```

8.6.2 Building script files using command assist

The command assist features provide several methods to build scripts. You can copy commands from the Help area of the console or from the log into Jython scripts.

The command assist notifications also provide an integration point with Rational Application Developer, which provides tools that allow you to monitor commands as they are created and to insert the monitored commands into a script.

Working with Jython scripts

To work with Jython scripts in Rational Application Developer, you create a Jython project and Jython script files in the project from any perspective. When you open a new Jython script, it opens with the Jython editor.

You can use the Jython editor in Rational Application Developer V8.0 to perform a variety of tasks, such as:

- ▶ View the administrative console.
- ▶ Develop and edit Jython script files.
- ▶ Import existing Jython files for structured viewing.
- ▶ Set breakpoints for debugging your scripts.

The Jython editor has many text editing features, such as syntax highlighting, unlimited undo or redo, and automatic tab indentation.

When you tag a comment in a Jython script with the "#TODO" tag, the editor automatically creates a corresponding task as a reminder in the Tasks view. Then, if you open the task later, the editor synchronizes automatically to that TODO entry in the script source.

Other helpful features are content assist and tips, which provide a list of acceptable continuations depending on where the cursor is located in a Jython script file or what you just typed. The Jython editor is not integrated to a compiler. As a result, the Jython editor does not perform syntax verification on your scripts.

Using the command assist notifications

The command assistance in the administrative console sends JMX notifications containing command data. You can monitor these notifications from Rational Application Developer. This monitoring requires that you define the server that is producing the notifications as a server in the workspace.

To monitor the commands that are produced as actions, which are taken on the administrative console of the server, complete the following steps:

1. In the Servers view, right-click the server and click **Administration** → **WebSphere administration command assist**. The WebSphere Administration Command view opens.
2. In the Select Server to Monitor list, select the servers that you want the tool to monitor as you interact with its administrative console. The Select Server to Monitor list is available in the toolbar of the WebSphere Administration Command view (see Figure 8-4).

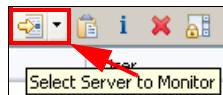


Figure 8-4 Select Server to Monitor icon

You need to start the server that you want to monitor in profile or debug mode. The server is disabled for selection in the Select Server to Monitor list when the server is stopped.

As commands are generated by the console, they display in the WebSphere Application Command Assist view. The commands are shown at WebSphere Administration Command view.

With the Jython script open in the Jython editor and with the monitored command data in the WebSphere Administration Command view, you can insert the commands directly into the script file. Place the cursor in the script file where you want the command to go. Then right-click the command, and select **Insert**, as shown in Figure 8-5. Double-clicking the command also inserts it into the script.

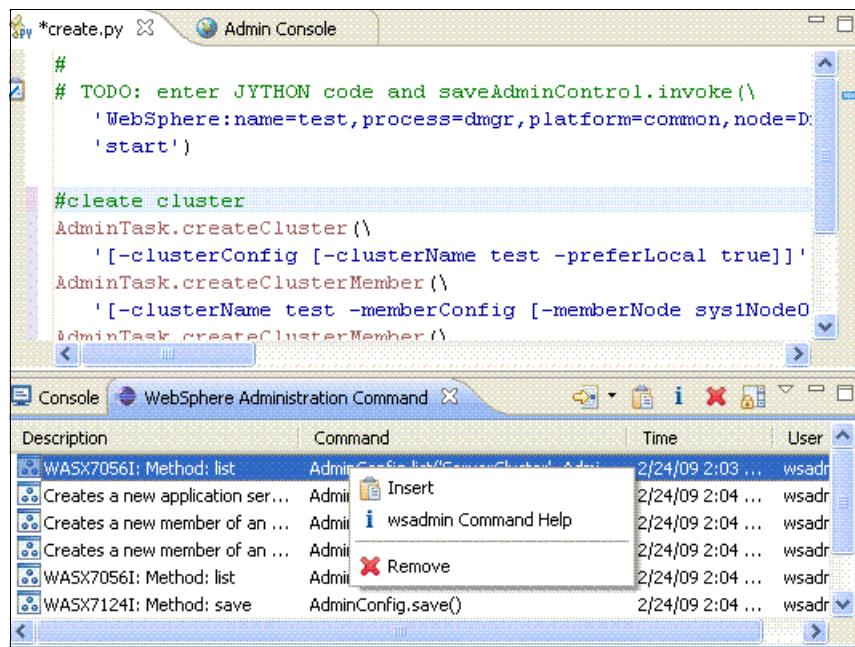


Figure 8-5 Jython editor running the command output automatically

8.7 Example: Using scripts with the job manager

This section provides an example of how to use scripting to automate a WebSphere installation that uses a flexible management environment.

Most companies have routine tasks that occur in different phases of the software development life cycle. In an environment with multiple WebSphere Application Server installations, automation of these tasks can save a lot of time. The combination of **wsadmin** scripts, script libraries, and the automated management provided in a flexible management environment provides an automation solution.

8.7.1 Introduction

The scenario that we describe here uses a simple WebSphere environment to illustrate how to automate tasks. You can use these techniques in more complex environments. This scenario contains the following steps:

1. Write a customized script to automate the tasks.
2. Configure the job manager.
3. Verify the results.

Figure 8-6 shows the scenario environment. A single application server is configured on Node B. The deployment manager for the cell is registered to a job manager on Node A.

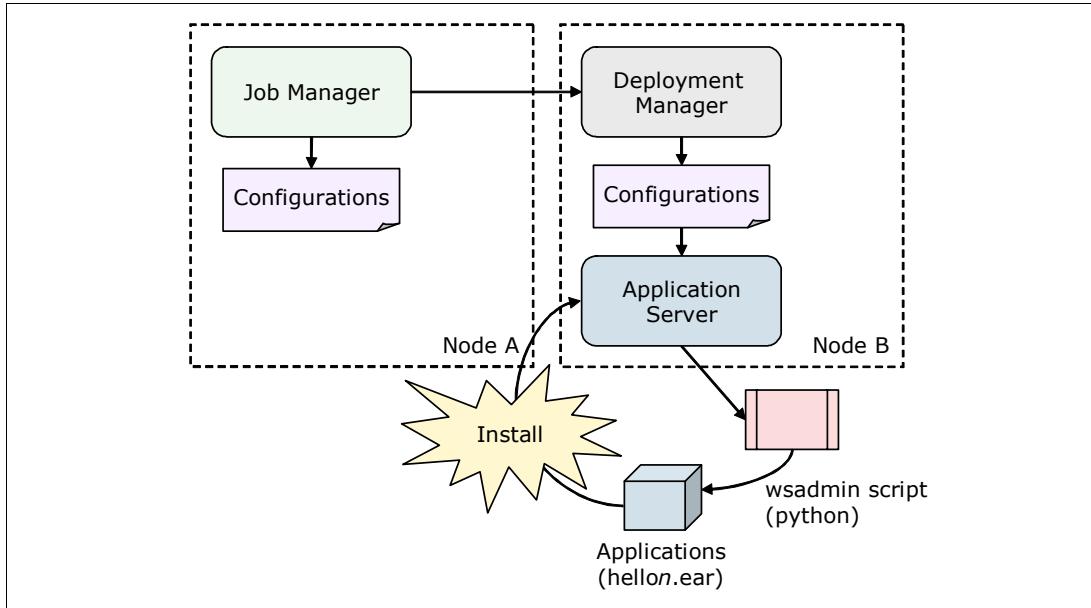


Figure 8-6 The environment details

Applications in this environment are installed frequently, and the administrator needs a quick way to install these applications, which can be accomplished by completing the following steps:

1. A `wsadmin` script is prepared to install an application. The script makes use of the script libraries.
2. A job is scheduled to run the script at regular intervals.
3. The script checks a text file that lists the new applications to be installed. The new application information stored in a text file includes the application name, application location, node name, and server name (see Figure 8-7).

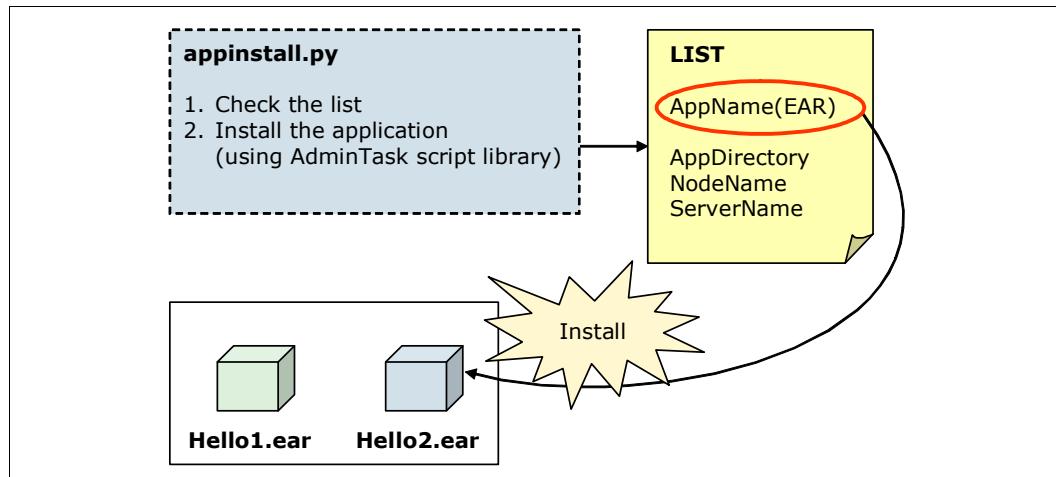


Figure 8-7 Scripting details

4. If the job runs and finds that the application is not installed, the application is installed. If the application is already installed, it is uninstalled and then re-installed.

5. The text file is renamed *filename.txt.old* when the job is executed.
6. If the job executes and no text file exists, no actions are taken. It is only when you distribute a new text file and application that the job will perform the install.

To test, we used the following applications:

- ▶ hello1.ear
- ▶ hello2.ear

8.7.2 Creating the customized script

The **AdminApplication** script in the script libraries includes procedures that will accomplish the installation and update of applications. In this scenario, we used the following procedures from the script libraries:

- ▶ AdminApplication.uninstallApplication

This procedure is used to uninstall an application. The arguments specify the application name.

- ▶ AdminApplication.installAppWithNodeAndServerOptions

This procedure is used to install an application. This procedure is selected because there is only a single server in this environment. In an environment with clustered application servers, use the `installAppWithClusterOption` procedure instead.

The script file is written in Python and is called `appInstall.py` (see Example 8-28).

Example 8-28 The appInstall.py script

```
#Check the list and install/update the applications.
#



import sys
import java
import os
import re
import time
from java.lang import *

dir = "C:/WebSphereV8/AppServer/profiles/dmgr40 downloadedContent/inputfile"
#
sep = System.getProperty("line.separator")
for fname in os.listdir(dir):
    print fname
    path = os.path.join(dir, fname)
    if (os.path.isfile(path)) and (not re.match(".*old$",path) and (not
re.match("appInstall.py",fname))):
        print "processing %s" % (path)
        inp = open(path, 'r')
        for line in inp.readlines():

            itemList = line[:-1].split(',')
            appName = itemList[0]
            earFile = itemList[1]
            nodeName= itemList[2]
            serverName = itemList[3]
```

```

# application existence check
print "application existence check"
existAppList = AdminApp.list().split(sep)
isExist = 0
for existApp in existAppList:
    if(appName == existApp):
        isExist = 1
        break
# acquire application manager
print "acquire application manager"

appMgrID =
AdminControl.queryNames("type=ApplicationManager,node="+nodeName+",process="+serverName+",*" )

# if exists, uninstall application
print "app exists - uninstall"
if( isExist == 1 ):
    appId = ""
    try:
        _excp_ = 0
        appID =
AdminControl.completeObjectName("type=Application,node="+nodeName+,Server="+serverName+,name="
+appName+",*" )
    except:
        _type_, _value_, _tbck_ = sys.exc_info()
        _excp_ = 1
    #endTry
    # if running, stop application
    print "appID is %s" % (appID)
    if(len(appID) > 0):
        print "stopping %s" % (appName)
        stopped = AdminControl.invoke(appMgrID, "stopApplication", appName)
        sleep(1)
    # uninstall application
    print "Uninstalling %s" % (appName)
    AdminApplication.uninstallApplication(appName)

    # install application
    print "Installing %s" % (appName)
    AdminApplication.installAppWithNodeAndServerOptions(appName, earFile, nodeName,
serverName)
    print "Starting %s" % (appName)
    started = AdminControl.invoke(appMgrID, "startApplication", appName)
    sleep(1)

    inp.close()
    os.rename(fname, fname + ".old")
#endif
#endif

```

Example 8-29 shows the hello.txt input file.

Example 8-29 The hello.txt input file

hello,/websphrev8/appserver/profiles/dmgr40/downloadedContent/app1/hello1.ear,node40a,server40a1

The sample script is first tested using `wsadmin` running in script mode. Example 8-30 shows the result of the sample script.

Example 8-30 Testing the sample script

```
C:\WebSphereV8\AppServer\profiles\dmgr40\bin>wsadmin -lang jython -f c:\webspher
e8\appserver\profiles\dmgr40\downloadedContent\appinstall.py -username admin -p
assword admin
WASX7209I: Connected to process "dmgr" on node dmgr40node using SOAP connector;
The type of process is: DeploymentManager
hello.txt
processing C:\WebSphereV8\AppServer\profiles\dmgr40\downloadedContent\inputfile\h
ello.txt
application existence check
acquire application manager
app exists - uninstall
Installing hello
-----
AdminApplication:      Install application with -node and -server options
Application name:     hello
Ear file to deploy:   /websphrev8/appserver/profiles/dmgr40/downloadedConten
t/app1/hello1.ear
Node name:            node40a
Server name:          server40a1
Usage: AdminApplication.installAppWithNodeAndServerOptions("hello", "/websphere
v8/appserver/profiles/dmgr40/downloadedContent/app1/hello1.ear", "node40a", "ser
ver40a1")
-----
ADMA5016I: Installation of hello started.
ADMA5058I: Application and module versions are validated with versions of deployment targets.
ADMA5005I: The application hello is configured in the WebSphere Application Server repository.
ADMA5053I: The library references for the installed optional package are created .
ADMA5005I: The application hello is configured in the WebSphere Application Server repository.
ADMA5001I: The application binaries are saved in C:\WebSphereV8\AppServer\profil
es\dmgr40\wstemp\Script120f8e64a06\workspace\cells\Cell40\applications\hello.ea
r\hello.ear
ADMA5005I: The application hello is configured in the WebSphere Application Server repository.
SECJ0400I: Successfully updated the application hello with the appContextIDForSe
curity information.
ADMA5005I: The application hello is configured in the WebSphere Application Server repository.
CWSAD0040I: The application hello is configured in the Application Server reposi
tory.
ADMA5113I: Activation plan created successfully.
ADMA5011I: The cleanup of the temp directory for application hello is complete.
ADMA5013I: Application hello installed successfully.
OK: installAppWithNodeAndServerOptions('hello', '/websphrev8/appserver/profiles
/dmgr40/downloadedContent/app1/hello1.ear', 'node40a', 'server40a1', 'false'):
```

8.7.3 Submitting the job

To use the job manager to execute the script, complete the following steps:

1. Before running the appInstall.py script, you must transfer it, along with the hello.txt file and the hello1.ear file, from the job manager to the deployment manager using the distributeFile job (see 6.3.2, “Distributing files using the job manager” on page 260 to define the directories that are required for this task).

In Rational Application Developer, export the appInstall.py script file and application EAR file to the *jmgr_profile_root/config/temp/JobManager* directory.

Manually copy the hello.txt file to the same directory.

2. In the Job manager console, click **Job** → **Submit** to launch the Job properties wizard.
3. Use the Distribute file job to transfer each file. In each case, select the admin agent as the target node. The source location refers to the file in the *jmgr_profile_root/config/temp/JobManager* directory. The format of the file is as follows:
file:/file_name

4. Distribute the files as follows:

- Distribute the hello.txt file to the following directory:

dmgr_profile_root/downloadedContent/inputfile

- Distribute the appInstall.py file to the following directory:

dmgr_profile_root/downloadedContent

- Distribute the appInstall.py file to the following directory:

dmgr_profile_root/downloadedContent/app1

5. Submit the **wsadmin** script for execution. Click **Job** → **Submit** to launch the Job properties wizard, and then complete the following steps:

- a. Click **Run wsadmin script** as the job type, and then enter a description. Click **Next** (Figure 8-8).

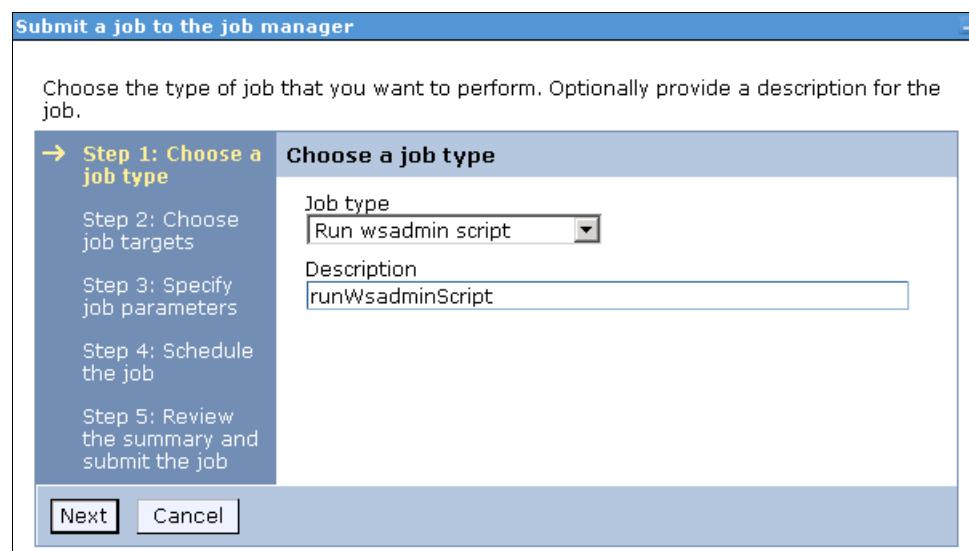


Figure 8-8 Step 1 - Choose the job type

- b. Select the job target. In this case, the deployment manager node is selected. Enter the user ID and password that are required for administration tasks on the deployment manager. Click **Next**.
- c. Specify the script location. This location is the same location that you used when you distributed the file. The current directory is the *dmgr_profile_root/downloadedContent* directory. Click **Next** (Figure 8-9).

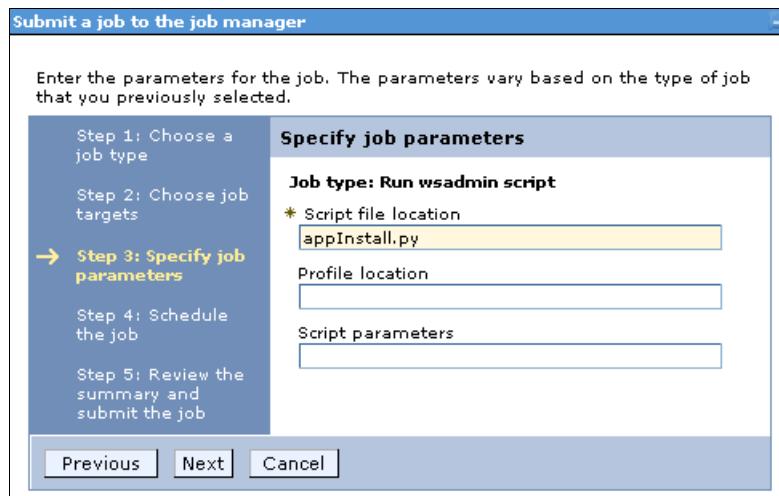


Figure 8-9 Step 3 - Specify job parameters

- d. Next, you can schedule the job to run once or to run multiple times at regular intervals. You can also define when the job is available for execution and when it expires.

In this example, the execution of the script from the job manager is tested first using the “Run once” option in the Availability interval field. After the job is tested, you can repeat the job submit process and set an interval so that the job runs automatically.

Click **Next**.

- e. Review the summary, and then click **Finish**.
- 6. Monitor the job status to ensure that it completes successfully. If the job does not complete successfully, click the Job ID to see the messages produced.

8.7.4 Verifying the results

You can verify the results by displaying the new application in the administrative console, starting the application, and then accessing the application from a web browser, as follows:

1. Access the deployment manager console, expand **Applications** from the navigation tree, and click **Application Types** → **WebSphere enterprise** application.
2. Verify that the new application is in the list.

8.8 Online resources

The following websites are also relevant as further information sources:

- ▶ Command assistance simplifies administrative scripting in WebSphere Application Server:
http://www.ibm.com/developerworks/websphere/library/techarticles/0812_rhodes/0812_rhodes.html
- ▶ Sample Scripts for WebSphere Application Server Versions 5 and 6:
<http://www.ibm.com/developerworks/websphere/library/samples/SampleScripts.html>



Accessing databases from WebSphere

When an application or WebSphere component requires access to a database, that database must be defined to WebSphere as a data source. Two basic definitions are required:

- ▶ A JDBC provider definition defines an existing database provider, including the type of database access that it provides and the location of the database vendor code that provides the implementation.
- ▶ A data source definition defines which JDBC provider to use, the name and location of the database, and other connection properties.

In this chapter, we provide information about the various considerations for accessing databases from WebSphere.

We cover the following topics:

- ▶ JDBC resources
- ▶ Steps in defining access to a database
- ▶ Example: Connecting to an IBM DB2 database
- ▶ Example: Connecting to an Oracle database
- ▶ Example: Connecting to an SQL Server database
- ▶ Example: Connecting to an Informix Dynamic Server database
- ▶ Configuring connection pooling properties

9.1 JDBC resources

The JDBC API provides a programming interface for data access of relational databases from the Java programming language. WebSphere Application Server V8 supports the following JDBC APIs:

- ▶ JDBC 4.0
- ▶ JDBC 3.0
- ▶ JDBC 2.1 and Optional Package API (2.0)

In the following sections, we explain how to create and configure data source objects for use by JDBC applications. This method is the method suggested by best practice to connect to a database and the only method if you intend to use connection pooling and distributed transactions.

The following database platforms are supported for JDBC:

- ▶ DB2
- ▶ Oracle
- ▶ Sybase
- ▶ IBM Informix®
- ▶ SQL Server
- ▶ Apache Derby (test and development only)
- ▶ Third-party vendor JDBC data source using SQL99 standards

9.1.1 JDBC providers and data sources

A *data source* represents a real-world data source, such as a relational database. When a data source object is registered with a JNDI naming service, an application can retrieve it from the naming service and use it to make a connection to the data source that it represents.

Information about the data source and how to locate it, such as its name, the server on which it resides, its port number, and so on, is stored in the form of *properties* on the DataSource object. Storing this information in this manner makes an application more portable because it does not need to hard code a driver name, which often includes the name of a particular vendor. It also makes maintaining the code easier because if, for example, the data source is moved to a different server, all that needs to be done is to update the relevant property in the data source. None of the code using that data source needs to be touched.

After a data source is registered with an application server's JNDI name space, application programmers can use it to make a connection to the data source that it represents.

The connection usually is a *pooled connection*. In other words, when the application closes the connection, the connection is returned to a connection pool, rather than being destroyed.

Data source *classes* and JDBC *drivers* are implemented by the data source vendor. By configuring a JDBC provider, you provide information about the set of classes that are used to implement the data source and the database driver. Also, you provide the environment settings for the DataSource object. A driver can be written purely in the Java programming language or in a mixture of the Java programming language and the Java Native Interface (JNI) native methods.

In the next sections, we describe how to create and configure data source objects, as well as how to configure the connection pools used to serve connections from the data source.

9.1.2 WebSphere support for data sources

The programming model for accessing a data source is as follows:

1. An application retrieves a `DataSource` object from the JNDI naming space.
2. After the `DataSource` object is obtained, the application code calls the `getConnection()` request on the data source to get a `Connection` object. The connection is obtained from a pool of connections.
3. After the connection is acquired, the application sends SQL queries or updates to the database.

In addition to the data source support for Java EE6, Java EE 5, J2EE 1.4, and J2EE 1.3 applications, support is also provided for J2EE 1.2 data sources. The two types of support differ in how connections are handled. However, from an application point of view, they look the same.

Data source support

The primary data source support is intended for J2EE 1.3 and J2EE 1.4, Java EE 5 and Java EE6 applications. Connection pooling is provided by two components, a JCA Connection Manager, and a relational resource adapter. See Figure 9-1.

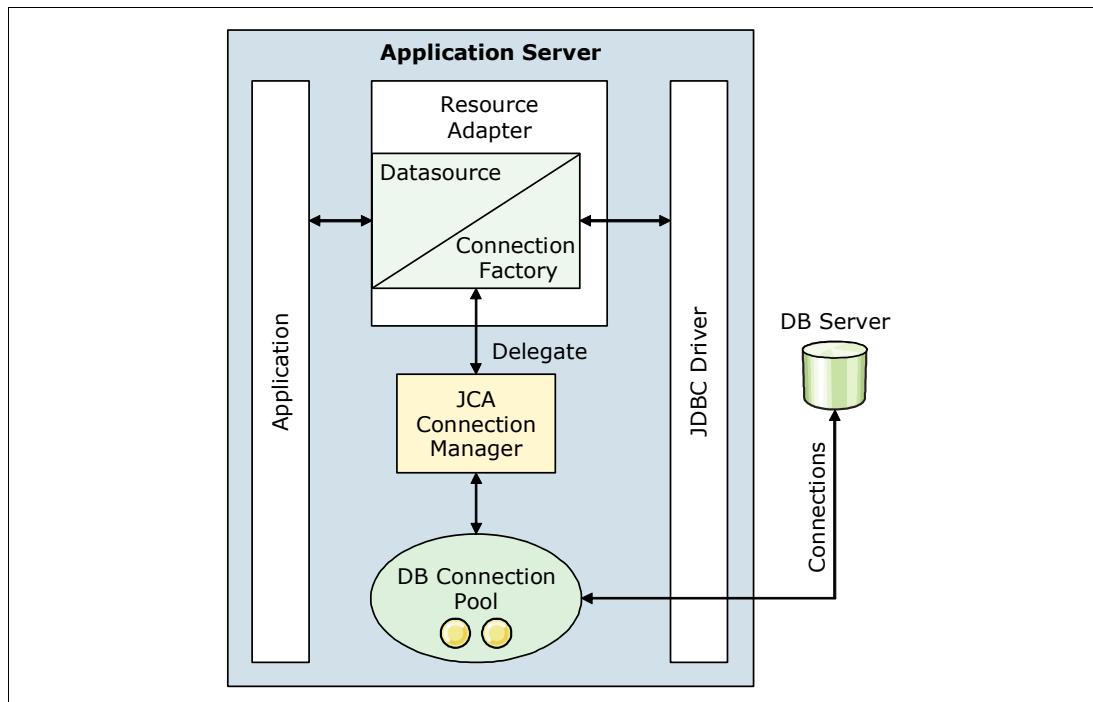


Figure 9-1 Resource adapter in J2EE connector architecture

The JCA Connection Manager provides connection pooling, local transactions, and security support.

The relational resource adapter provides JDBC wrappers and the JCA common client interface (CCI) implementation that allows BMP, JDBC applications, and CMP beans to access the database.

Figure 9-2 shows the relational resource adapter model.

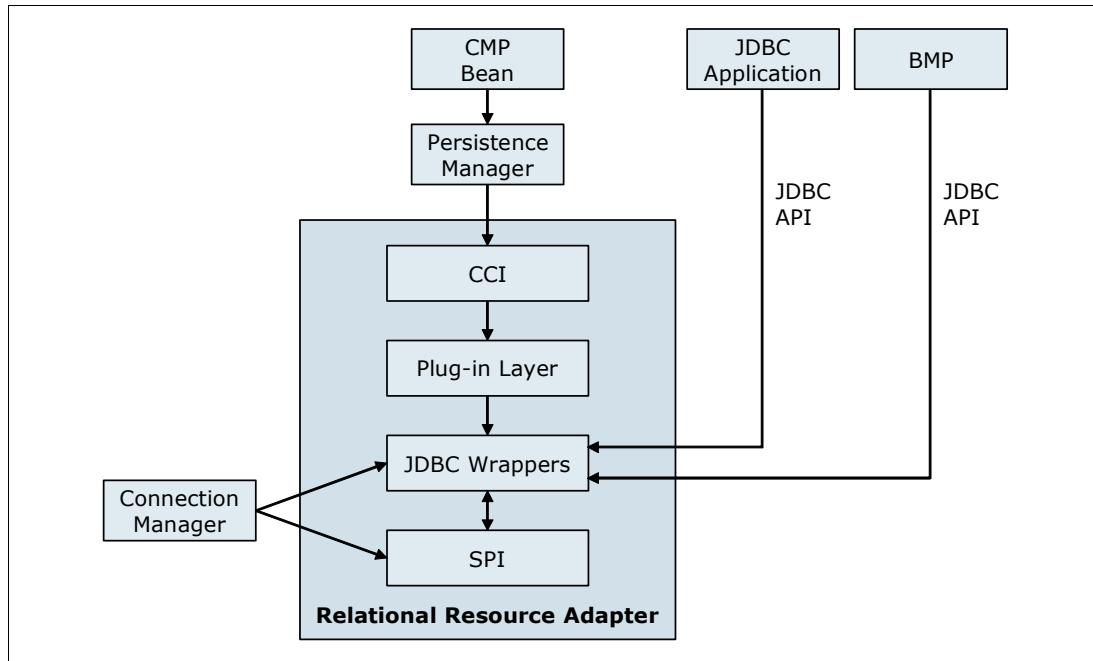


Figure 9-2 Persistence resource adapter model

WebSphere Application Server has a Persistence Resource Adapter that provides relational persistence services to EJB beans as well as providing database access to BMP and JDBC applications. The Persistence Resource Adapter has two components:

- ▶ The Persistence Manager, which supports the EJB CMP persistence model
- ▶ The Relational Resource Adapter

The Persistence Resource Adapter code is included in the following Java packages:

- ▶ The `com.ibm.ws.rsadapter.cci` package contains the CCI implementation and JDBC wrappers.
- ▶ The `com.ibm.ws.rsadapter.spi` package contains the service provider interface (SPI) implementation.
- ▶ The `com.ibm.ws.rsadapter.jdbc` package contains all the JDBC wrappers.
- ▶ The `com.ibm.websphere.rsadapter` package contains `DataStoreHelper`, `WSCallerHelper`, and `DataAccessFunctionSet`.

The Relational Resource Adapter is the Persistence Manager's vehicle to handle data access to and from the back-end store, providing relational persistence services to EJB beans. The implementation is based on the J2EE Connector Architecture (JCA) specification and implements the JCA CCI and SPI interfaces.

When an application uses a data source, the data source uses the JCA connector architecture to get to the relational database.

For an EJB, the sequence is as follows:

1. An EJB performs a JNDI lookup of a data source connection factory and issues a `getConnection()` request.
2. The connection factory delegates the request to a connection manager.

3. The connection manager looks for an instance of a connection pool in the application server. If no connection pool is available, then the manager uses the ManagedConnectionFactory to create a physical, or nonpooled, connection.

Version 4 data source

WebSphere Application Server V4 provided its own JDBC connection manager to handle connection pooling and JDBC access. This support is included with WebSphere Application Server V8 to provide support for J2EE 1.2 applications. If an application chooses to use a Version 4 data source, the WebSphere Application Server V8 application has the same connection behavior as in Version 4 of the application server.

Use the Version 4 data source for the following purposes:

- ▶ J2EE 1.2 applications
 - All EJB beans, JDBC applications, or Version 2.2 servlets must use the Version 4 data source.
- ▶ EJB 1.x modules with 1.1 deployment descriptor
 - All of these modules must use the Version 4 data source.

9.2 Steps in defining access to a database

The following steps are involved in creating a data source:

1. Verify that connection to the database server is supported by WebSphere Application Server. See the following website for more information:
<http://www-01.ibm.com/support/docview.wss?rs=180&uid=swg27006921#8.0>
2. Ensure that the database has been created and can be accessed by the systems that will use it.
3. Ensure that the JDBC provider classes are available on the systems that will access the database. If you are not sure which classes are required, consult the documentation for the provider.
4. Create an authentication alias that contains the user ID and password that will be used to access the database.
5. Create a JDBC provider.

The JDBC provider gives the class path of the data source implementation class and the supporting classes for database connectivity. This is vendor-specific.

The Information Center provides information about JDBC driver support and requirements. To determine if your provider is supported, refer to the JDBC Provider Summary article at the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=/com.ibm.websphere.nd.multiplatform.doc/info/ae/ae/udat_minreq.html

6. Create a data source.

The JDBC data source encapsulates the database-specific connection settings. You can create many data sources that use the same JDBC provider.
7. Save the changes to the master repository and synchronize with the nodes involved.
8. Test the connection to the data source.
9. Review and adjust the connection pool settings (this should be done on a periodic basis).

z/OS note: Always run a test connection with server scope when using the DB2 Universal JDBC Driver Provider Type 2 in WebSphere Application Server Network Deployment for z/OS environment. For example, a node scoped test connection or server scoped test connection (when the particular server is not running) executed in the nodeagent address space. The runtime resource manager does not run in nodeagent. The test will therefore display a failure to load the type 2 native driver library.

9.2.1 Creating an authentication alias

The examples in this chapter assume that the database is password protected and that the user ID and password will be defined at run time.

To create a J2C authentication alias that contains the user ID and password that is required to access the database, complete the following steps:

1. Click **Security** → **Global security**.
2. In the Authentication area, expand **Java Authentication and Authorization Service** and click **J2C authentication data**.
3. Click **New**.
4. Enter an alias name, user ID, and password, as shown in Figure 9-3. The alias name will be used later when you create a resource to identify this as the authentication alias to use. The user ID and password must be valid for the database system and have authority to the database.

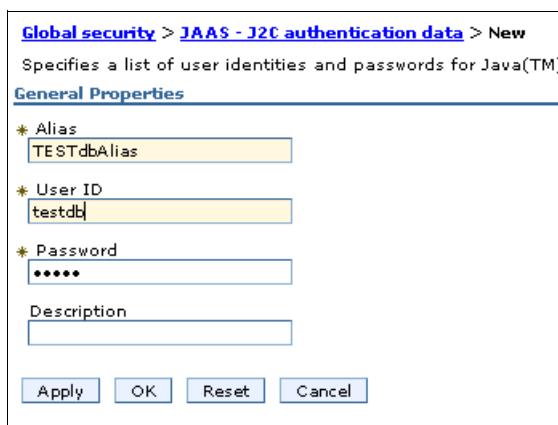


Figure 9-3 Define an authentication alias

5. Click **OK**.

9.3 Example: Connecting to an IBM DB2 database

In this section, we illustrate how to configure a JDBC provider using a DB2 provider as an example.

9.3.1 Creating the JDBC provider

To create a JDBC provider, complete the following steps from the administrative console:

1. Ensure that the implementation classes for the provider are available to the system. The class files will need to be located on each system where the application servers will run.
2. In the administrative console, expand **Resources** → **JDBC** in the navigation tree.
3. Click **JDBC Providers**.
4. Select the scope. (Although you can click **All scopes** to view all resources, you must select a specific scope to create a resource.)

Note: JDBC resources are created at a specific scope level. The data source scope level is inherited from the JDBC provider. For example, if we create a JDBC provider at the node level and then create a data source using that JDBC provider, the data source inherits:

- ▶ The JDBC provider settings, such as class path, implementation class, and so on.
- ▶ The JDBC provider scope level.

In this example, if the scope were set to node-level, all application servers running on that node register the data source in their name space.

The administrative console now shows all the JDBC providers that are created at that scope level.

5. Click **New** to start the wizard and to create a new JDBC provider.
6. In Step 1 of the wizard, define the type of provider you will use. See Figure 9-4.

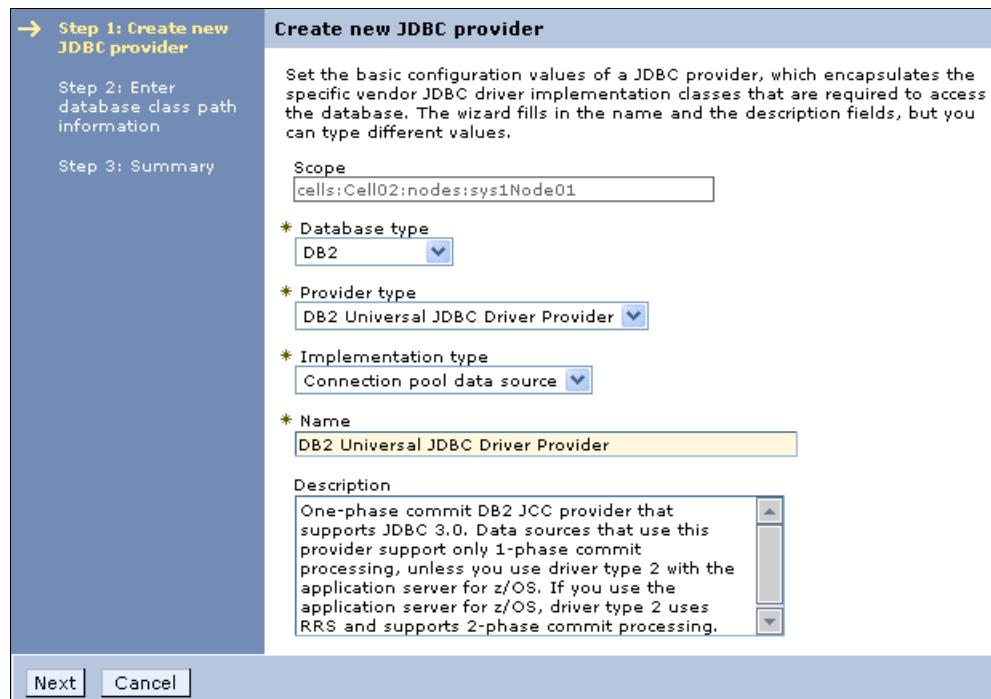


Figure 9-4 Define a new JDBC provider - Window 1

Specify the following information:

- Database type

Select the vendor-specific database type. If the database type you need is not in the list, click **User-defined**, and consult the vendor documentation for the specific properties that are required.

- Provider type

Select from a predefined list of supported provider types, based on the database type that you select.

- Implementation type

Select from the implementation types for the provider type that you selected.

- Name

Specify a name for this driver.

Click **Next**.

7. The settings window for your JDBC database class path opens. Figure 9-5 shows the configuration window for a Universal JDBC Provider.

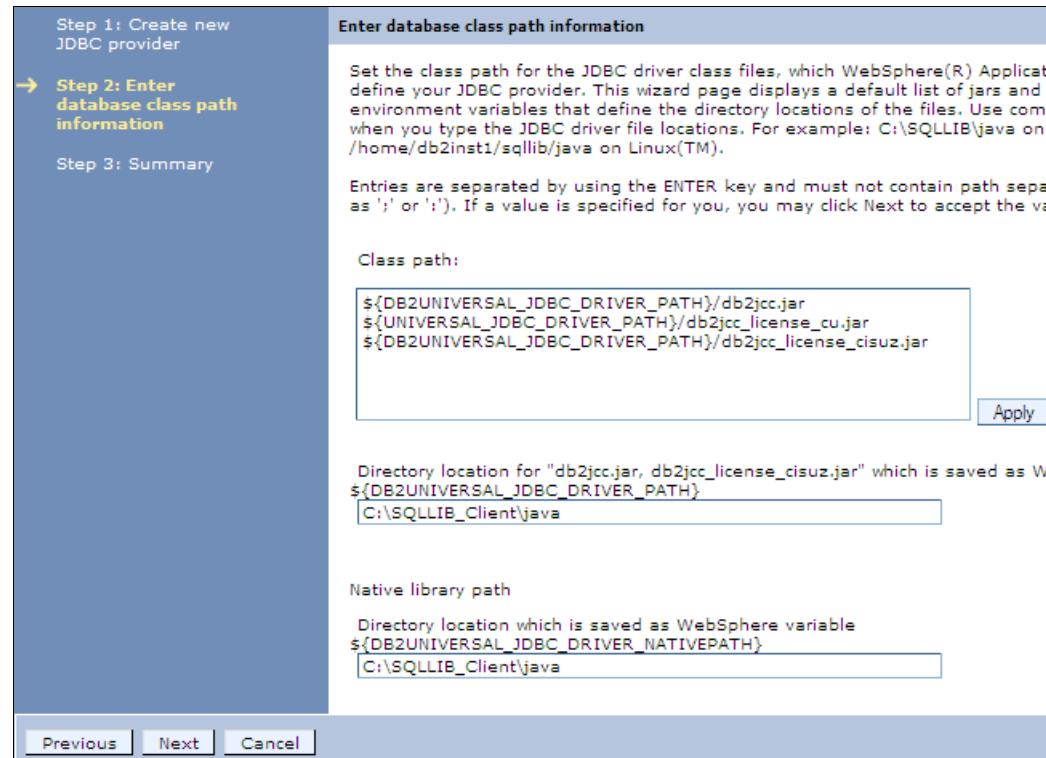


Figure 9-5 Define a new JDBC provider - Window 2

Enter the JDBC provider properties:

- Class path

This field is a list of paths or JAR file names that together form the location for the resource provider classes. This field is pre-set using variable names that are specific to each type of provider. If you are creating a user-defined provider, specify the entries by pressing **Enter** between each entry.

The remaining properties are dependent upon the type of provider. They represent the variables that are used in the class path and their value. If you enter a value for a variable on this window, the corresponding variables are populated automatically with these values. Conversely, if the variables are already defined, these fields are populated with the variables.

You can view or modify the variables by clicking **Environment** → **WebSphere Variables** in the navigation menu.

Because this example is for DB2, the following fields are available:

- Path to db2jcc.jar, db2jcc_license_cisuz.jar

This field specifies the values for the global variable DB2UNIVERSAL_JDBC_DRIVER_PATH, which indicates the class path jar's location.

- Native Library Path

This field is an optional path to any native libraries. Entries are required if the JDBC provider chosen uses non-Java, or native, libraries. The global variable for this is UNIVERSAL_JDBC_DRIVER_NATIVEPATH.

Click **Next**.

8. After verifying the settings, click **Finish** to enable the links to create data sources under the Additional Properties section.

Tip: To make a data source available on multiple nodes using different directory structures, complete the following steps using the administrative console:

1. Define the JDBC provider and data source at the cell scope. Use WebSphere environment variables for the class path and native path.
2. Define the variables at the node scope for each node to specify the driver location for the node.

For example, \${DRIVER_PATH} can be used for the class path in the provider definition. You can then define a variable called \${DRIVER_PATH} at the cell scope to act as a default driver location. Then you can override that variable on any node by defining \${DRIVER_PATH} at the node scope. The node-level definition takes precedence over the cell-level definition.

9.3.2 Creating the data source

Data sources are associated with a specific JDBC provider and can be viewed or created from the JDBC provider configuration window. You have two options when creating a data source, depending on the J2EE support of the application:

- ▶ J2EE 1.2 application: All EJB 1.1 enterprise beans, JDBC applications, or Servlet 2.2 components must use the 4.0 data source.
- ▶ J2EE 1.3 (and subsequent releases) application:
 - EJB 1.1 module: All EJB 1.x beans must use the 4.0 data source.
 - EJB 2.0 (and subsequent releases) module: Enterprise beans that include container-managed persistence (CMP) Version 1.x, 2.0 and beyond must use the new data source.
 - JDBC applications and Servlet 2.3+ components: Must use the new data source.

In this section, we provide information about creating or modifying data sources for Java EE6, Java EE5, J2EE 1.4, and J2EE 1.3 applications. For information about using data sources with J2EE 1.2 applications, see the topic “Data sources (Version 4)” in the Information Center at the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=%2Fcom.ibm.websphere.nd.doc%2Finfo%2Fae%2Fae%2Fudat_was4datson.html

The administrative console provides a wizard that helps you create a data source. Keep in mind, however, that although the wizard provides a good way to establish connections quickly, it also establishes default-sized connection pool settings that you need to tune properly before production.

Complete the following steps to create a data source.

1. Expand **Resources** → **JDBC** in the navigation tree and click **Data sources**.
2. Select the scope. Although you can select **All** to view all resources, you must select a specific scope to create a resource.
The scope determines which applications can use this data source. Select the narrowest scope that is required, while also ensuring that the applications that require the resource can access it. For information about selecting a scope, see “Selecting a scope” on page 237.
3. Click **New** to create a new data source and to start the wizard. See Figure 9-6.

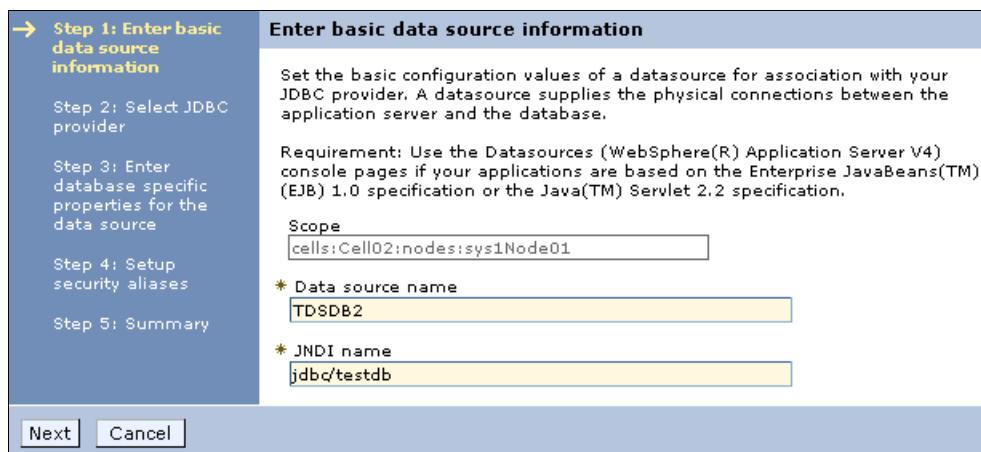


Figure 9-6 Data source general properties

Specify the following information:

– Data source name:

This field is a name by which to administer the data source. Use a name that is suggestive of the database name or function.

– JNDI name:

This field refers to the data source's name as registered in the application server's name space.

When installing an application that contains modules with JDBC resource references, the resources need to be bound to the JNDI name of the resources, for example, `jdbc/<database_name>`.

Click **Next**.

4. Now you need to select an existing JDBC provider or create a new one, as shown in Figure 9-7.

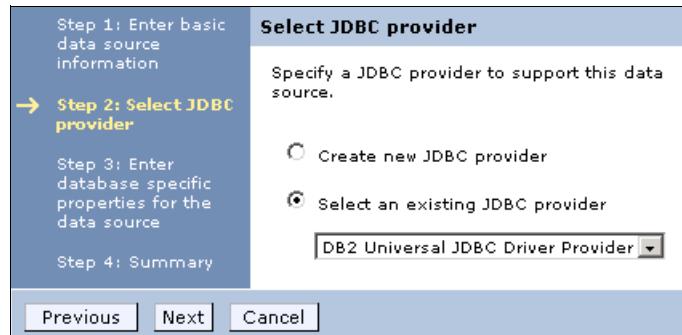


Figure 9-7 Select a JDBC provider

This window allows you to select a JDBC provider or to create a new one. If you create a new JDBC provider, you will be routed through the windows shown in 9.3.1, “Creating the JDBC provider” on page 385. If you select an existing JDBC provider, continue with the next step.

In this case, select an existing JDBC provider and click **Next**.

The entries shown in Figure 9-8 are specific to the JDBC driver and data source type, which show the properties for the Universal data source.

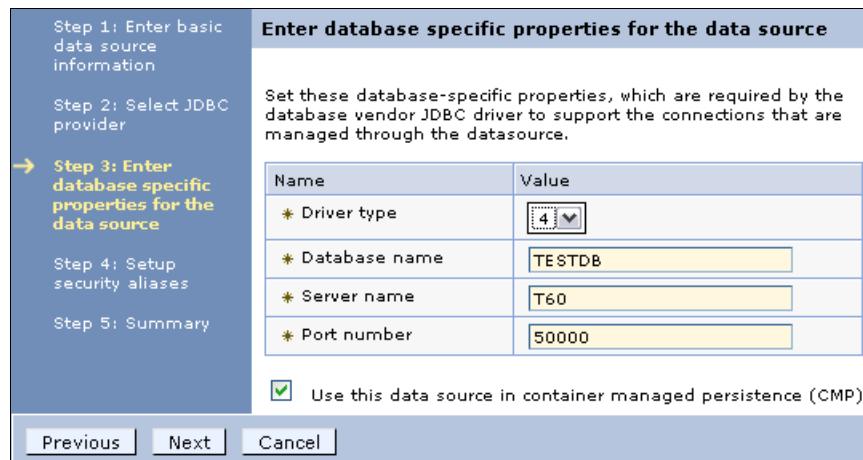


Figure 9-8 Database-specific properties

Specify the following information:

- **Driver type:**

The type of JDBC Driver (2 or 4) used to access the database. To determine the best type of driver to use for your circumstances, consult the documentation for the specific driver that you use.

In general, use type 2 for databases on the same system as the application server and type 4 for remote databases.

- **Database Name:**

The name of the database (or the catalogued alias).

- Server name and port:
The database server name and its listening port (the default for DB2 is 50000).
- Container managed persistence (CMP):
This field specifies if the data source is to be used for container managed persistence of EJB beans.

Deep-dive: Selecting the **Use this data source in container managed persistence (CMP)** option causes a CMP connection factory that corresponds to this data source to be created for the relational resource adapter. The name of the connector factory that is created is <datasourcename>_CF and the connector factory is registered in JNDI under the entry eis/<jndi_name>_CMP.

To view the properties of the just created connection factory, click **Resources** → **Resource Adapters** → **Resource Adapters**. Select the **Show built-in resources** check box in the preferences. Click **WebSphere Relational Resource Adapter** → **CMP Connection Factories**. Be sure to set the scope so that it is the same scope as that for the data source.

Click **Next**.

5. Select or define a J2C authentication alias for the database. The authentication alias simply contains the user ID and password required to access the database. See Figure 9-9.



Figure 9-9 Specify the authentication alias

Click **Next**.

6. A summary of the options that you chose displays. Click **Next** to create the data source.

The new data source is listed in the table of resources. You can test the new connection by selecting the check box to the left of the data source and clicking **Test Connection**. You can view or modify settings for the new data source by clicking the name in the resources list.

9.4 Example: Connecting to an Oracle database

This example illustrates a connection to an Oracle Express 10g database.

Ensure that the implementation classes for the provider are available to the system. The class files need to be located on each system where the application servers will run.

9.4.1 Creating the JDBC provider

Complete the following steps to create the JDBC provider.

1. In the administrative console, expand **Resources** → **JDBC** in the navigation tree.
2. Click **JDBC Providers**.
3. Select the scope. (Although you can select **All scopes** to view all resources, you must select a specific scope to create a resource.)
4. Click **New** to start the wizard and to create a new JDBC provider.
5. In step 1 of the wizard, define the type of provider that you will use. See Figure 9-10.

The screenshot shows the 'Create new JDBC provider' wizard in progress. The left sidebar lists steps: Step 1: Create new JDBC provider (highlighted), Step 2: Enter database class path information, and Step 3: Summary. The main panel has a title 'Create new JDBC provider' with a descriptive text about setting basic configuration values. It contains several input fields:

- Scope:** A dropdown menu showing 'cells:sys2Cell01:clusters:TradeCluster'.
- * Database type:** A dropdown menu showing 'Oracle'.
- * Provider type:** A dropdown menu showing 'Oracle JDBC Driver'.
- * Implementation type:** A dropdown menu showing 'XA data source'.
- * Name:** An input field containing 'Oracle JDBC Driver (XA)', which is highlighted with a yellow background.
- Description:** A text area containing 'Oracle JDBC Driver (XA)'.

At the bottom are 'Next' and 'Cancel' buttons.

Figure 9-10 Define a new Oracle JDBC provider - Step 1

The database type is Oracle and the provider type is Oracle JDBC driver.

Options of implementation type are XA data source or connection pool data source. XA data source types support two-phase commit transactions.

Click **Next**.

- In the next window (Figure 9-11), enter the directory location for the Oracle JDBC drivers. In this example, the ojdbc6.jar is selected by the wizard.

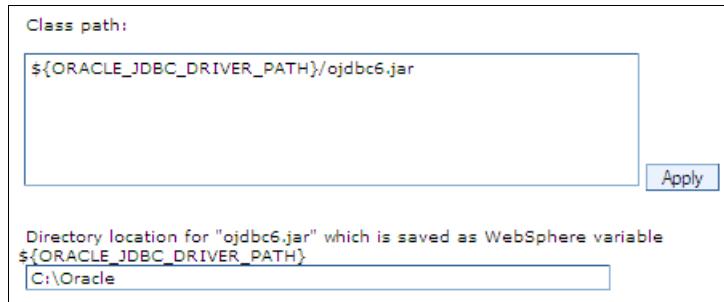


Figure 9-11 Define a new Oracle JDBC provider - Step 2

If you have predefined the ORACLE_JDBC_DRIVER_PATH variable, the driver location is already entered. If you enter a value here, it is saved in the variable.

Click **Next**.

- Review the summary of the settings and click **Finish**. The new JDBC provider displays in the list of providers.

9.4.2 Creating the data source

Complete the following steps to create a data source.

- Expand **Resources** → **JDBC** in the navigation tree and click **Data sources**.
- Select the scope. Although you can select **All scopes** to view all resources, you must select a specific scope to create a resource.
- Click **New** to create a new data source and to start a wizard. See Figure 9-12.

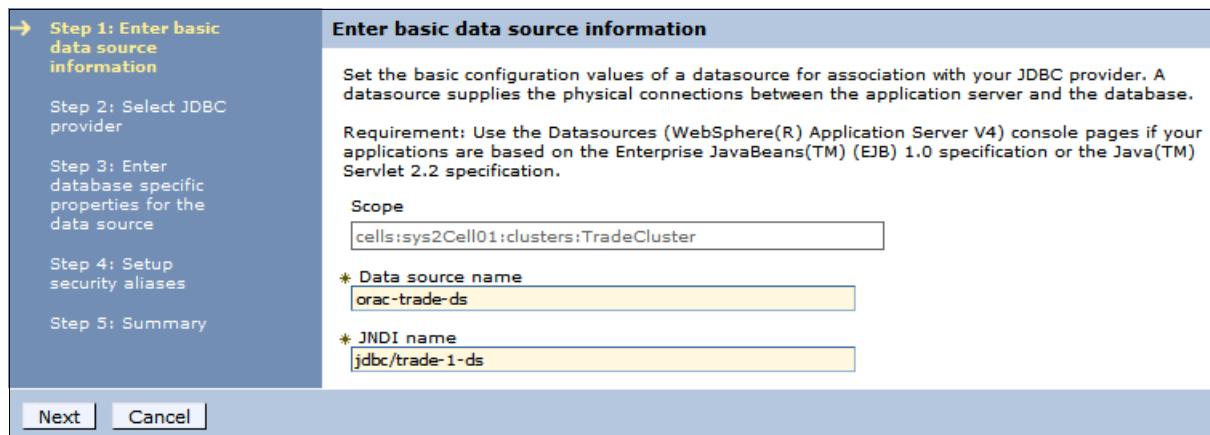


Figure 9-12 Create a data source - Step 1

Enter a name for the new data source. This is used for administrative purposes. Enter the JNDI name that will be used to access the data source and click **Next**.

4. Select the Oracle JDBC driver and click **Next**. See Figure 9-13.

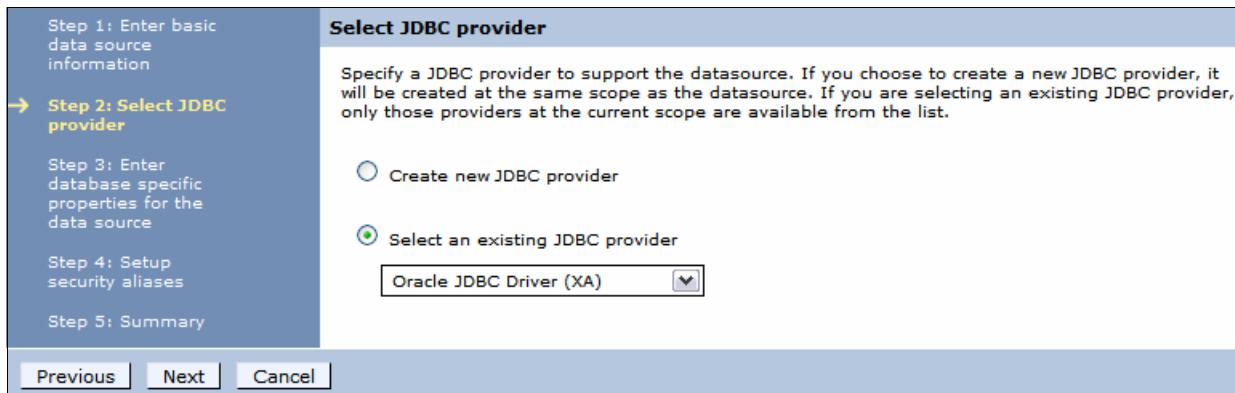


Figure 9-13 Create a data source - Step 2

5. Enter the properties for the database, as shown in Figure 9-14.

Enter database specific properties for the data source	
Set these database-specific properties, which are required by the database vendor JDBC driver to support the connections that are managed through the datasource.	
Name	Value
* URL	<code>jdbc:oracle:thin:@sys2.itso.rl.ibm.com:1521:XE</code>
* Data store helper class name	<code>Oracle11g data store helper</code>
<input checked="" type="checkbox"/> Use this data source in container managed persistence (CMP)	

Figure 9-14 Create a data source - Step 3

Specify the following information:

- The URL for the connection to the XE database is in the following format:
`jdbc:oracle:thin:@host_name:port:service`
In this case:
`jdbc:oracle:thin:@sys2.itso.rl.ibm.com:1521:XE`
 - Select the data store helper class name.
- Click **Next**.

- Select the authentication alias that will provide the user ID and password required to access the database. Click **Next**. See Figure 9-15.



Figure 9-15 Create a data source - Step 4

- Review the summary of your selections and click **Finish**.
- When the data source creation is complete, save the configuration and synchronize the changes with the nodes.
- Test the new connection by selecting the new data source and clicking **Test connection**, as shown in See Figure 9-16.

New	Delete	Test connection	Manage state...			
You can administer the following resources:						
Select	Name	JNDI name	Scope	Provider	Description	Category
<input checked="" type="checkbox"/>	orac-trade-ds	jdbc/trade-1-ds	Cluster=TradeCluster	Oracle JDBC Driver (XA)	New JDBC Datasource	
<input type="checkbox"/>	trade-app-ds	jdbc/trade-ds	Cluster=TradeCluster	DB2 Using IBM JCC Driver (XA)	DB2 JCC XA-capable data source	

Figure 9-16 Test the connection

9.5 Example: Connecting to an SQL Server database

This example illustrates a connection to a Microsoft SQL Server Express 2005 database.

Ensure that the implementation classes for the provider are available to the system. The class files need to be located on each system where the application servers will run.

In general, JDBC drivers are provided by the database vendor. Information about the location and features of the JDBC provider is provided by the database vendor, not the WebSphere documentation.

9.5.1 Creating the JDBC provider

Complete the following steps to create a JDBC provider:

1. In the administrative console, expand **Resources** → **JDBC** from the navigation tree.
2. Click **JDBC Providers**.
3. Select the scope. (Although you can select **All scopes** to view all resources, you must select a specific scope to create a resource.)
4. Click **New** to start the wizard to create a new JDBC provider.
5. In Step 1 of the wizard, define the type of provider that you will use. See Figure 9-17.

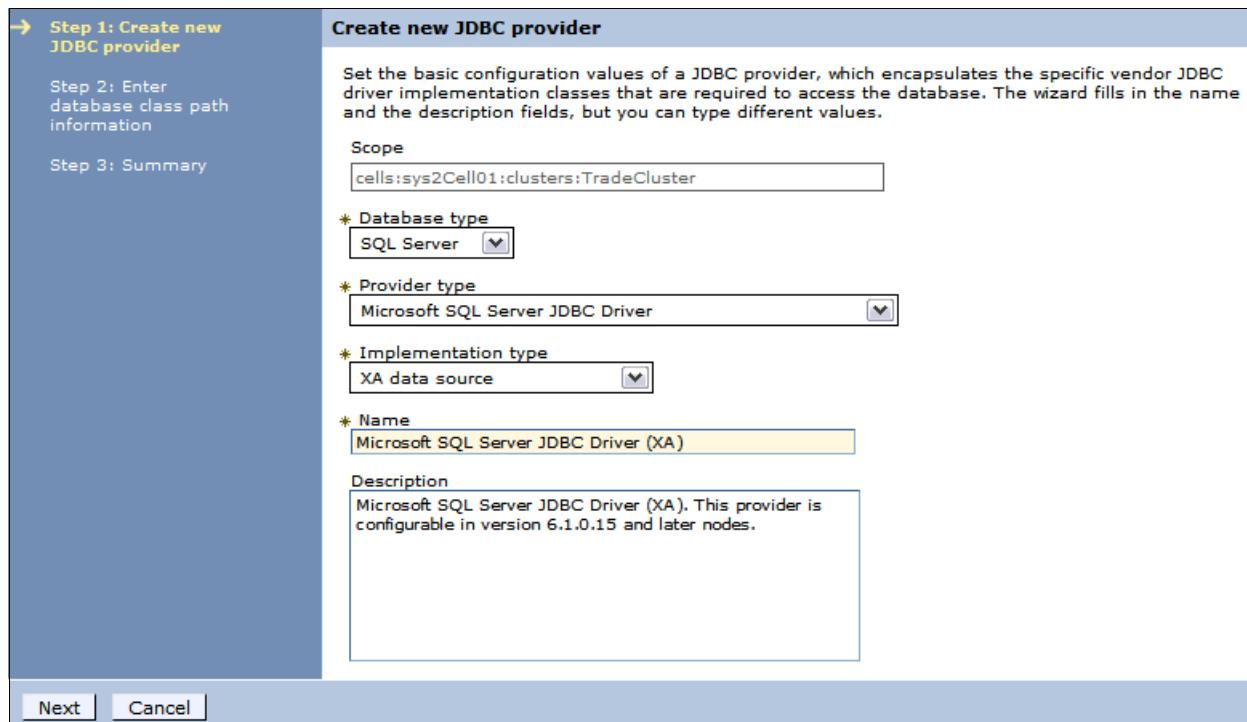


Figure 9-17 Define a new SQL Server JDBC provider - Window1

The database type is SQL Server and provider type is Microsoft SQL Server JDBC driver.

Options of implementation type are XA data source or connection pool data source. XA data source types support two-phase commit transactions.

Click **Next**.

6. Enter the directory location for the SQL Server JDBC drivers. See Figure 9-18.

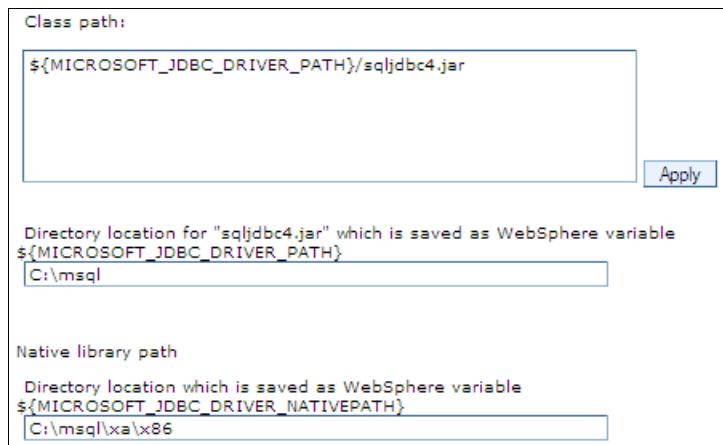


Figure 9-18 Define a new SQL Server JDBC provider - Window 2

If you have predefined the variables used here, the driver locations are already entered. If you enter a value here, it is saved in the appropriate variable.

Click **Next**.

7. Review the summary of the settings and click **Finish**. The new JDBC provider displays in the list of providers.

- Click the JDBC provider name to open the configuration window (Figure 9-19). Verify that the correct driver is used in the class path as advised by the vendor.

JDBC providers > Microsoft SQL Server JDBC Driver (XA)

Use this page to edit properties of a Java Database Connectivity (JDBC) provider. The JDBC provider object encapsulates the specific JDBC driver implementation class for access to the specific vendor database of your environment.

Configuration																			
<table border="1"> <tr> <td colspan="2">General Properties</td> </tr> <tr> <td>* Scope</td> <td>cells:sys2Cell01:clusters:TradeCluster</td> </tr> <tr> <td>* Name</td> <td>Microsoft SQL Server JDBC Driver (XA)</td> </tr> <tr> <td>Description</td> <td>Microsoft SQL Server JDBC Driver (XA). This provider is configurable in version 6.1.0.15 and later nodes.</td> </tr> <tr> <td>Class path</td> <td>`\${MICROSOFT_JDBC_DRIVER_PATH}/sqljdbc4.jar</td> </tr> <tr> <td>Native library path</td> <td>`\${MICROSOFT_JDBC_DRIVER_NATIVEPATH}</td> </tr> <tr> <td colspan="2"> <input type="checkbox"/> Isolate this resource provider </td> </tr> <tr> <td>* Implementation class name</td> <td>com.microsoft.sqlserver.jdbc.SQLServerXADataSource</td> </tr> <tr> <td colspan="2"> <input type="button" value="Apply"/> <input type="button" value="OK"/> <input type="button" value="Reset"/> <input type="button" value="Cancel"/> </td> </tr> </table>		General Properties		* Scope	cells:sys2Cell01:clusters:TradeCluster	* Name	Microsoft SQL Server JDBC Driver (XA)	Description	Microsoft SQL Server JDBC Driver (XA). This provider is configurable in version 6.1.0.15 and later nodes.	Class path	`\${MICROSOFT_JDBC_DRIVER_PATH}/sqljdbc4.jar	Native library path	`\${MICROSOFT_JDBC_DRIVER_NATIVEPATH}	<input type="checkbox"/> Isolate this resource provider		* Implementation class name	com.microsoft.sqlserver.jdbc.SQLServerXADataSource	<input type="button" value="Apply"/> <input type="button" value="OK"/> <input type="button" value="Reset"/> <input type="button" value="Cancel"/>	
General Properties																			
* Scope	cells:sys2Cell01:clusters:TradeCluster																		
* Name	Microsoft SQL Server JDBC Driver (XA)																		
Description	Microsoft SQL Server JDBC Driver (XA). This provider is configurable in version 6.1.0.15 and later nodes.																		
Class path	`\${MICROSOFT_JDBC_DRIVER_PATH}/sqljdbc4.jar																		
Native library path	`\${MICROSOFT_JDBC_DRIVER_NATIVEPATH}																		
<input type="checkbox"/> Isolate this resource provider																			
* Implementation class name	com.microsoft.sqlserver.jdbc.SQLServerXADataSource																		
<input type="button" value="Apply"/> <input type="button" value="OK"/> <input type="button" value="Reset"/> <input type="button" value="Cancel"/>																			
<table border="1"> <tr> <td colspan="2">Additional Properties</td> </tr> <tr> <td colspan="2"> <ul style="list-style-type: none"> ■ Data sources ■ Data sources (WebSphere Application Server V4) </td> </tr> </table>		Additional Properties		<ul style="list-style-type: none"> ■ Data sources ■ Data sources (WebSphere Application Server V4) 															
Additional Properties																			
<ul style="list-style-type: none"> ■ Data sources ■ Data sources (WebSphere Application Server V4) 																			

Figure 9-19 Configure the class path

Click **OK** to create the JDBC provider.

9.5.2 Creating the data source

To create a data source, complete the following steps:

1. Expand **Resources** → **JDBC** in the navigation tree and click **Data sources**.
2. Select the scope. Although you can select **All scopes** to view all resources, you must select a specific scope to create a resource.

- Click **New** to create a new data source and to start a wizard. See Figure 9-20.

Step 1: Enter basic data source information

Set the basic configuration values of a datasource for association with your JDBC provider. A datasource supplies the physical connections between the application server and the database.

Requirement: Use the Datasources (WebSphere(R) Application Server V4) console pages if your applications are based on the Enterprise JavaBeans(TM) (EJB) 1.0 specification or the Java(TM) Servlet 2.2 specification.

Scope
cells:sys2Cell01:clusters:TradeCluster

*** Data source name**
ms-trade

*** JNDI name**
jdbc/trade-2-ds

Next | **Cancel**

Figure 9-20 Create a data source - Step 1

Enter a name for the new data source. This name is used for administrative purposes. Enter the JNDI name that will be used to access the data source and click **Next**.

- Select the Microsoft SQL Server JDBC driver and click **Next**. See Figure 9-21.

Step 1: Enter basic data source information

Step 2: Select JDBC provider

Specify a JDBC provider to support the datasource. If you choose to create a new JDBC provider, it will be created at the same scope as the datasource. If you are selecting an existing JDBC provider, only those providers at the current scope are available from the list.

Create new JDBC provider

Select an existing JDBC provider

Microsoft SQL Server JDBC Driver (XA)

Previous | **Next** | **Cancel**

Figure 9-21 Create a data source - Step 2

- Enter the properties for the database. See Figure 9-22.

Step 1: Enter basic data source information

Step 2: Select JDBC provider

Step 3: Enter database specific properties for the data source

Set these database-specific properties, which are required by the database vendor JDBC driver to support the connections that are managed through the datasource.

Name	Value
Database name	TRADE
Port number	1433
Server name	sys2.itso.ral.ibm.com

Use this data source in container managed persistence (CMP)

Previous | **Next** | **Cancel**

Figure 9-22 Create a data source - Step 3

Specify the following information:

- Enter the database name.
- Enter the port number on which the database server listens.
- Enter the host name of the SQL Server installation.

Click **Next**.

6. Select the authentication alias that provides the user ID and password that are required to access the database. Click **Next**. See Figure 9-23.



Figure 9-23 Create a data source - Step 4

7. Review the summary of your selections and click **Finish**.
8. When the data source creation is complete, save the configuration, and synchronize the changes with the nodes.
9. Test the new connection by selecting the new data source and clicking **Test connection**.

9.6 Example: Connecting to an Informix Dynamic Server database

This example illustrates a connection to an Informix Dynamic Server (IDS) database using the Informix JDBC driver.

Before starting the configuration, ensure that the implementation classes for the provider are available to the system. The class files need to be located on each system where the application servers will run.

Also make sure that an authentication alias is created for the user ID that will be used to connect to the database. For more information, see 9.2.1, “Creating an authentication alias” on page 384.

9.6.1 Creating the JDBC provider

Complete the following steps to create the JDBC provider:

1. In the administrative console, expand **Resources** → **JDBC** from the navigation tree.
2. Click **JDBC Providers**.
3. Select the scope. (Although you can select **All scopes** to view all resources, you must select a specific scope to create a resource.)
4. Click **New** to start the wizard to create a new JDBC provider.
5. In Step 1 of the wizard, define the type of provider that you will use. See Figure 9-24.

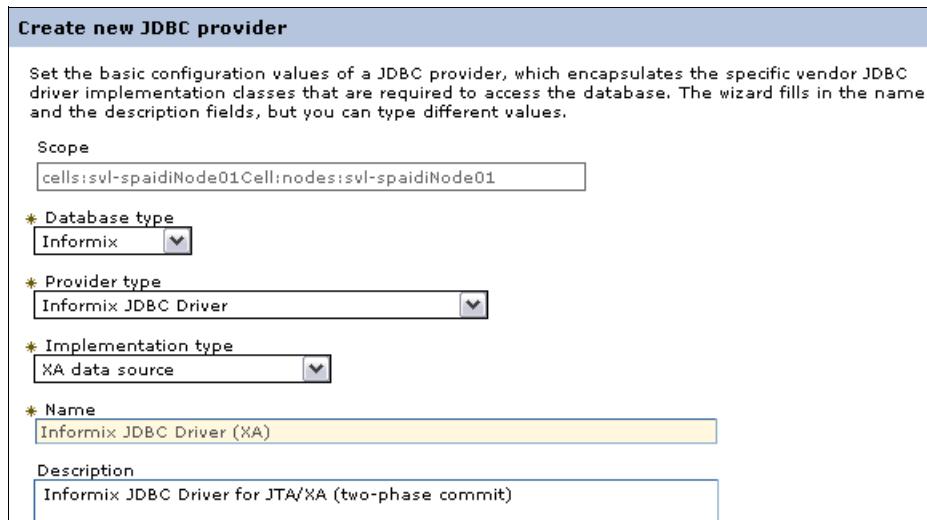


Figure 9-24 Define a new Informix JDBC provider - Step 1

The database type is Informix and provider type is Informix JDBC driver.

Options of implementation type are XA data source or connection pool data source. XA data source types support two-phase commit transactions.

Click **Next**.

6. Enter the directory location for the Informix JDBC drivers. See Figure 9-25.

In this example, ifxjdbc.jar and ifxjdbcx.jar are selected by the wizard.

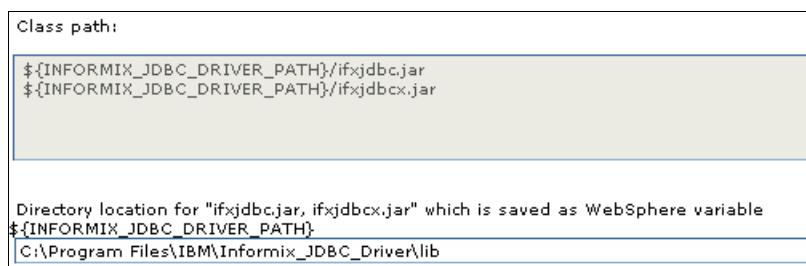


Figure 9-25 Define a new Informix JDBC provider - Step 2

If you have predefined the INFORMIX_JDBC_DRIVER_PATH variable, the driver location is already entered. If you enter a value here, it is saved in the variable.

Click **Next**.

7. Review the summary of the settings and click **Finish**. The new JDBC provider displays in the list of providers.
8. Click the JDBC provider name to open the configuration window (Figure 9-26). If you plan to use SQLJ for queries, change the class path field to add the ifxsqlj.jar file. The implementation class name stays the same.

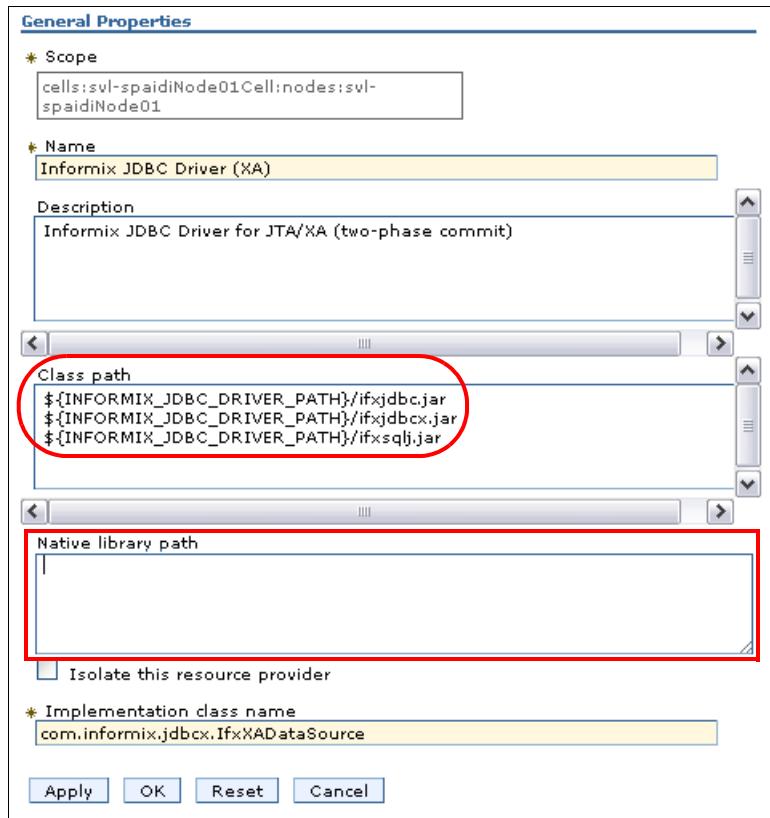


Figure 9-26 Configure the class path

Click **OK** to create the JDBC provider.

9.6.2 Creating the data source

Complete the following steps to create a data source:

1. Expand **Resources** → **JDBC** in the navigation tree and click **Data sources**.
2. Select the scope on the right. Although you can select **All scopes** to view all resources, you must select a specific scope to create a resource.

- Click **New** to create a new data source and to start a wizard. See Figure 9-27.

Enter basic data source information

Set the basic configuration values of a datasource for association with your JDBC provider. This supplies the physical connections between the application server and the database.

Requirement: Use the Datasources (WebSphere(R) Application Server V4) console pages if applications are based on the Enterprise JavaBeans(TM) (EJB) 1.0 specification or the Java 2.2 specification.

Scope
cells:svl-spaidiNode01Cell:nodes:svl-spaidiNode01

* Data source name
stores

* JNDI name
jdbc/stores

Figure 9-27 Create a data source - Step 1

Enter a name for the new data source. This name is used for administrative purposes. Enter the JNDI name that will be used to access the data source and click **Next**.

- In the next window (Figure 9-28), select the Informix JDBC driver and click **Next**.

Select JDBC provider

Specify a JDBC provider to support the datasource. If you choose to create a new provider, it will be created at the same scope as the datasource. If you are selecting an existing provider, those providers at the current scope are available from the list.

Create new JDBC provider
 Select an existing JDBC provider
Informix JDBC Driver (XA)

Figure 9-28 Create a data source - Step 2

- Enter the properties for the database, as shown in Figure 9-29

Enter database specific properties for the data source

Set these database-specific properties, which are required by the database vendor to support the connections that are managed through the datasource.

Name	Value
* Informix lock mode wait	2
* Server name	ol_ids_1150_1
* Database name	stores
Port number	9088
ifxIFXHOST	localhost

Use this data source in container managed persistence (CMP)

Figure 9-29 Create a data source - Step 3

Specify the following information:

- Enter the Informix lock mode wait. The default is 2.
- Enter the server name. This name is the INFORMIXSERVER value, that is, the Informix instance name.

- Enter the database name.
- Enter the port number. This number is the o1soctcp protocol port number. Check your SQLHOSTS file on UNIX, Linux, or the Windows registry on Windows for the correct value to enter.
- Enter the ifxIFXHOST name. This name is host name or the IP address of the host that is running your Informix instance.

Click **Next**.

6. Select the authentication alias that provides the user ID and password required to access the database. In this step, we assume that an authentication alias is created already. Click **Next**. See Figure 9-30.

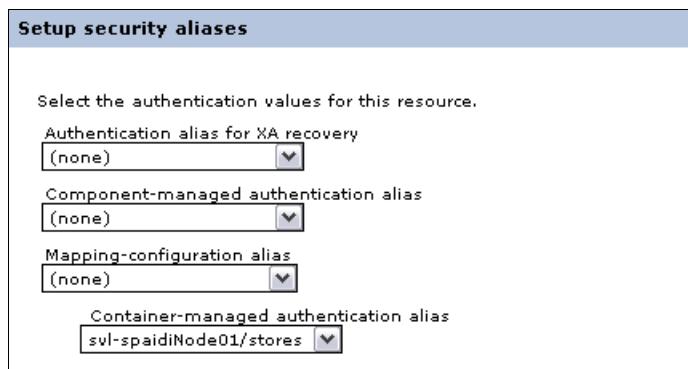


Figure 9-30 Create a data source - Step 4

7. Review the summary of your selections and click **Finish**.
8. When the data source creation is complete, save the configuration, and synchronize the changes with the nodes.
9. Test the new connection by selecting the new data source and clicking **Test connection**. See Figure 9-31.

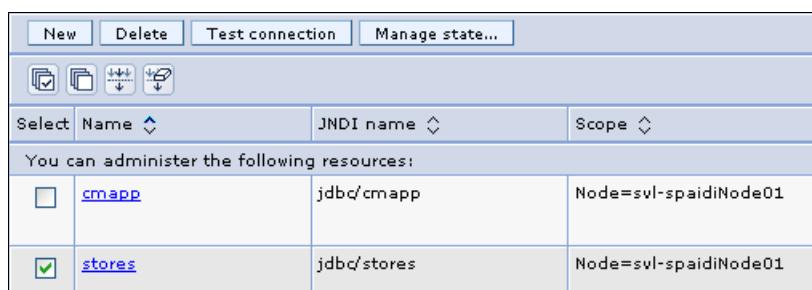


Figure 9-31 Test the connection

9.7 Configuring connection pooling properties

Performance of an application that connects to a database can be greatly affected by the availability of connections to the database and how those connections affect the performance of the database itself. There are no simple rules that tell you how to configure the connection pool properties. Your configuration is highly dependent on application, network, and database characteristics. You need to coordinate the values that you specify in WebSphere closely with the database administrator.

Remember to include all resources in capacity planning. If 10 applications all connect to a database using separate connection pools of 10 maximum connections, this means that there is a theoretical possibility of 100 concurrent connections to the database. Make sure that the database server has sufficient memory and processing capacity to support this requirement.

Complete the following steps to access the connection pool properties:

1. Navigate to **Resources** → **JDBC** → **Data sources** and click the data source name.
2. Click **Connection pool properties** in the Additional Properties section. The window shown in Figure 9-32 opens.

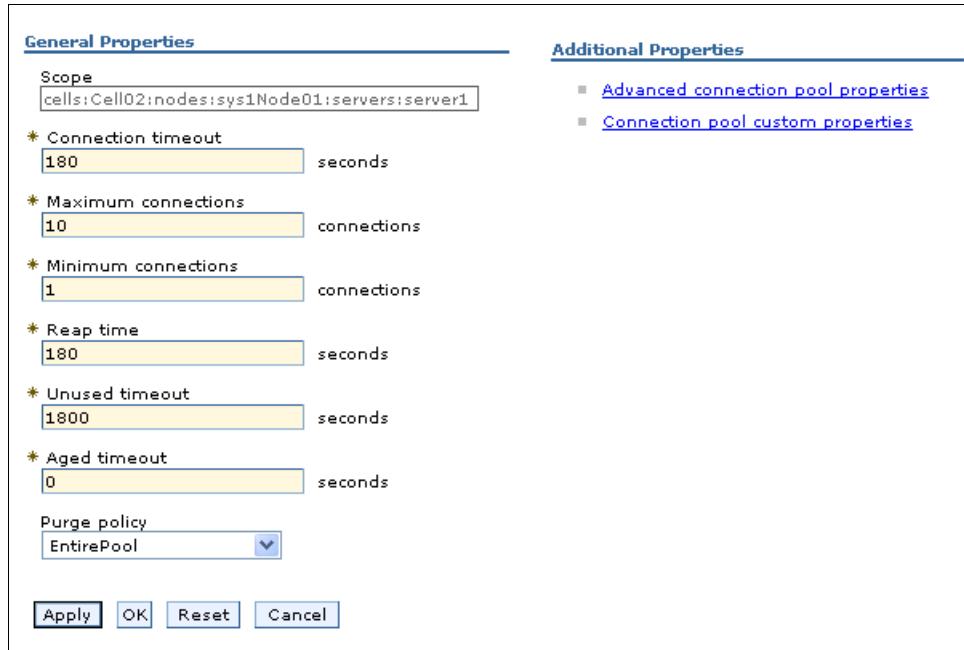


Figure 9-32 Data source connection pool properties

Specify the following information:

- Connection Timeout

Specify the interval, in seconds, after which a connection request times out and a `ConnectionWaitTimeoutException` is thrown. This action can occur when the pool is at its maximum (Max Connections) and all of the connections are in use by other applications for the duration of the wait.

For example, if Connection Timeout is set to 300 and the maximum number of connections is reached, the Pool Manager waits for 300 seconds for an available physical connection. If a physical connection is not available within this time, the Pool Manager throws a `ConnectionWaitTimeoutException`.

Tip: If Connection Timeout is set to 0, the pool manager waits as long as necessary until a connection is allocated.

- Max Connections

Specify the maximum number of physical connections that can be created in this pool.

These connections are the physical connections to the back-end database. After this number is reached, no new physical connections are created and the requester waits until a physical connection that is currently in use is returned to the pool, or a ConnectionWaitTimeoutException is thrown.

For example, if Max Connections is set to 5, and there are five physical connections in use, the Pool Manager waits for the amount of time specified in Connection Timeout for a physical connection to become free. If, after that time, there are still no free connections, the Pool Manager throws a ConnectionWaitTimeoutException to the application.

- Min Connections

Specify the minimum number of physical connections to be maintained. Until this number is reached, the pool maintenance thread does not discard any physical connections. However, no attempt is made to bring the number of connections up to this number.

For example, if Min Connections is set to 3, and one physical connection is created, that connection is not discarded by the Unused Timeout thread. By the same token, the thread does not automatically create two additional physical connections to reach the Min Connections setting.

Tip: Set Min Connections to zero (0) if the following conditions are true:

- ▶ You have a firewall between the application server and database server.
- ▶ Your systems are not busy 24x7.

- Reap Time

Specify the interval, in seconds, between runs of the pool maintenance thread.

For example, if Reap Time is set to 60, the pool maintenance thread runs every 60 seconds. The Reap Time interval affects the accuracy of the Unused Timeout and Aged Timeout settings. The smaller the interval you set, the greater the accuracy. When the pool maintenance thread runs, it discards any connections that have been unused for longer than the time value specified in Unused Timeout, until it reaches the number of connections specified in Min Connections. The pool maintenance thread also discards any connections that remain active longer than the time value specified in Aged Timeout.

Tip: If the pool maintenance thread is enabled, set the Reap Time value less than the values of Unused Timeout and Aged Timeout.

The Reap Time interval also affects performance. Smaller intervals mean that the pool maintenance thread runs more often and degrades performance.

- Unused Timeout

Specify the interval in seconds after which an unused or idle connection is discarded.

Tips:

- ▶ Set the Unused Timeout value higher than the Reap Timeout value for optimal performance. Unused physical connections are only discarded if the current number of connections not in use exceeds the Min Connections setting.
- ▶ Make sure that the database server's timeout for connections exceeds the Unused timeout property specified here. Long lived connections are normal and desirable for performance.

For example, if the unused timeout value is set to 120, and the pool maintenance thread is enabled (Reap Time is not 0), any physical connection that remains unused for two minutes is discarded. Note that accuracy of this timeout, as well as performance, is affected by the Reap Time value. See the Reap Time bullet for more information.

- Aged Timeout

Specify the interval in seconds before a physical connection is discarded, regardless of recent usage activity.

Setting Aged Timeout to 0 allows active physical connections to remain in the pool indefinitely. For example, if the Aged Timeout value is set to 1200, and the Reap Time value is not 0, any physical connection that remains in existence for 1200 seconds (20 minutes) is discarded from the pool. Note that accuracy of this timeout, as well as performance, is affected by the Reap Time value. See Reap Time for more information.

Tip: Set the Aged Timeout value higher than the Reap Timeout value for optimal performance.

- Purge Policy

Specify how to purge connections when a stale connection or fatal connection error is detected.

Valid values are EntirePool and FailingConnectionOnly. If you choose EntirePool, all physical connections in the pool are destroyed when a stale connection is detected. If you choose FailingConnectionOnly, the pool attempts to destroy only the stale connection. The other connections remain in the pool. Final destruction of connections that are in use at the time of the error might be delayed. However, those connections are never returned to the pool.

Tip: Many applications do not handle a StaleConnectionException in the code. Test and ensure that your applications can handle them.

Clicking the **Advanced connection pool properties** link allows you to modify the additional connection pool properties. These properties require advanced knowledge of how connection pooling works and how your system performs. For information about these settings, see the “Connection pool advanced settings” topic in the Information Center at the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=%2Fcom.ibm.websphere.nd.multiplatform.doc%2Finfo%2Fae%2Fae%2Fxml_launchscript.html&resultof=%22wsadmin%22

9.7.1 WebSphere Application Server data source properties

You can set the properties that apply to the WebSphere Application Server connection, rather than to the database connection. To access the connection pool properties, navigate to **Resources → JDBC → Data sources** and click the data source name. Click **WebSphere Application Server data source properties** in the Additional Properties section. See Figure 9-32 on page 404.

Clicking the link opens the window shown in Figure 9-33.

The screenshot shows the 'General Properties' configuration window. It includes sections for 'Statement cache size' (set to 10), 'Error detection model' (set to 'Use WebSphere Application Server exception mapping model'), 'Connection validation properties' (checkboxes for validating new and existing pooled connections, with specific retry interval settings), and 'Validation options' (a query box containing 'SELECT CURRENT SQLID FROM SYSIBM.SYSDUMMY1').

Figure 9-33 WebSphere data source custom properties

Specify the following information:

► Statement Cache Size

Specify the number of prepared statements that are cached per connection. A prepared statement is a precompiled SQL statement that is stored in a prepared statement object. This object is used to execute the given SQL statement multiple times. The WebSphere Application Server data source optimizes the processing of prepared statements.

In general, the more statements your application has, the larger the cache should be. For example, if the application has five SQL statements, set the statement cache size to 5, so that each connection has five statements.

Tip: This setting is vital to performance of the database and will most likely require tuning to suit the specific application. In general, the default is not high enough for best performance.

- ▶ Enable multi-threaded access detection

If you enable this feature, the application server detects the existence of access by multiple threads.

- ▶ Enable database reauthentication

Connection pool searches do not include the user name and password. If you enable this feature, a connection can still be retrieved from the pool, but you must extend the DataStoreHelper class to provide implementation of the `doConnectionSetupPerTransaction()` method where the reauthentication takes place.

Connection reauthentication can help improve performance by reducing the impact of opening and closing connections, particularly for applications that always request connections with different user names and passwords.

- ▶ Enable JMS one-phase optimization support

Activating this support enables the Java Message Service (JMS) to get optimized connections from the data source. Activating this support also prevents JDBC applications from obtaining connections from the data source. For further explanation of JMS one-phase support, refer to the article entitled “Sharing connections to benefit from one-phase commit optimization” at the following website.

<http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp>

- ▶ Log missing transaction context

Specifies whether the container issues an entry to the activity log when an application obtains a connection without a transaction context.

- ▶ Non-transactional data source

Setting the flag to true will cause the Application Server to never enlist the connections from the data source in global or local transactions. Applications must explicitly call `setAutoCommit(false)` on the connection if they want to start a local transaction on the connection, and they must commit or roll back the transaction that they started. This property should rarely be set to true.

- ▶ Error detection model

The error detection model has been expanded and the data source has a configuration option that you can use to select the exception mapping model or the exception checking model for error detection.

- ▶ Connection validation properties

There are two properties, and you can choose both. If you select the **Validate new connections** check box, the application server tries to connect to the database. If you select this property, you can specify how often, in seconds (interval).

If you select the **Validate existing pooled connections** check box, the application server retries to make a connection. If you select this property, you can specify the retry interval for the server reroute. The pretest SQL string is sent to the database to test the connection.

Tip: Connection validation by SQL query is deprecated in WebSphere Application Server V8.0. You can use validation by the JDBC Driver instead. If you use the property of validation by JDBC driver, you need JDBC 4.0 or later. If you do not have JDBC 4.0, you have to update the JDBC driver first.

- ▶ Advanced DB2 features
 - Optimize for get/use/close/connection pattern with heterogeneous pooling

If you check this property, the heterogeneous pooling feature allows you to extend the data source definition. You can specify the retry interval for client reroute, how often to retry, alternate server name or names for the DB2 server, port number, and JNDI name. Details are described in 9.7.2, “Extended DB2 data source” on page 409.
 - DB2 automatic client reroute options

Client reroute for DB2 allows you to provide an alternate server location in case the connection to the database server fails. If you decide to use client reroute with the persistence option, the alternate server information will persist across Java Virtual Machines (JVMs). In the event of an application server crash, the alternate server information will not be lost when the application server is restored and attempts to connect to the database.

9.7.2 Extended DB2 data source

The DB2 Universal JDBC Driver and DB2 using IBM JCC Driver support extends a DB2 data source with what is known as *heterogeneous pooling*. The extended DB2 data source configures a WebSphere DB2 data source with a set of core data source properties. An application can define one or more non-core set of data source properties and associate each with a different resource-reference that points to the main WebSphere DB2 data source.

The benefit of using an extended DB2 data source is that it allows applications to share the same WebSphere connection pool even though each application can have its own set of data source properties.

The sharing leads to a reduction of the number of open connections, and it pushes to reduce resource consumption on both the client side (WebSphere) and the server side (database layer).

For more information, refer to the following article in the Information Center at the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=/com.ibm.websphere.express.doc/info/exp/ae/tdat_heteropool.html

9.7.3 JDBCProviderManagement group commands

Since WebSphere Application Server V6.1, you can use JMX APIs to manage JDBC resources. These APIs were available for use by `wsadmin` scripting or Java programs. For more information about managing JDBC resources through the JMX APIs in a Java program, see the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.base.doc/info/aes/ae/tdat_cjmxapis.html

(New in Version 8) WebSphere Application Server V8 adds two `AdminTask` commands to the JDBCProviderManagement group:

- ▶ `deleteJDBCProvider`
- ▶ `deleteDatasource`

Consider using these commands in your **wsadmin** scripts to either delete a JDBC Provider or delete a data source, respectively. These new commands provide performance enhancements and allow the Deployment Manager to comprehensively administer all JDBC resources in a mixed-cell deployment.

Example of the usage of the new APIs:

```
AdminTask.deleteJDBCProvider('(cells/BB0CELL|resources.xml#JDBCProvider_1309401978500)')
AdminTask.deleteDatasource('(cells/BB0CELL|resources.xml#DataSource_1309399997453')')
```

Note: AdminTask.deleteJDBCProvider deletes all nested data sources and associated CMP connector factories. AdminTask.deleteDatasource deletes only the selected data source.

For general information about the **AdminTask** command, refer to 8.4.5, “AdminTask” on page 351.

For more information about this topic, see the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=%2Fcom.ibm.websphere.base.doc%2Finfo%2Faes%2Fae%2Frxml_atjdbcp provider.html

9.7.4 DB2 Lock sharing between transaction branches

(New in Version 8) WebSphere Application Server V8 allows for a sharing of access to data in a single DB2 database under the same global transaction on the same or on different servers.

Previously, application components had to be collocated on the same server to be able to share locks when accessing data. Sharing locks between transaction branches means that multiple DB2 Java Database Connectivity (JDBC) connections to the same database that are in the same transaction, from the same or different servers, can share locks when accessing data. In this way, multiple components can access the data without causing timeouts or other unwanted situations.

This feature aims to avoid this inconsistency by exploiting a recent capability of DB2 that allows database locks to be shared across multiple transaction branches.

To allow lock sharing, set the **branch-coupling** attribute on each of the DB2 resource references of the application to a value of TIGHT. Use Rational Application Developer V8.0.3 and above to enable this feature:

```
<resource-ref name="jdbc/DataSource_LockSharing" branch-coupling="TIGHT"/>
```

A proprietary extension has been added to the resource reference section of the EJB deployment descriptor and can be found in the **ibm-ejb-jar-ext.xmi** file.

Note: To share locks between transaction branches, the following conditions apply:

- ▶ The database must be DB2 on a distributed or z/OS operating system. The JDBC provider must be DB2 Using IBM JCC Driver Version 3.51 and later, Version 3.6 and later, or Version 4.1 and later.
- ▶ Connections must use JDBC type 4 connectivity to one of the following:
 - IBM DB2 Universal Database™ (DB2 UDB) Version 8 and later
 - DB2 UDB for z/OS Version 8 with program temporary fix (PTF) UK27815 and later
 - DB2 UDB for z/OS Version 9.1 with Fix Pack 4 and later
 - DB2 UDB for z/OS Version 9.5 and later

More information about this feature can be found in the Information Center at the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=%2Fcom.ibm.websphere.express.doc%2Finfo%2Fexp%2Fae%2Fcjta_lockshare.html



Accessing EIS applications from WebSphere

The J2EE Connector architecture (JCA) defines a standard architecture for connecting the J2EE platform to heterogeneous Enterprise Information Systems (EIS), for example, ERP, mainframe transaction processing, database systems, and existing applications not written in the Java programming language. By defining a set of scalable, secure, and transactional mechanisms, the JCA enables the integration of EISs with application servers and enterprise applications. WebSphere Application Server V8 provides a complete implementation of the JCA 1.5 specification, including the features of the JCA 1.0 Specification.

In this chapter, we provide information about the various considerations for accessing EIS applications from WebSphere.

We cover the following topics:

- ▶ JCA resource adapters
- ▶ Resource adapters
- ▶ Configuring J2C connection factories
- ▶ Resource authentication

10.1 JCA resource adapters

The JCA Resource Adapter is a system-level software driver supplied by EIS vendors or other third-party vendors. It provides the following functionality:

- ▶ Provides connectivity between J2EE components, such as an application server or an application client and an EIS.
- ▶ Plugs into an application server.
- ▶ Collaborates with the application server to provide important services, such as connection pooling, transaction, and security services.

JCA defines the following set of system-level contracts between an application server and EIS:

- A *connection management contract* lets an application server pool connect to an underlying EIS, and lets application components connect to an EIS. This contract leads to a scalable application environment that can support a large number of clients requiring access to EISs.
- A *transaction management contract* between the transaction manager and an EIS supports transactional access to EIS resource managers. This contract lets an application server use a transaction manager to manage transactions across multiple resource managers. This contract also supports transactions that are managed internally to an EIS resource manager without the necessity of involving an external transaction manager.
- A *security contract* enables a secure access to an EIS. This contract provides support for a secure application environment, reducing security threats to the EIS and protecting valuable information resources managed by the EIS.

The resource adapter implements the EIS-side of these system-level contracts.

- ▶ Implements the Common Client Interface (CCI) for EIS access.

The CCI defines a standard client API through which a J2EE component accesses the EIS. This simplifies writing code to connect to an EIS data store.

The resource adapter provides connectivity between the EIS, the application server, and the enterprise application through the CCI.

- ▶ Implements the standard Service Provider Interface (SPI).

The SPI integrates the transaction, security, and connection management facilities of an application server (JCA Connection Manager) with those of a transactional resource manager.

Multiple resource adapters (one resource adapter per type of EIS) are pluggable into an application server. This capability enables application components deployed on the application server to access the underlying EISs, as shown in Figure 10-1.

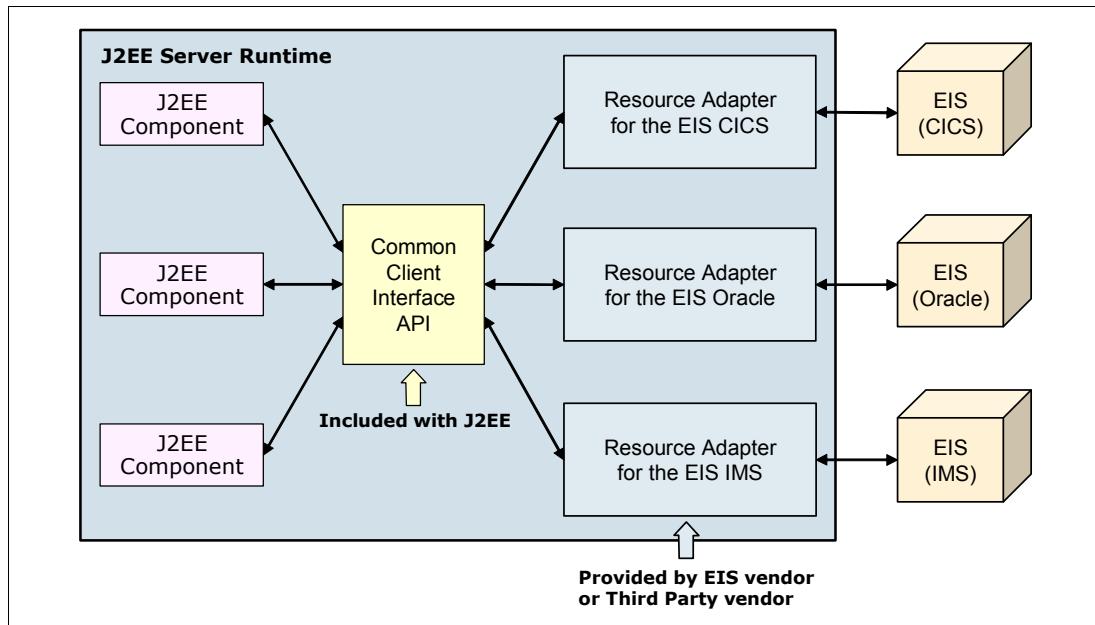


Figure 10-1 Common Client Interface API

10.1.1 WebSphere Application Server JCA support

In WebSphere Application Server, two types of objects are configured for JCA support:

- ▶ Resource adapters
- ▶ Connection factories

The role of the WebSphere administrator is to:

- ▶ Install and define the resource adapter.
- ▶ Define one or more connection factories associated with the resource adapter.

From the application point of view, the application using the resource adapter requests a connection from the connection factory through a JNDI lookup. The connection factory connects the application to the resource adapter.

10.2 Resource adapters

A WebSphere resource adapter administrative object represents the library that supplies implementation code for connecting applications to a specific EIS, such as CICS or SAP. Resource adapters are stored in a Resource Adapter Archive (RAR) file, which is a Java archive (JAR) file used to package a resource adapter for the connector architecture. The file has a standard file extension of .rar.

A RAR file can contain the following elements:

- ▶ EIS-supplied resource adapter implementation code in the form of JAR files or other executables, such as DLLs
- ▶ Utility classes

- ▶ Static documents, such as HTML files for developer documentation, which are not used for run time
- ▶ J2C common client interfaces, such as `cci.jar`
- ▶ A mandatory deployment descriptor (`ra.xml`)

This deployment descriptor instructs the application server about how to use the resource adapter in an application server environment. The deployment descriptor contains information about the resource adapter, including security and transactional capabilities, and the `ManagedConnectionFactory` class name.

The RAR file or JCA resource adapter is provided by your EIS vendor.

Registering the resource adapter with the high availability manager specifies that the high availability (HA) manager will manage the life cycle of a JCA 1.5 resource adapter in a cluster. This manager ensures that applications using resource adapters for inbound communication remain highly available. To that end, appropriate use of the HA capability options enable you to set up an environment that will be able to implement failover for inbound activity when a server goes down.

WebSphere provides two JCA resource adapters:

- ▶ The WebSphere Relational Resource Adapter, used to connect to relational databases using JDBC
- ▶ The SIB JMS Resource Adapter, used to connect to the default messaging provider

10.2.1 Connection factory

The WebSphere connection factory administrative object represents the configuration of a specific connection to the EIS supported by the resource adapter. The connection factory can be thought of as a holder of a list of connection configuration properties.

Application components, such as CMP enterprise beans, have `cmpConnectionFactory` descriptors that refer to a specific connection factory, not to the resource adapter.

10.2.2 Installing and configuring resource adapters

To use a resource adapter, you need to install the resource adapter code and create connection factories that use the adapter. The resource adapter configuration is stored in the `resources.xml` file.

There are two ways to make a resource adapter (.rar file) available to applications. One way is to install the adapter into WebSphere Application Server. The other way is to install the adapter in the application (embedded adapter). For example, Rational Application Developer embeds resource adapters when you create a J2C application. This chapter describes installing the adapter into WebSphere Application Server.

Complete the following steps to install an adapter:

1. From the administrative console, expand **Resources** from the navigation tree, click **Resource Adapters**, and select a scope (Figure 10-2).

Note that you can see all the WebSphere built-in resources by selecting the **Show built-in resources** preference.

The screenshot shows the 'Resource adapters' page with the following details:

- Scope:** Cell=Cell02, Node=sys1Node01 (selected)
- Preferences:**
 - Maximum rows: 20
 - Retain filter criteria (unchecked)
 - Show items at the following authorization group level: All Roles
 - Show built-in resources (unchecked)
- Buttons:** Apply, Reset
- Action Bar:** Install RAR, New, Delete, Update RAR
- Toolbar:** Select, Name, Scope
- Table:** You can administer the following resources:

Select	Name	Scope
<input type="checkbox"/>	WebSphere MQ Resource Adapter	Node=sys1Node01

Total 1

Figure 10-2 JCA resource adapters

2. Click **Install RAR** to install a new resource adapter.

3. Enter the path to the RAR file supplied by your EIS vendor. It can reside locally, on the same machine as the browser, or on any of the nodes in your cell. See Figure 10-3.

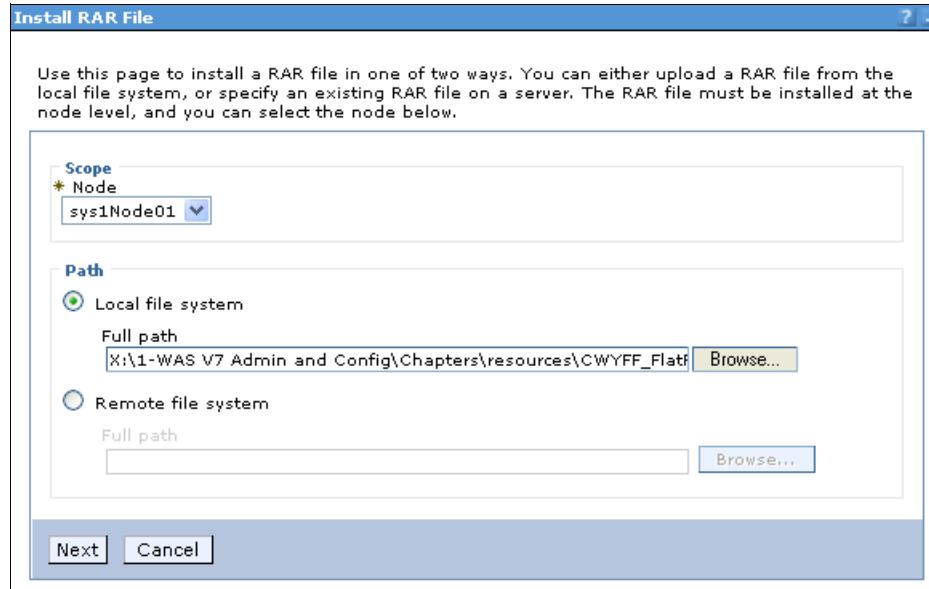


Figure 10-3 RAR file location

Select the node where you want to install the RAR file. You have to install the file on each node separately.

Click **Next**.

- The Configuration window for the resource adapter selected opens, as shown in Figure 10-4.

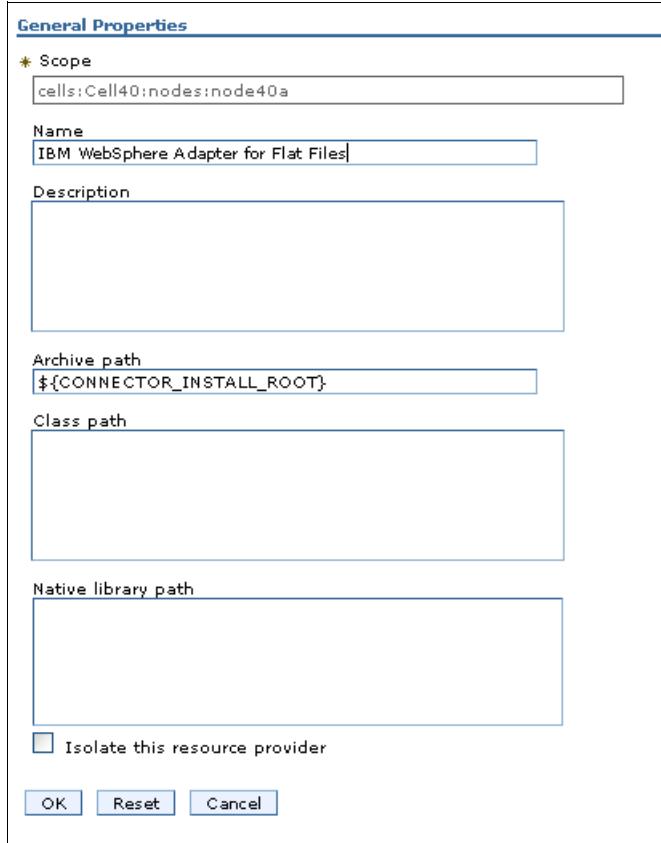


Figure 10-4 JCA resource adapter properties

In this example, you do not have to configure any properties. The defaults combined with the information supplied in the RAR file provide all the information needed. However, you have the option of configuring the following properties:

- Name:**
Create an administrative name for the resource adapter.
- Description:**
Create an optional description of the resource adapter for your administrative records.
- Archive path:**
This field is the path where the RAR file is installed. If this property is not specified, the archive will be extracted to the absolute path represented by the \${CONNECTOR_INSTALL_ROOT} variable. The default is *profile_root/installedConnectors/adapter_name.rar*.
- Class path:**
A list of paths or JAR file names that together form the location for the resource adapter classes. The resource adapter code base itself, the RAR file, is automatically added to the class path.
- Native path:**
This is a list of paths that together form the location for the resource adapter native libraries (.dll and .so files).

5. Click **OK**.
6. Save the configuration and synchronize the nodes.

10.3 Configuring J2C connection factories

Terms: The terms J2C and JCA both refer to J2EE Connector Architecture and they are used here interchangeably.

A J2C connection factory represents a set of connection configuration values. Application components such as EJBs have <resource-ref> descriptors that refer to the connection factory, not the resource adapter. The connection factory is just a holder of a list of connection configuration properties. In addition to the arbitrary set of configuration properties defined by the vendor of the resource adapter, there are several standard configuration properties that apply to the connection factory. These standard properties are used by the connection pool manager in the application server run time and are not used by the vendor supplied resource adapter code.

Complete the following steps to create a J2C connection factory:

1. Click **Resources** → **Resource Adapters** → **J2C connection factories**. You see a list of J2C connection factories at the selected scope.
2. Click **New** to create a new connection factory or select an existing one to modify the connection factory properties.

The J2C Connection Factory Configuration window is shown in Figure 10-5.



Figure 10-5 J2C connection factory properties

The general properties are:

– Name:

Enter an administrative name for the J2C connection factory.

– JNDI name:

This field is the connection factory name to be registered in the application server's name space, including any naming sub context.

When installing an application that contains modules with J2C resource references, the resources defined by the deployment descriptor of the module need to be bound to the JNDI name of the resource.

As a convention, use the value of the Name property prefixed with eis/, for example, eis/<ConnectionFactoryName>

– Description:

This is an optional description of the J2C connection factory, for your administrative records.

- Connection factory interface:
This field is the name of the connection factory interfaces supported by the resource adapter.
 - Category:
Specify a category that you can use to classify or group the connection factory.
 - Security settings:
You have multiple options when securing access to the J2C resource.
While component-managed might be faster in some instances, it is not the best solution for security. Container-managed authentication is the preferred method.
For more information, see 10.4, “Resource authentication” on page 422.
3. Click **Apply**. The links under the Additional Properties section for connection pool, advanced connection factory, and custom properties become active.
- The connection pool properties can affect performance of your application. You should monitor and adjust these settings to maximize performance.
- The advanced connection factory properties are shown in Figure 10-6.

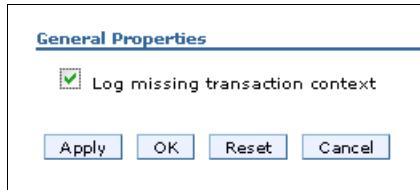


Figure 10-6 Advanced connection factory properties

The J2EE programming model indicates that connections should always have a transaction context. However, some applications do not have a context associated with them. The Log missing transaction context option tells the container to log the fact that there is a missing transaction context in the activity log when the connection is obtained.

10.4 Resource authentication

Resources often require you to perform authentication and authorization before an application can access them. You can configure the settings to determine how this is done in a number of ways. This section provides information about the configuration settings and how to use them. However, before implementing any security, you should review the information in *WebSphere Application Server V7.0 Security Guide*, SG24-7660.

The party responsible for the authentication and authorization is determined by the res-auth setting found in the web and EJB deployment descriptors. There are two possible settings:

- res-auth=Container: WebSphere is responsible.
The authentication data is supplied by the application server.
- res-auth=Application: The application, or component, is responsible.
The authentication data is taken from the following elements, in order:
 - The user ID and password that are passed to the getConnection method. (This is not a best practice for obvious reasons. This implies that the user ID and password are coded in the application).

- The component-managed authentication alias in the connection factory or the data source.
- The custom properties for the user name and password in the data source. The names of the custom properties are specific to the JDBC provider associated with the data source but for many JDBC drivers, user and password are supported. Consult the documentation for the JDBC driver for the correct property names. These properties should be configured by creating two properly named custom properties on the data source and setting the value appropriately.

These settings can be configured during application assembly using Rational Application Developer in the EJB or web deployment descriptor. They can also be set or overridden during application installation. See Table 10-1.

Table 10-1 Authentication settings

Authentication type	Setting at assembly Authorization type	Setting during installation Resource authorization
WebSphere managed: res-auth=Container	Container	Container
Application (component) managed: res-auth=Application	Per_Connection_Factory	Per application

10.4.1 Container-managed authentication

Container-managed authentication removes the requirement that the component programmatically supply the credentials for accessing the resource. Instead of calling the `getConnection()` method with a `ConnectionSpec` object, `getConnection()` is called with no arguments. The authentication credentials are then supplied by the web container, application container, or the EJB container, depending on from where the resource is accessed. WebSphere Application Server supports the Java Authentication and Authorization Service (JAAS) specification, so the credentials can be mapped from any of the configured JAAS authentication login modules, including any custom JAAS authentication login module.

The default selection for the JAAS application login module (in the mapping-configuration-alias field of the J2C connection factory), `DefaultPrincipleMapping`, maps the user ID and password using a pre-configured J2C authentication alias.

Container-managed authentication is the preferred method.

10.4.2 Component-managed authentication

In the case of component-managed authentication, the application component accessing the resource or adapter is responsible for programmatically supplying the credentials. WebSphere can also supply a default component-managed authentication alias if available. After obtaining the connection factory for the resource from JNDI, the application component creates a connection to the resource using the `create` method on the connection factory supplying the credentials. If no credentials are supplied when creating a connection and a component-managed authentication alias has been specified on the J2C connection factory, the credentials from the authentication alias will be used.

Assuming that the credentials are valid, future requests using the same connection will use the same credentials.

The application follows these basic steps:

1. Get the initial JNDI context.
2. Look up the connection factory for the resource adapter.
3. Create a ConnectionSpec object holding credentials.
4. Obtain a connection object from the connection factory by supplying the ConnectionSpec object.



Configuring messaging providers

In this chapter, we introduce how to configure messaging providers step by step.

This chapter contains the following topics:

- ▶ Messaging providers introduction
- ▶ Configuring the default messaging provider
- ▶ Configuring the WebSphere MQ provider
- ▶ Configuring a generic JMS provider

11.1 Messaging providers introduction

WebSphere Application Server V8 supports a variety of JMS providers, including:

- ▶ The WebSphere Application Server default messaging provider, which is a JCA resource adapter implementation that is fully integrated in WebSphere. The default messaging provider uses a service integration bus as the messaging system.
- ▶ The WebSphere MQ messaging provider, which uses a WebSphere MQ installation as the provider.
- ▶ Third-party messaging provider, which implements either a JCA Version 1.5 resource adapter or the Application Server Facilities component of the JMS Version 1.1 specification.

Deprecated feature: The Version 5 default messaging provider is deprecated. For backwards compatibility with earlier releases, WebSphere Application Server continues to support this default messaging provider. Your applications that use these resources can communicate with Version 5 nodes in mixed cells at later versions.

For detailed information about the basic concept of messaging, refer to *WebSphere Application Server V7 Messaging Administration Guide*, SG24-7770.

11.2 Configuring the default messaging provider

To enable a messaging application to use the default messaging provider, configure the following resources:

- ▶ A connection factory
- ▶ A JMS destination
- ▶ A JMS activation specification

The sections that follow introduce how to configure these resources step by step.

11.2.1 Configuring a connection factory

To configure a JMS connection factory for the default messaging provider, complete the following steps.

1. If you have not created a service integration bus, create it now. (Refer to *WebSphere Application Server V7 Messaging Administration Guide*, SG24-7770 for details.) In this example, we use a bus called *SampleBus*.

2. Click **Resources** → **JMS** → **Connection factories**. In the Connection factories window, shown in Figure 11-1, choose the scope (the **Cell scope** in this sample), and then click **New**.

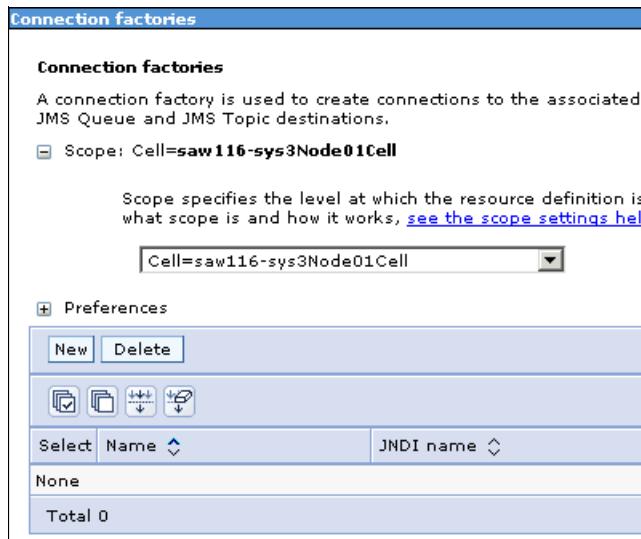


Figure 11-1 Create a new connection factory

3. Select the **Default messaging provider** option and input the following basic properties:
- Name
 - JNDI name
 - Bus names

In this example, shown in Figure 11-2, SampleJMSCF connects to SampleBus. The JMS application accesses the factory using the JNDI name jms/SampleJMSCF.

Connection factories > Default messaging provider > New...

A JMS connection factory is used to create connections to the associated publish/subscribe messaging. Use connection factory administrative provider.

Configuration

General Properties

Administration

- Scope: Cell=saw116-sys3Node01Cell
- Provider: Default messaging provider
- * Name: SampleJMSCF
- * JNDI name: jms/SampleJMSCF
- Description: (empty)
- Category: (empty)

Connection

- * Bus name: SampleBus

Figure 11-2 Default messaging JMS connection factory properties

You can also configure other properties for the connection factory if needed. For detailed information about other properties, refer to *WebSphere Application Server V7 Messaging Administration Guide*, SG24-7770.

- Click **OK**. The new connection factory is created, as shown in Figure 11-3. Save the changes to the master configuration.

New	Delete	Select	Name	JNDI name	Provider
			SampleJMSCF	jms/SampleJMSCF	Default messaging provider
You can administer the following resources:					
<input type="checkbox"/>	SampleJMSCF				
Total 1					

Figure 11-3 Default messaging JMS connection factory

11.2.2 Configuring JMS destinations

You can configure both queue and topic destinations for the default messaging provider. In this section, we introduce how to create and configure a JMS queue. For information about how to create and configure JMS topics, refer to *WebSphere Application Server V7 Messaging Administration Guide*, SG24-7770.

To configure JMS destinations, complete the following steps:

1. Create a new destination in the service integration bus.

Click **Service integration → Buses**. Then, click **SampleBus**, navigate to **Destination resources → Destination**, and click **New**, as shown in Figure 11-4. Select **Queue** as the destination type, and then click **Next**.

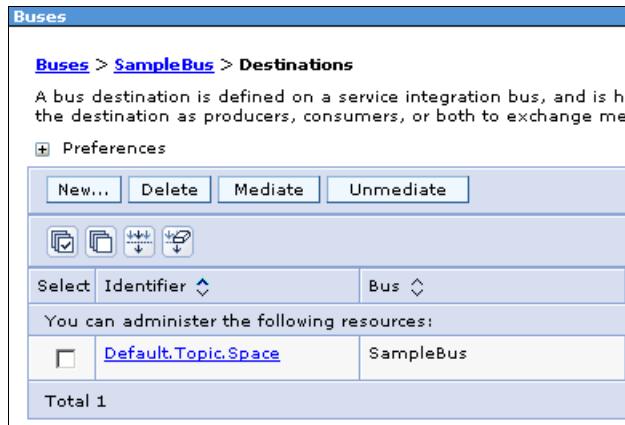


Figure 11-4 Create new destination

Enter the identifier for the queue destination, as shown in Figure 11-5. In this sample, other options use the default value.

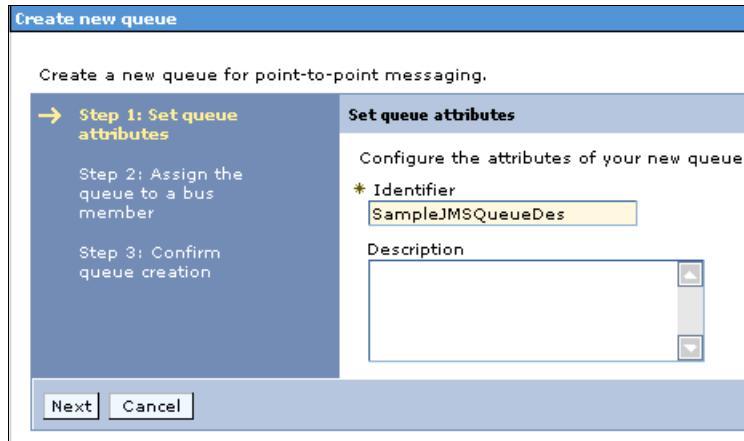


Figure 11-5 Configure the destination

Click **OK**, and save the changes to the master configuration. A new queue name *SampleQueueDes* is created.

2. Click **Resources → JMS → Queues**. In the Queues window, choose **Scope** (use the Cell scope in this sample) and then click **New**.

3. Select the **Default messaging provider** option and then enter the following basic properties:

- Name
- JNDI name
- Bus name
- Queue name

In this example, shown in Figure 11-6, the queue name is *SampleJMSQueue*.

Queues > Default messaging provider > New...

A JMS queue is used as a destination for point-to-point messaging. Use JMS queue de messaging provider.

General Properties

Administration

Scope: Cell=saw116-sys3Node01Cell

Provider: Default messaging provider

* Name: SampleJMSQueue

* JNDI name: jms/SampleJMSQueue

Description:

Connection

Bus name: SampleBus

* Queue name: SampleQueueDes

Figure 11-6 Default messaging queue properties

You can configure other properties for this queue if needed. For detailed information about other properties, refer to *WebSphere Application Server V7 Messaging Administration Guide*, SG24-7770.

4. Click **OK**. The new queue is created, as shown in Figure 11-7. Save the changes to the master configuration.

Select	Name	JNDI name	Provider
You can administer the following resources:			
<input type="checkbox"/>	SampleJMSQueue	jms/SampleJMSQueue	Default messaging provider
Total 1			

Figure 11-7 Default messaging JMS queue

11.2.3 Configuring JMS activation specifications

A JMS activation specification is associated with a message-driven bean during application installation. To configure a JMS activation specification for the default messaging provider, complete the following steps:

1. Click **Resources** → **JMS** → **Activation specifications**. In the Activation specifications window, choose **Scope** (use **Cell scope** in this sample) and then click **New**.
2. Select the **Default messaging provider** option and then enter the following basic properties:
 - Name
 - JNDI name
 - Destination type
 - Destination JNDI name
 - Bus name

In this example, shown in Figure 11-8, the activation specification name is *SampleJMSAS*.

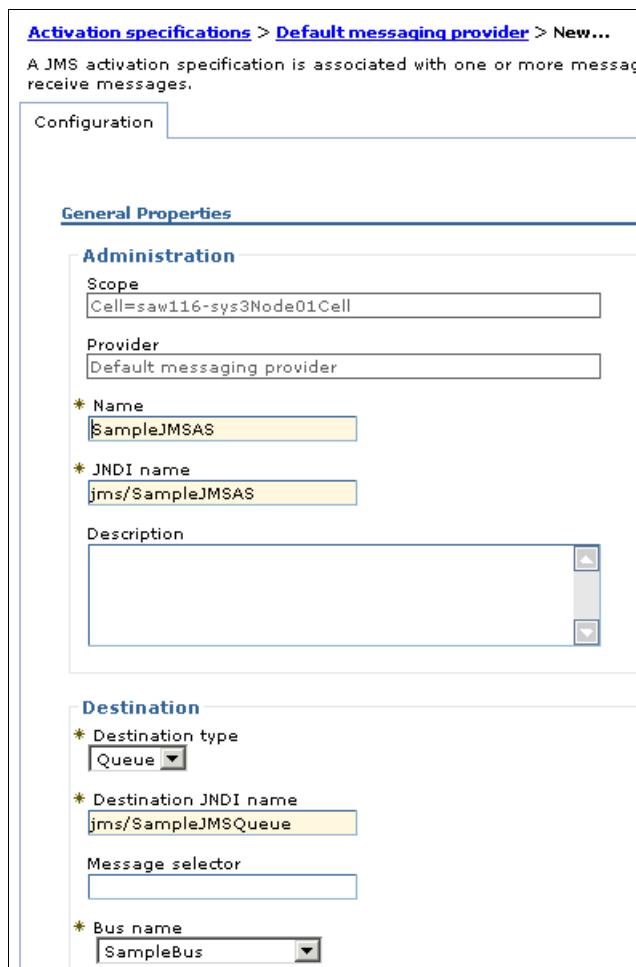


Figure 11-8 Default messaging activation specification properties

You can configure other properties for this activation specification if needed. For detailed information about other properties, refer to *WebSphere Application Server V7 Messaging Administration Guide*, SG24-7770.

- Click **OK**. The new activation specification is created, as shown in Figure 11-9. Save the changes to the master configuration.



Figure 11-9 Default messaging activation specification

11.3 Configuring the WebSphere MQ provider

In this section, we explain how to configure the WebSphere MQ messaging provider to communicate with WebSphere MQ using a bindings or client connection.

11.3.1 Configuring a connection factory

To configure a JMS connection factory for the WebSphere MQ provider, complete the following steps.

- Click **Resources** → **JMS** → **Connection factories**. In the Connection factories window, shown in Figure 11-3 on page 428, click **Scope** (use **Cell scope** in this sample), and then click **New**.
- Select the **WebSphere MQ messaging provider** option, and then click **OK**.
- Enter the name for the connection factory and the JNDI name that binds it to the name space, and then click **Next** (Figure 11-10).

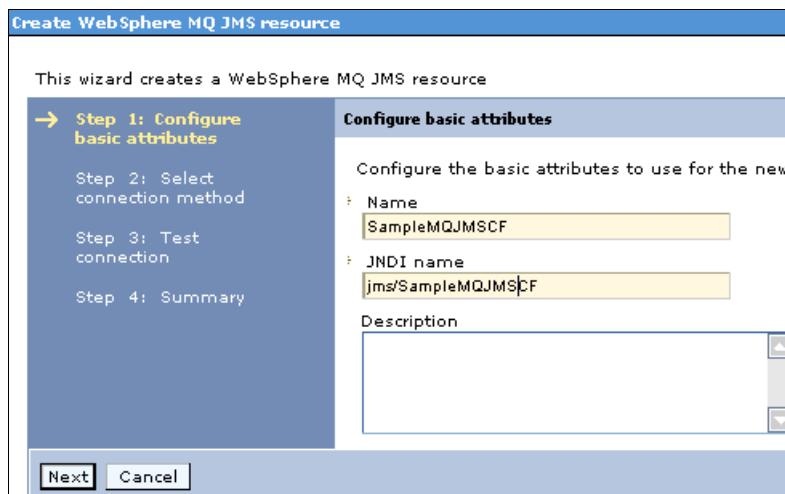


Figure 11-10 MQ messaging provider connection factory - Specify name and JNDI name

4. Select the **Enter all the required information into this wizard** option, as shown in Figure 11-11, and then click **Next**.

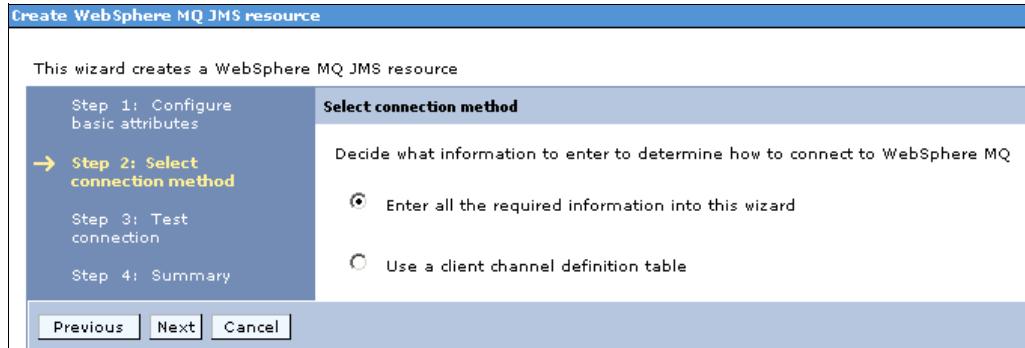


Figure 11-11 MQ messaging provider connection factory - Select connection method

For information about how to use a client channel definition table, refer to *WebSphere Application Server V7 Messaging Administration Guide*, SG24-7770.

5. Specify the queue manager or queue sharing group name, and then click **Next** (Figure 11-12).

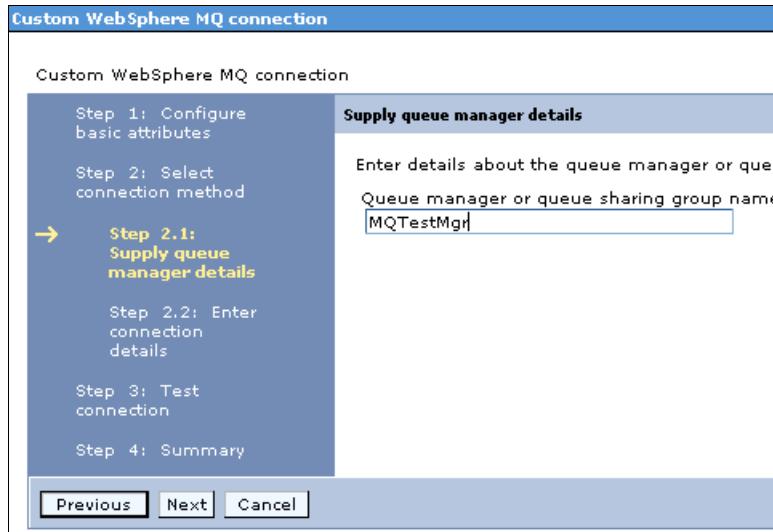


Figure 11-12 MQ messaging provider connection factory - Specify MQ queue manager name

- Choose the transport type, and specify the host name and port properties of the WebSphere MQ queue manager, as shown in Figure 11-13. The port must match the listener port that is defined for the queue manager, for example, 1424. Click **Next**.

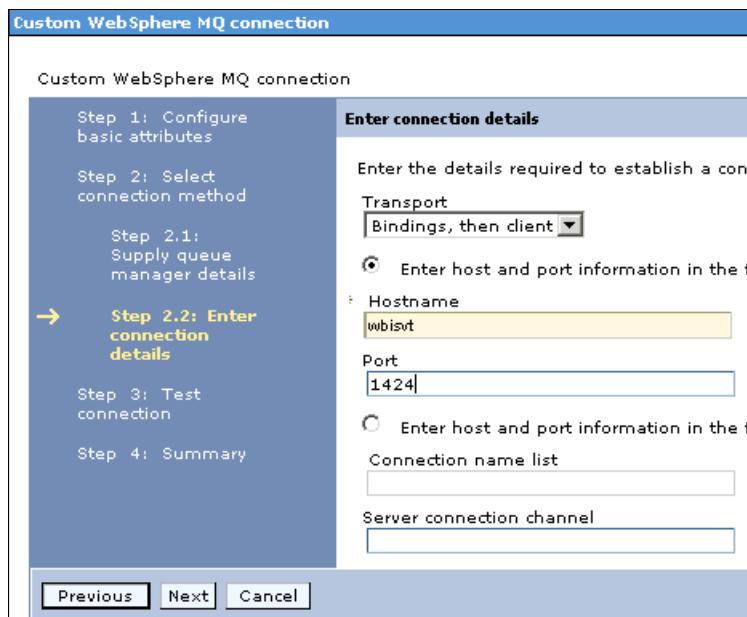


Figure 11-13 MQ messaging provider connection factory - Connection detail

New in Version 8: WebSphere Application Server V8 provides first class support for connecting to multi-instance WebSphere MQ queue managers. You can provide host and port information in the form of a connection name list, which a connection factory or activation specification uses to connect to a multi-instance queue manager.

For details about how to configure multi-instance WebSphere MQ queue manager, refer to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/fep/topic/com.ibm.websphere.nd.multiplatform.doc/info/ae/ae/tmj_wmq_miqm.html

- Click **Test connection** to verify that you can connect to the WebSphere MQ queue manager. Click **Next**.

8. In the summary window, review the summary, and then click **Finish**. The new connection factory is created, as shown in Figure 11-14. Save the changes to the master configuration.

Select	Name	JNDI name	Provider
You can administer the following resources:			
<input type="checkbox"/>	SampleJMSCF	jms/SampleJMSCF	Default messaging provider
<input type="checkbox"/>	SampleMQJMSCF	jms/SampleMQJMSCF	WebSphere MQ messaging provider
Total 2			

Figure 11-14 MQ messaging provider connection factory - Created

(New in Version 8) WebSphere Application Server V8 exposes the client reconnection properties for connection factories. You can use this property to specify whether a client mode connection reconnects automatically in the event of a communications or queue manager failure and also to specify a timeout value for reconnection attempts.

You can find this property on the Advanced properties window of a defined WebSphere MQ connection factory, as shown in Figure 11-15.

Connection factories > SampleMQJMSCF > Advanced properties

These properties are used to configure advanced functionality when using MQ.

Configuration

General Properties

Client reconnect

Client reconnect options: DISABLED

Client reconnect timeout: 1800 seconds

Figure 11-15 MQ messaging provider connection factory - New property

For detailed information about other advanced properties, refer to *WebSphere Application Server V7 Messaging Administration Guide*, SG24-7770.

11.3.2 Configuring MQ destinations

You can configure both the queue and topic destinations for the WebSphere MQ messaging provider. In this section, we introduce how to create and configure the WebSphere MQ queue destination. For information about creating and configuring the WebSphere MQ topic destination, refer to *WebSphere Application Server V7 Messaging Administration Guide*, SG24-7770.

To configure a queue destination for the WebSphere MQ messaging provider, complete the following steps.

1. Click **Resources** → **JMS** → **Queues**. In the Queues window, click **Scope** (use **Cell scope** in this sample), and then click **New**.
2. Select the **WebSphere MQ messaging provider** option, and enter the following basic properties:
 - Name
 - JNDI name
 - Queue name

In this example, shown in Figure 11-16, the queue name is *SampleMQQueue*. This name should match the queue name that is defined in the WebSphere MQ queue manager to which you are connecting.

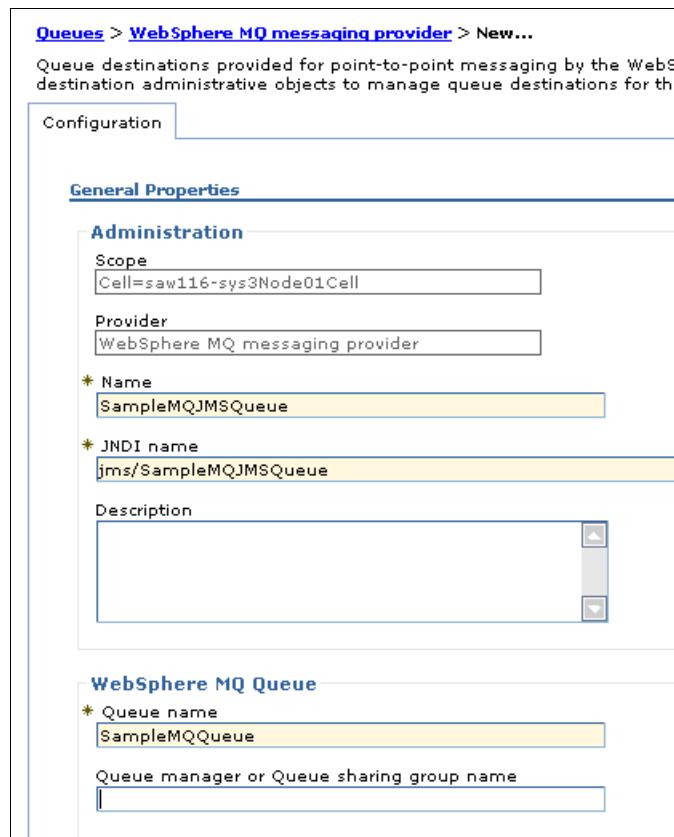


Figure 11-16 MQ messaging provider queue configuration

For detailed information about other additional properties configuration, refer to *WebSphere Application Server V7 Messaging Administration Guide*, SG24-7770.

3. Click **OK**. The new queue is created, as shown in Figure 11-17. Save the changes to the master configuration.

You can administer the following resources:			
Select	Name	JNDI name	Provider
<input type="checkbox"/>	SampleJMSQueue	jms/SampleJMSQueue	Default messaging provider
<input type="checkbox"/>	SampleMQJMSQueue	jms/SampleMQJMSQueue	WebSphere MQ messaging provider
Total 2			

Figure 11-17 MQ messaging provider queue created

(New in Version 8) WebSphere Application Server V8 provides the following new properties for WebSphere MQ queue or topic destination:

- An option on whether to append RFH Version 2 header to messages sent to this destination.

You can use this option to specify whether an application processes the RFH Version 2 header of a WebSphere MQ message as part of the JMS message body.

You can set this property in the Advanced properties window of a defined WebSphere MQ queue, as shown in Figure 11-18.

Message format

Coded character set identifier
1208

Use native encodings

Integer encoding
Normal

Decimal encoding
Normal

Floating point encoding
IEEENormal

Append RFH version 2 headers to messages sent to this destination

Figure 11-18 MQ messaging provider queue - New RHF property

► Reply to style

You can use this property to specify how the JMSReplyTo header field in a WebSphere MQ messaging provider message is generated. You can set this property in the Advanced properties window of a defined WebSphere MQ queue, as shown in Figure 11-19.

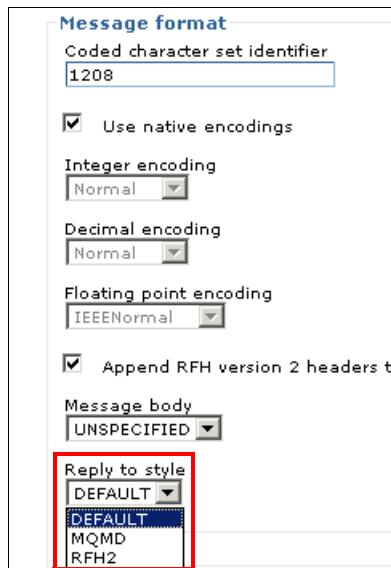


Figure 11-19 MQ messaging provider queue - New reply to style property

► Message descriptor

You can use this set of properties to specify the following information:

- Whether an application can read or write the values of MQMD fields from JMS messages that have been sent or received using the WebSphere MQ messaging provider
- Which message context options are specified when sending messages to a destination

You can set these properties in the Advanced properties window of a defined WebSphere MQ queue, as shown in Figure 11-20.



Figure 11-20 MQ messaging provider queue - New message descriptor property

11.3.3 Configuring MQ activation specifications

To configure an activation specification for WebSphere MQ messaging provider, complete the following steps:

1. Click **Resources** → **JMS** → **Activation specifications**. In the Activation specifications window, click **Scope** (use **Cell scope** in this sample), and then click **New**.
2. Select the **WebSphere MQ messaging provider** option, and enter the following basic properties:
 - Name
 - JNDI name
 - Destination type
 - Destination JNDI name
 - MQ connection propertiesFor detailed information about these properties, refer to *WebSphere Application Server V7 Messaging Administration Guide*, SG24-7770.
3. In the Summary window, review the summary, and then click **Finish**. The new activation specification is created, as shown in Figure 11-21. Save the changes to the master configuration.

Activation specifications			
You can administer the following resources:			
Select	Name	JNDI name	Provider
<input type="checkbox"/>	SampleJMSAS	jms/SampleJMSAS	Default messaging provider
<input type="checkbox"/>	SampleMQAS	jms/SampleMQAS	WebSphere MQ Resource Adapter

Figure 11-21 MQ messaging provider activation specification created

(New in Version 8) WebSphere Application Server V8 exposes the following WebSphere MQ connection properties that are used to configure the WebSphere MQ resource adapter that is used by the WebSphere MQ messaging provider. These properties affect the connection pool that is used by activation specifications:

- ▶ `maxConnections`
- ▶ `connectionConcurrency`
- ▶ `reconnectionRetryCount`
- ▶ `reconnectionRetryInterval`

11.4 Configuring a generic JMS provider

If you use a generic JMS provider, you can use the administrative console to configure JMS administered objects within the JNDI name space of the application server. In this section, we describe how to configure a generic JMS provider using the administrative console.

11.4.1 Configuring a connection factory

To configure a JMS connection factory for a generic JMS provider, complete the following steps:

1. Make sure that you have installed the provider and defined it to WebSphere Application Server. For the detailed steps, refer to *WebSphere Application Server V7 Messaging Administration Guide*, SG24-7770.
2. Click **Resources** → **JMS** → **Connection factories**. In the Connection factories window, click **Scope** (use **Cell scope** in this sample), and then click **New**.
3. Choose the generic JMS provider that you created in Step 1, and then click **OK**. Enter the following basic properties:
 - Name
 - JNDI name
 - External JNDI name

Figure 11-22 shows the configuration for this example.

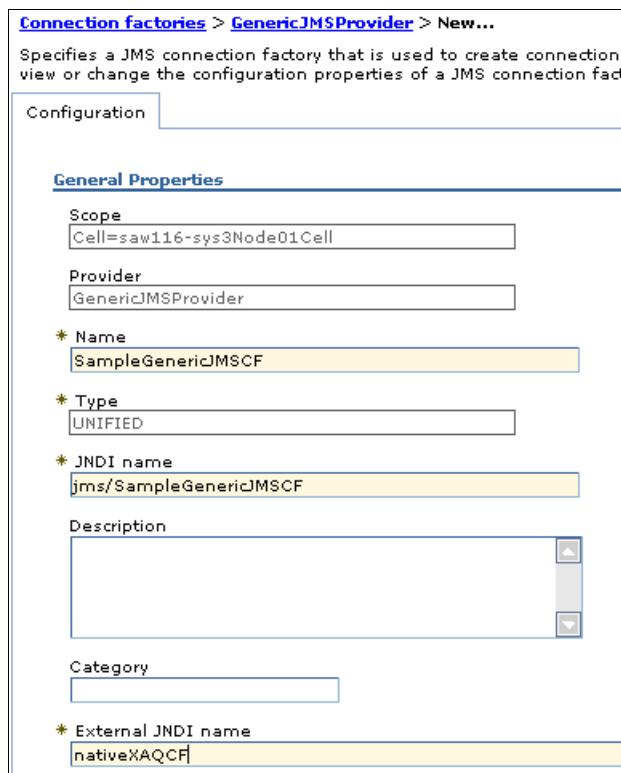


Figure 11-22 Generic JMS connection factory properties

You can configure other properties for this connection factory if needed. For detailed information about other properties, refer to *WebSphere Application Server V7 Messaging Administration Guide*, SG24-7770.

- Click **OK**. The new connection factory is created, as shown in Figure 11-23. Save the changes to the master configuration.

Select	Name	JNDI name	Provider
You can administer the following resources:			
<input type="checkbox"/>	SampleGenericJMSCF	jms/SampleGenericJMSCF	GenericJMSPublisher
<input type="checkbox"/>	SampleJMSCF	jms/SampleJMSCF	Default messaging provider
<input type="checkbox"/>	SampleMQJMSCF	jms/SampleMQJMSCF	WebSphere MQ messaging provider

Figure 11-23 Generic JMS connection factory created

11.4.2 Configuring JMS destinations

You can configure both queue and topic destinations for the generic JMS messaging provider. In this section, we introduce how to create and configure a JMS queue. For information about creating and configuring a JMS topic, refer to *WebSphere Application Server V7 Messaging Administration Guide*, SG24-7770.

To configure JMS destination, complete the following steps:

- Click **Resources** → **JMS** → **Queues**. In the Queues window, click **Scope** (use **Cell scope** in this sample), and then click **New**.
- Select the **Generic JMS provider** option, and then enter the following basic properties:
 - Name
 - JNDI name
 - External JNDI name

Figure 11-24 shows the configuration for this example.

The screenshot shows the 'Queues > GenericJMSProvider > New...' configuration dialog. It includes fields for General Properties such as Scope (Cell=saw116-sys3Node01Cell), Provider (GenericJMSProvider), Name (SampleGenericJMSQueue), Type (QUEUE), JNDI name (jms/SampleGenericJMSQueue), Description (empty), Category (empty), and External JNDI Name (nativeReceiveQueue).

Figure 11-24 Generic JMS queue properties

You can configure other properties for this queue if needed. For detailed information about the properties, refer to *WebSphere Application Server V7 Messaging Administration Guide*, SG24-7770.

3. Click **OK**. The new queue is created, as shown in Figure 11-25. Save the changes to the master configuration.

Select	Name	JNDI name	Provider
You can administer the following resources:			
<input type="checkbox"/>	SampleGenericJMSQueue	jms/SampleGenericJMSQueue	GenericJMSProvider
<input type="checkbox"/>	SampleJMSQueue	jms/SampleJMSQueue	Default messaging provider
<input type="checkbox"/>	SampleMQJMSQueue	jms/SampleMQJMSQueue	WebSphere MQ messaging provider

Figure 11-25 Generic JMS queue created



Configuring and managing web servers

This chapter provides details about configuring and managing web servers in a WebSphere Application Server environment.

In this chapter, we cover the following topics:

- ▶ Web server support overview
- ▶ Installation
- ▶ Web server configuration using WebSphere Customization Toolbox (WCT)
- ▶ Working with web servers
- ▶ Working with the plug-in configuration file
- ▶ IBM HTTP Server and Web Server Plug-ins for IBM WebSphere Application Server for z/OS

12.1 Web server support overview

WebSphere Application Server provides web server plug-ins that work with a web server to route requests for dynamic content, such as servlets, from the web server to the proper application server. A web server plug-in is specific to the type of web server. It is installed on the web server machine and configured in the web server configuration.

A plug-in configuration file generated on the application server and placed on the web server is used for routing information. To manage the generation and propagation of these plug-in configuration files, web servers are defined to the WebSphere Application Server configuration repository. In some cases, web server configuration and management features are also available from the WebSphere administrative tools.

Web servers are defined to WebSphere Application Server. A web server resides on a managed or unmanaged node. If located on a managed node in a distributed server environment only, a node agent is installed on the web server system and belongs to a WebSphere Application Server administrative cell. The administrative tools communicate with the web server through the node agent.

If located on an unmanaged node, the web server is defined to the cell, but does not have a node agent running to manage the process. In either case, the web server definition allows you to generate the plug-in configuration file for the web server.

Web applications can be mapped to web servers. This mapping is used to generate routing information during plug-in configuration generation.

IBM HTTP Server V8 is bundled with WebSphere Application Server V8. The administrative functionality is integrated into WebSphere Application Server to provide remote administration through the administrative console. This enhanced administrative function is only available to the IBM HTTP Server.

The following web servers are the supported web servers for WebSphere Application Server V8:

- ▶ Apache HTTP Server
- ▶ IBM Lotus Domino® Web Server
- ▶ IBM HTTP Server
- ▶ Microsoft Internet Information Services
- ▶ Sun Java System Web Server (formerly Sun ONE and iPlanet)

For the latest list of supported web servers and the versions supported, see the prerequisite document at the following website:

<http://www.ibm.com/software/webservers/appserv/doc/latest/prereq.html>

12.1.1 Request routing using the plug-in

The web server plug-in uses an XML configuration file to determine whether a request is for the web server or the application server. When a request reaches the web server, the URL is compared to the URLs managed by the plug-in. If a match is found, the plug-in configuration file contains the information needed to forward that request to the web container using the web container inbound transport chain. See Figure 12-1.

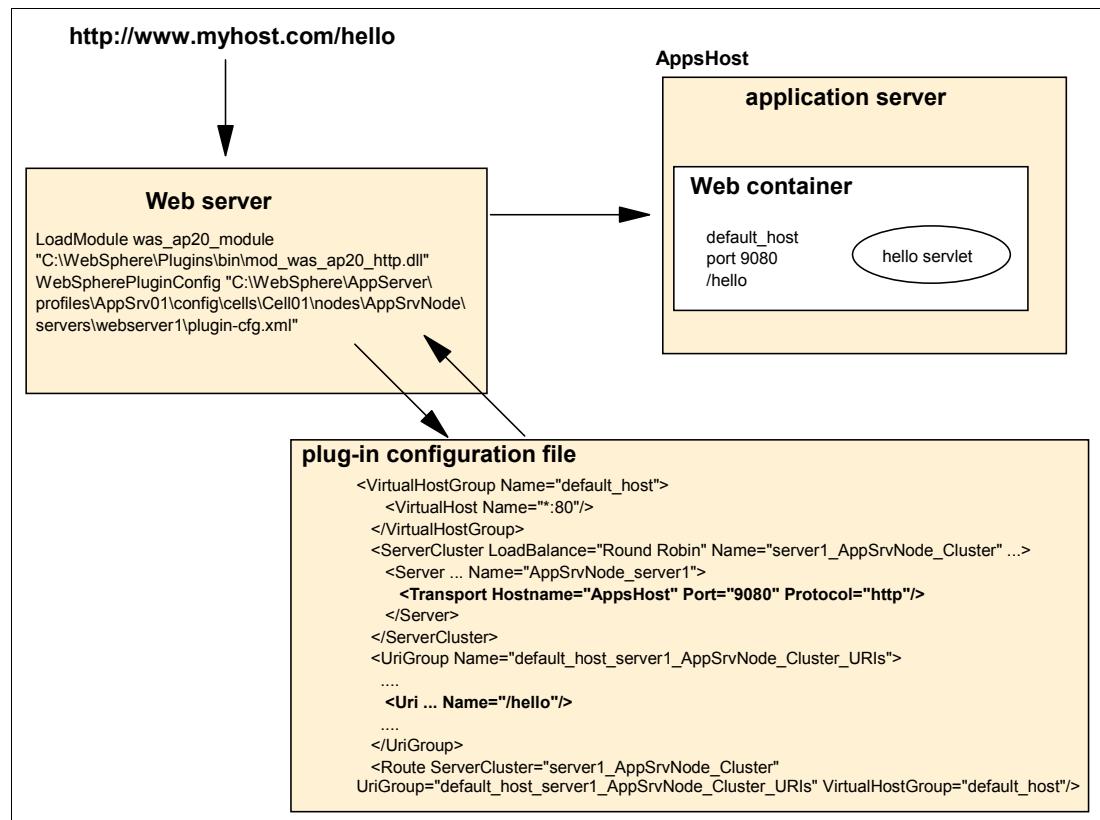


Figure 12-1 Web server plug-in routing

The plug-in configuration file is generated using the WebSphere administrative tools. Each time you make a change to the WebSphere Application Server configuration that would affect how requests are routed from a web server to the application server, you need to regenerate and propagate the plug-in configuration file to the web server. You can propagate the file manually or configure the propagation to be done automatically.

12.1.2 Web server and plug-in management

The setup of your web server and web server plug-in environment is defined in a web server definition. The web server definition includes information about the location of the web server, its configuration files, and plug-in configuration. Each web server is associated with a node, either managed or unmanaged. The web server definition is configured as part of the plug-in installation process. The web server definition is also used during application deployment. Web modules can be mapped to a web server, ensuring the proper routing information is generated for the plug-in configuration file.

Web server definitions are located under **Servers** → **Web servers** in the administrative console. See Figure 12-2.

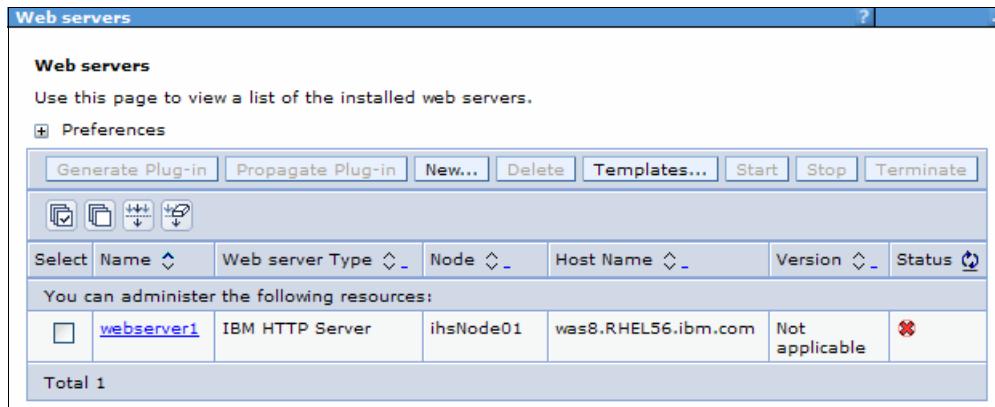


Figure 12-2 Web server definition

Contrasting managed and unmanaged

In a distributed server environment, you can define multiple web servers. These web servers can be defined on managed or unmanaged nodes. It is important to understand the concept of managed versus unmanaged nodes. A supported web server can be on a managed node or an unmanaged node, depending on the environment in which you are running the web server.

WebSphere Application Server supports basic administrative functions for all supported web servers. For example, generation of a plug-in configuration can be performed for all web servers. If the web server is defined on a managed node, automatic propagation of the plug-in configuration can be performed using node synchronization. If the web server is defined on an unmanaged node, automatic propagation of a plug-in configuration is only supported for IBM HTTP Servers.

WebSphere Application Server supports some additional administrative console tasks for IBM HTTP Servers on managed and unmanaged nodes. For example, you can start IBM HTTP Servers, stop them, terminate them, display their log files, and edit their configuration files.

Unmanaged nodes

An *unmanaged node* does not have a node agent to manage its servers. In a stand-alone server environment, you can define one web server and it, by necessity, resides on an unmanaged node. In a distributed server environment, web servers defined to an unmanaged node are typically remote web servers. The web server is usually found in the demilitarized zone (DMZ) outside the firewall. The DMZ is a safe zone between firewalls that is typically located between a client and a back-end server.

Figure 12-3 displays a web server configured on an unmanaged node. In this configuration, the plug-in configuration file is generated on the deployment manager. The plug-in configuration file must be manually propagated to the web server on the unmanaged node. To start or stop this web server, you need to use the specific web server administrative tools.

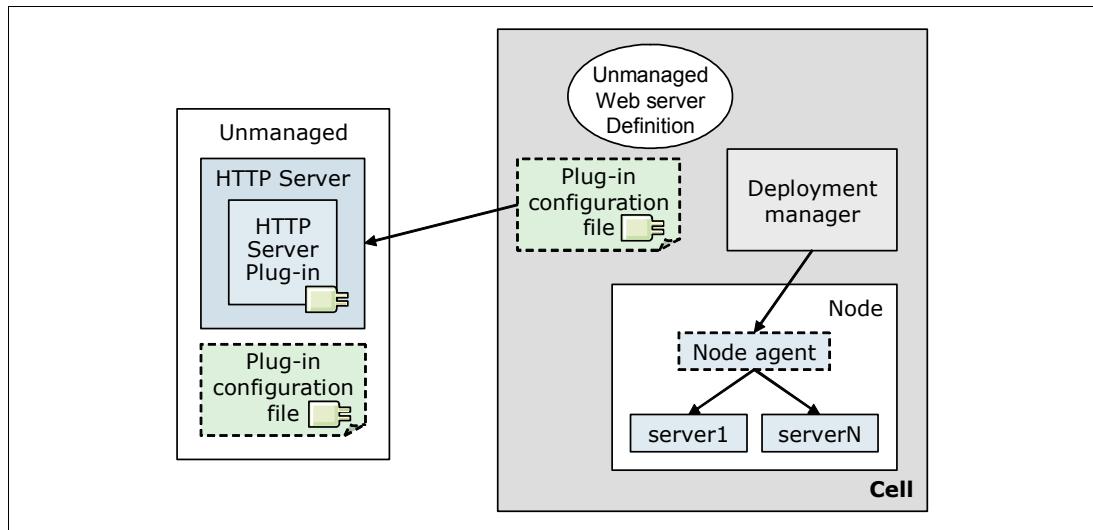


Figure 12-3 Unmanaged web server on unmanaged node

The IBM HTTP Server is a special case. If your web server is the IBM HTTP Server, you can configure the server on an unmanaged node, and still be able to administer the web server using the WebSphere administrative tools. This unmanaged node does not need a node agent on the web server machine. The IBM HTTP Server administration process provides administrative functions for the IBM HTTP Server within WebSphere.

Figure 12-4 shows an IBM HTTP Server web server configured on an unmanaged node. In this configuration, the plug-in configuration file is generated on the deployment manager. Because the web server is the IBM HTTP Server, you can automatically propagate the plug-in configuration file to the remote server, make configuration changes to the web server configuration file, and use the administrative console to start and stop the web server.

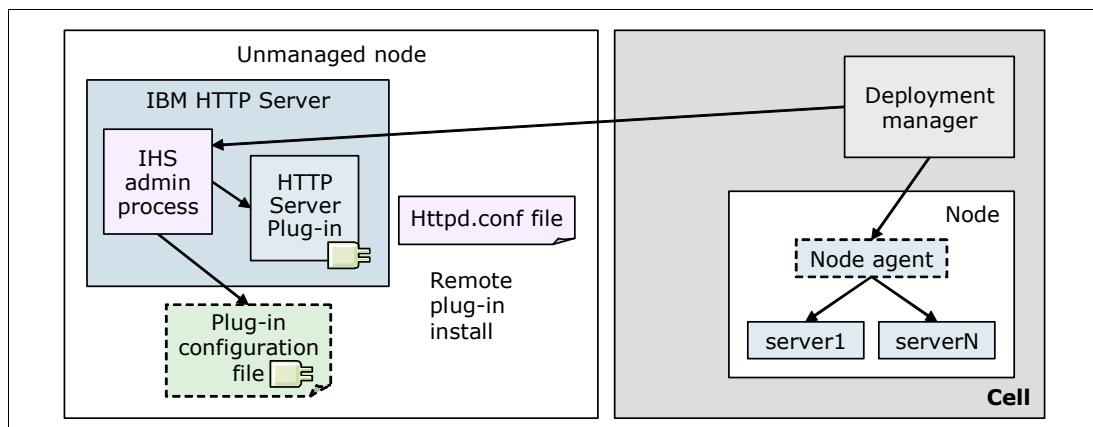


Figure 12-4 IBM HTTP Server on an unmanaged node

If the web server is defined to an unmanaged node, complete the following steps:

1. Check the status of the web server.
2. Generate a plug-in configuration file for that web server.

If the web server is an IBM HTTP Server and the IBM HTTP Server Administration server is installed and properly configured, you can also:

- a. Display the IBM HTTP Server Error log (`error.log`) and Access log (`access.log`) files.
- b. Start and stop the server.
- c. Display and edit the IBM HTTP Server configuration file (`httpd.conf`).
- d. Propagate the plug-in configuration file after it is generated.

You cannot propagate an updated plug-in configuration file to a non-IBM HTTP Server web server that is defined to an unmanaged node. You must install an updated plug-in configuration file manually to a web server that is defined to an unmanaged node.

Managed nodes

A *managed node* has a node agent for managing web servers. You can use the WebSphere administrative tools to communicate with web servers on a managed node. Figure 12-5 displays a web server configured on a managed node. In this configuration, the plug-in configuration file is generated on the deployment manager. The plug-in configuration file is automatically propagated to the web server on the managed node. To start or stop this web server, you can use the administrative console.

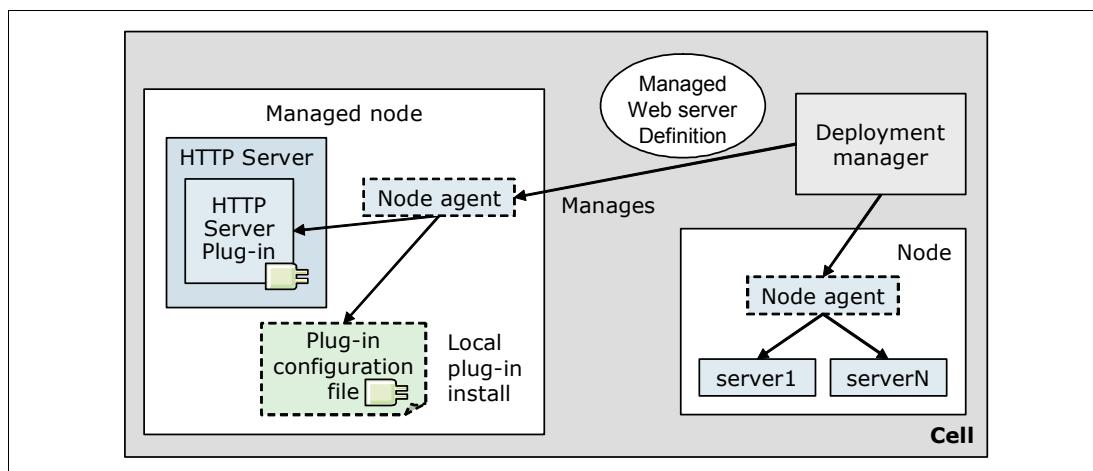


Figure 12-5 Managed web server on a managed node

If the web server is defined to a managed node, complete the following steps:

1. Check the status of the web server.
2. Generate a plug-in configuration file for that web server.
3. Propagate the plug-in configuration file after it is generated.

If the web server is an IBM HTTP Server and the IBM HTTP Server Administration server is installed and properly configured, you can also:

- a. Display the IBM HTTP Server Error log (`error.log`) and Access log (`access.log`) files.
- b. Start and stop the server.
- c. Display and edit the IBM HTTP Server configuration file (`httpd.conf`).

How nodes and servers are defined

During the installation of the plug-in, the Plug-ins installation wizard creates a web server configuration script named `configureWeb_server_name`. This configuration script is used to create the web server definition and, if necessary, the node definition in the configuration of the application server.

If a web server definition already exists for a stand-alone application server, running the script does not add a new web server definition. Each stand-alone application server can have only one web server definition. A managed node, conversely, can have multiple web server definitions. The script creates a new web server definition unless the web server name is the same.

The Plug-ins installation wizard stores the script in the `<plug-in_home>/bin` directory on the web server machine. If the plug-in is installed locally (on the same machine as the application server), the configuration script will be run automatically.

For remote installations, you must copy the script from the web server machine to the `<was_home>/bin` directory on the application server machine for execution. The script runs against the default profile. If one machine is running under Linux or UNIX and the other machine is running under Windows, use the script created in the `<plug-in_home>/Plugins/bin/crossPlatformScripts` directory.

Note: Always open a new command window in which to execute the `configureWeb_server_name` script. There is a potential conflict between a shell environment variable, the `WAS_USER_SCRIPT` variable, and the real default profile. The script always works against the default profile. However, if the `WAS_USER_SCRIPT` environment variable is set, a conflict arises as the script attempts to work on the profile identified by the variable.

If you need to create a web server definition for a distributed server environment, you must federate your stand-alone application servers to the deployment manager first. Any web server definitions created for a stand-alone application server will be lost when they are federated into a cell.

Using administrative tools: In a distributed server environment, the administrative console can also be used to define the nodes and web servers. For more details, see 12.4.1, “Manually defining nodes and web servers” on page 465.

12.2 Installation

You can install a web server and web server plug-in using the Installation Manager. You can perform the installation using the GUI, command line, console mode, or silently. To install the IBM HTTP Server and web server plug-in, complete the following steps:

1. Install IBM Installation Manager.
2. Launch the Installation Manager GUI.
3. Configure the Installation Manager repository to point to the Supplements package.
4. Click the Install wizard to begin the installation.
5. Select the following packages for installation:
 - IBM HTTP Server for WebSphere Application Server
 - Web Server Plug-ins for IBM WebSphere Application Server

Click **Next**.

6. Read and accept the license agreement. Click **Next**.
7. Select the installation directory for each package. You can keep the default paths or update them to suit your environment. Click **Next**.
8. On the Features window, verify the selected packages and click **Next**.
9. Configure a port number for the IBM HTTP Server to communicate. Keep the default port 80 or modify to a port number that is not in use. Click **Next**.
10. Review the settings on the Summary window and click **Install**.
11. When the installation is complete, review the summary. Click **Finish**. It is also a best practice to view the log file to verify that the installation was successful.

After installation of the web server and web server plug-in, you need to configure the web server plug-in. To configure the plug-in, the stand-alone WebSphere Customization Toolbox offering must be installed. You can find more information about the stand-alone WebSphere Customization Toolbox offering in 2.6, “WebSphere Customization Toolbox” on page 70.

12.3 Web server configuration using WebSphere Customization Toolbox (WCT)

(New in Version 8) After installing the web server plug-in, you need to configure it. A new tool in WebSphere Application Server V8 is the Web Server Plug-in Configuration Tool in the WebSphere Customization Toolbox, which is used for configuring web server plug-ins. The Web Server Plug-in Configuration Tool creates one or more configurations for the web server plug-ins that can direct requests from a web client through the web server, and then interact with applications running on an application server. The Web Server Plug-in Configuration Tool edits the configuration file or files for a web server by creating directives that point to the location of the binary plug-in module and the plug-in configuration file.

Before configuring the plug-in, you need to determine the topology setup. The options for defining and managing web servers depend on your chosen web server topology and your WebSphere Application Server package. Decisions to make include whether to collocate the web server with other WebSphere Application Server processes, and whether to make the web server managed or unmanaged.

The following examples outline the process required to create each sample topology. Note that each example assumes that only the WebSphere processes shown in the diagrams are installed on each system and that the profile for the process is the default profile.

This section is not a substitute for using the product documentation, but is intended to help you understand the process. For detailed information about how to complete a local or remote installation scenario, see the *Web Server Plug-in Installation Roadmaps for WebSphere Application Server Network Deployment Version 8.0* guide that comes with the plug-in. You can also find this information at the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.nd.doc/info/ae/ae/tins_road_plugins.html

12.3.1 Configuration files

When configuring a web server and plug-in, a number of configuration files are created and used:

- ▶ Web server configuration file:

This file is installed as part of the web server. For example, if you install the IBM HTTP Server, the web server configuration file is `httpd.conf`. During configuration of the plug-in, the Web Server Plug-in Configuration Tool adds directives to the file specifying the location of the binary web server plug-in file and the plug-in configuration file (`plugin-cfg.xml`).

- ▶ Binary web server plug-in file:

This file resides on the web server machine. An example of a binary web server plug-in file on Linux for the IBM HTTP Server is `mod_was_ap22_http.do`. The configuration file for the binary web server plug-in is the `plugin-cfg.xml` file. The binary module reads this XML file to discover where to route requests.

- ▶ Plug-in configuration file (`plugin-cfg.xml`):

The plug-in configuration file is generated on an application server or deployment manager machine and must be copied to the web server machine. As you make changes to your environment, such as deploying applications, creating clusters, updating virtual hosts, and so on, this information needs to be placed in the plug-in configuration file to ensure proper routing of content. The plug-in configuration file needs to be generated and then propagated to the web server. See 12.5.1, “Regenerating the plug-in configuration file” on page 477 and 12.5.2, “Propagating the plug-in configuration file” on page 483 for more information.

- ▶ Default (temporary) plug-in configuration file:

This is a temporary plug-in configuration file that is generated by the Web Server Plug-in Configuration Tool for every remote installation scenario. This file is replaced by the Plug-in configuration file (`plugin-cfg.xml`) that is relevant for your environment.

- ▶ The `configureweb_server_name` script:

This script is created by the Web Server Plug-in Configuration Tool on the web server machine. Copy this script to the `profile_root/bin` directory of the application server or deployment manager. You need to run the script to create a web server definition in the application server or deployment manager configuration.

12.3.2 Stand-alone server environment

In a stand-alone server environment, a web server can be remote to the application server machine or local, but there can only be one defined to WebSphere Application Server. The web server always resides on an unmanaged node.

Remote web server

In this scenario, the application server and the web server are on separate machines. The web server machine can reside in the internal network, or more likely, will reside in the DMZ. Use this configuration for a production environment. See Figure 12-6.

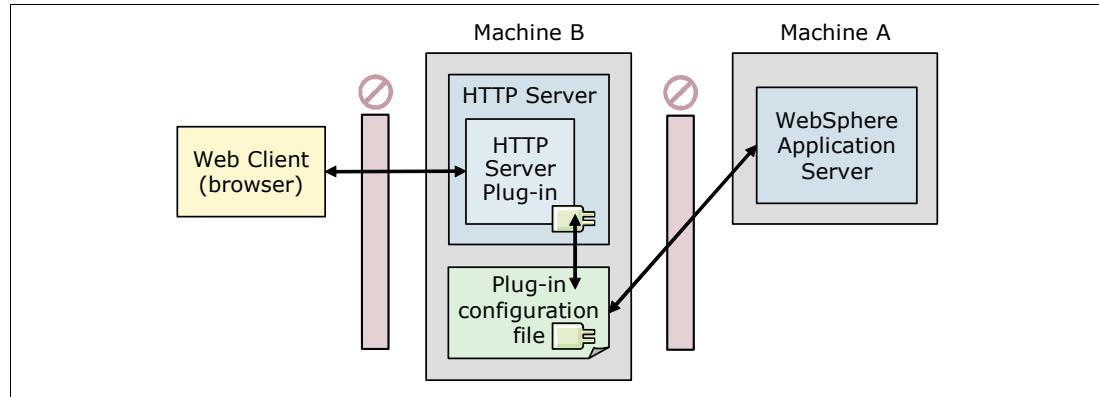


Figure 12-6 Remote web server in a stand-alone server environment

Assume the application server is already installed and configured on Machine A. Complete the following steps:

1. Install Installation Manager on Machine B.
2. Install the web server, web server plug-in, and WebSphere Customization Toolbox on Machine B.
3. Configure the web server plug-in on Machine B by completing the following steps:
 - a. Launch the Web Server Plug-ins Configuration Tool.
 - b. Configure a *remote* web server plug-in.
 - c. Configure a web server definition. The default is *webserver1*.

During configuration, the following tasks are completed:

- a. A default temporary plug-in configuration file is created and placed into the location specified.
- b. The web server configuration file is updated with the plug-in configuration, including the location of the plug-in configuration file.
- c. A script is generated to define the web server to WebSphere Application Server. The script is located in:

`<plug-in_home>/bin/configure<web_server_name>`

4. At the end of the plug-in configuration, copy the `configure<web_server_name>` script to the `<profile_root>/bin` directory of the application server machine, Machine A. Start the application server, and then execute the script.
5. When the web server is defined to WebSphere Application Server, the plug-in configuration file is generated automatically. For the IBM HTTP Server, the new plug-in file will be propagated to the web server automatically. For other web server types, you need to propagate the new plug-in configuration file to the web server.
6. Start the web server, and verify the configuration by accessing an application from a web client.

Local web server

In this scenario, a stand-alone application server exists on machine A. The web server and web server plug-in will also be installed on machine A. This topology is suited to a development environment or for internal applications. See Figure 12-7.

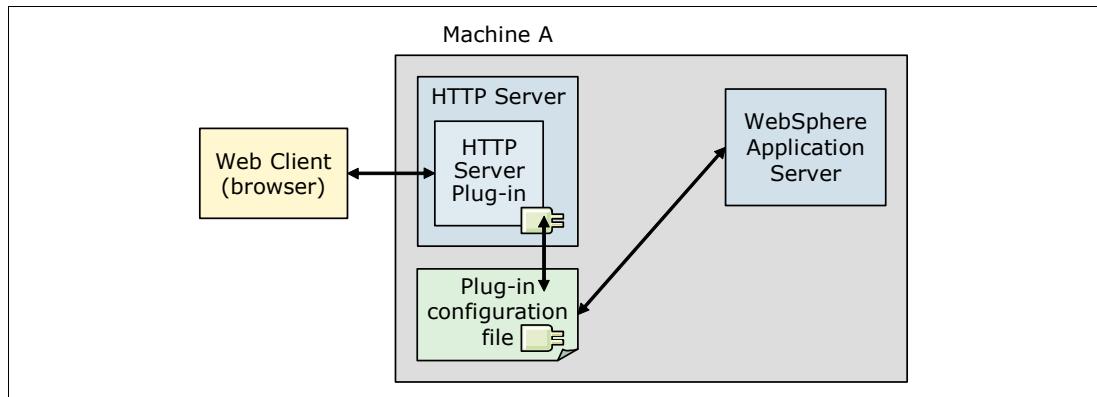


Figure 12-7 Local web server in a stand-alone server environment

Assume the application server is already installed and configured. Complete the following steps:

1. Install the web server, web server plug-in, and WebSphere Customization Toolbox on Machine A.
2. Configure the web server plug-in on Machine A by completing the following steps:
 - a. Launch the Web Server Plug-ins Configuration Tool.
 - b. Configure a *local* web server plug-in.
 - c. Configure a web server definition. The default is `webserver1`.

During configuration, the following tasks are performed:

- a. A default temporary plug-in configuration file is created and placed into the location specified.
- b. The web server configuration file is updated with the plug-in configuration, including the location of the plug-in configuration file.
- c. A script to define the web server to WebSphere Application Server is generated. The script is located in:

```
<plug-in_home>/bin/configure<web_server_name>
```

3. At the end of the plug-in configuration, copy the `configure<web_server_name>` script to the `<profile_root>/bin` directory of the application server machine, Machine A. Start the application server, and then execute the script.
4. When the web server is defined to WebSphere Application Server, the plug-in configuration file is generated automatically. Because this is a local installation, you do not have to propagate the new plug-in configuration to the web server. For the IBM HTTP Server, the new plug-in file will be propagated to the web server automatically. For other web server types, you need to propagate the new plug-in configuration file to the web server.
5. Start the application server and the web server, and verify the configuration by accessing an application from a web client.

12.3.3 Distributed server environment

Web servers in a distributed server environment can be local to the application server or remote. The web server can also reside on the deployment manager system. You have the possibility of defining multiple web servers and the web servers can reside on managed or unmanaged nodes.

Note: If you are planning to add the application server node into a deployment manager cell but have not done so yet, start the deployment manager and federate the node before configuring the plug-in. You cannot add an application server with a web server definition into the deployment manager cell.

Remote web server

The deployment manager and the web server are on separate machines. The web server machine can reside in the internal network, or more likely, it resides in the DMZ. Use this configuration for a production environment.

Note that this scenario and the process are almost identical to that outlined for a remote web server in a stand-alone server environment. The primary difference is that the script that defines the web server is run against the deployment manager and you will see an unmanaged node created for the web server node. In Figure 12-8, the node is unmanaged because there is no node agent on the web server system.

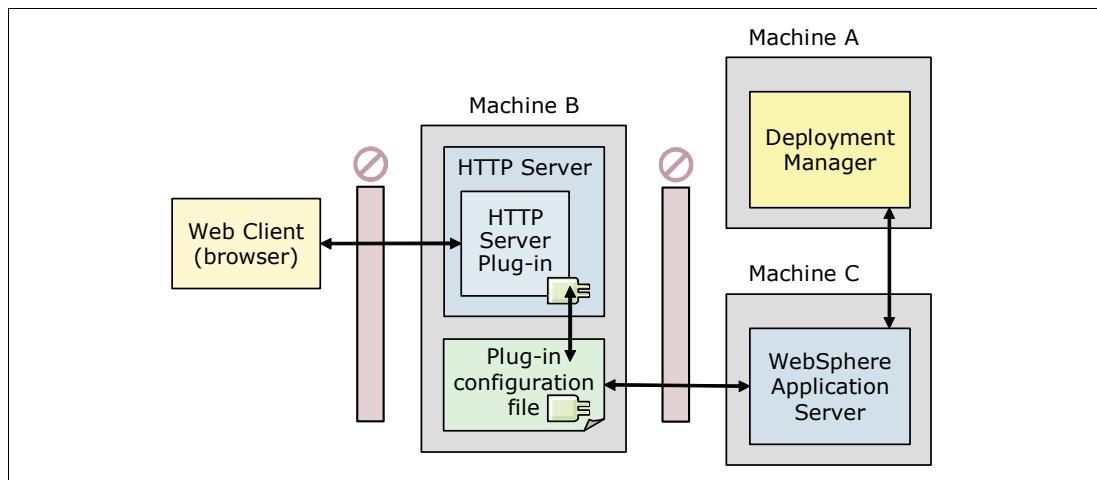


Figure 12-8 Remote web server in a stand-alone server environment

Assume that the deployment manager is already installed and configured on Machine A and the application server is already installed and configured on Machine C. Complete the following steps:

1. Install Installation Manager on Machine B.
2. Install the web server, web server plug-in, and WebSphere Customization Toolbox on Machine B.
3. Configure the web server plug-in on Machine B by completing the following steps:
 - a. Launch the Web Server Plug-ins Configuration Tool.
 - b. Configure a *remote* web server plug-in.
 - c. Configure a web server definition. The default is `webserver1`.

During configuration, the following tasks are performed:

- a. A default temporary plug-in configuration file is created and placed into the location specified.
- b. The web server configuration file is updated with the plug-in configuration, including the location of the plug-in configuration file.
- c. A script is generated to define the web server to WebSphere Application Server. The script is located in:

```
<plug-in_home>/bin/configure<web_server_name>
```

4. At the end of the plug-in configuration, copy the `configure<web_server_name>` script to the `<profile_root>/bin` directory of the deployment manager machine, Machine A. Start the deployment manager, and then execute the script.
5. When the web server is defined to WebSphere Application Server, the plug-in configuration file is generated automatically. For the IBM HTTP Server, the new plug-in file will be propagated to the web server automatically. For other web server types, you need to propagate the new plug-in configuration file to the web server.
6. Start the web server, and verify the configuration by accessing an application from a web client.

Local to a federated application server

In this scenario, the web server is installed on a machine that also has a managed node. Note that this scenario would also be the same if the deployment manager was also installed on Machine B. See Figure 12-9.

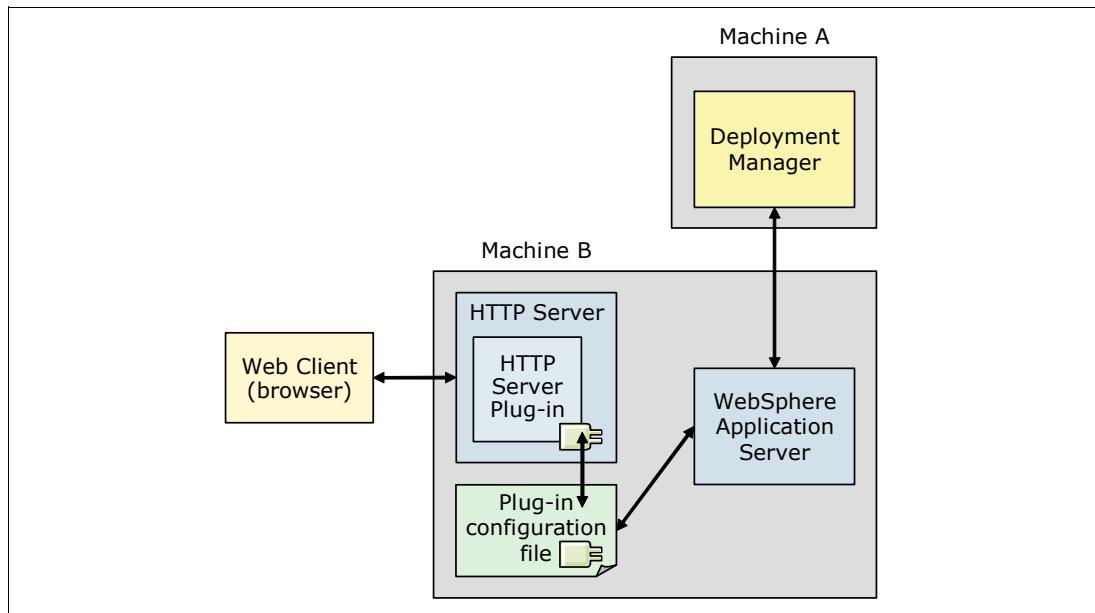


Figure 12-9 Web server installed locally on an application server system

Assume that the deployment manager is already installed and configured on Machine A and the application server is already installed and configured on Machine B. Complete the following steps:

1. Install the web server, web server plug-in, and WebSphere Customization Toolbox on Machine B.

2. Configure the web server plug-in on Machine B by doing the following:

- a. Launch the Web Server Plug-ins Configuration Tool.
- b. Configure a *local* web server plug-in.
- c. Configure a web server definition. The default is `webserver1`.

During configuration, the following tasks are performed:

- a. A default temporary plug-in configuration file is created and placed into the location specified.
- b. The web server configuration file is updated with the plug-in configuration, including the location of the plug-in configuration file.
- c. A script is generated to define the web server to WebSphere Application Server. The script is located in:

```
<plug-in_home>/bin/configure<web_server_name>
```

3. At the end of the plug-in configuration, copy the `configure<web_server_name>` script to the `<profile_root>/bin` directory of the deployment manager machine, Machine A. Start the deployment manager, and then execute the script.
4. When the web server is defined to WebSphere Application Server, the plug-in configuration file is generated automatically. For the IBM HTTP Server, the new plug-in file will be propagated to the web server automatically. For other web server types, you need to propagate the new plug-in configuration file to the web server.
5. Start the web server, and verify the configuration by accessing an application from a web client.

The plug-in configuration file will be generated automatically and will be propagated at the next node synchronization.

12.3.4 Configuring a remote web server in a distributed environment

Assume that the Installation Manager, deployment manager, and application server have all been installed and configured. The IBM HTTP Server and web server plug-in are also installed. Complete the following steps to configure a remote web server in a distributed environment using the Web Server Plug-ins Configuration Tool:

1. Launch the stand-alone WebSphere Customization Toolbox.
2. Select the Web Server Plug-ins Configuration Tool and click **Launch Selected Tool**.
3. Select a plug-in runtime location. Click **Add** in the Web Server Plug-in Runtime Locations tab.
 - Enter a name for the plug-in location.
 - Browse to the location of where the plug-in is installed.

See Figure 12-10.

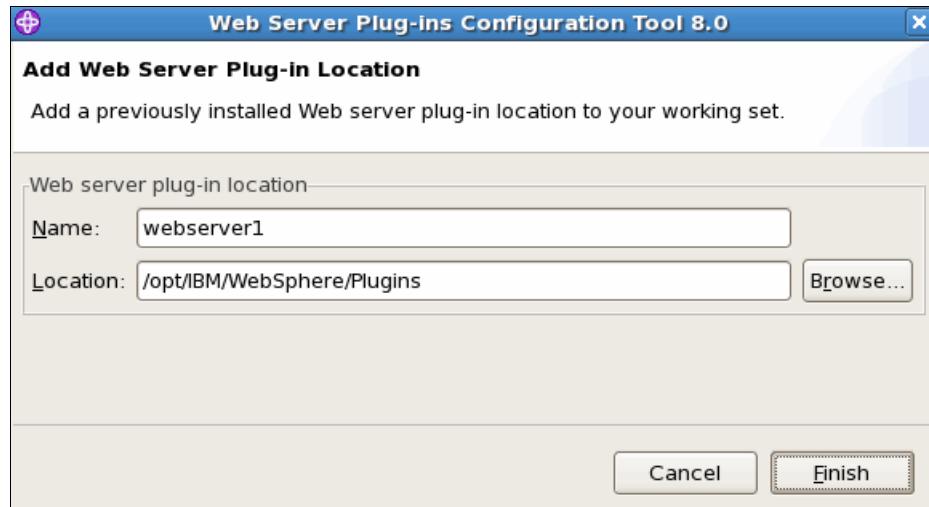


Figure 12-10 Web server plug-in location

Click **Finish**.

4. Add the web server configuration information. In the Web Server Plug-in Configurations area, click **Create**.
5. Select the web server you want to configure and click **Next**. See Figure 12-11.



Figure 12-11 Web server selection

6. Select the existing HTTP Server configuration file and provide the web server port. See Figure 12-12.

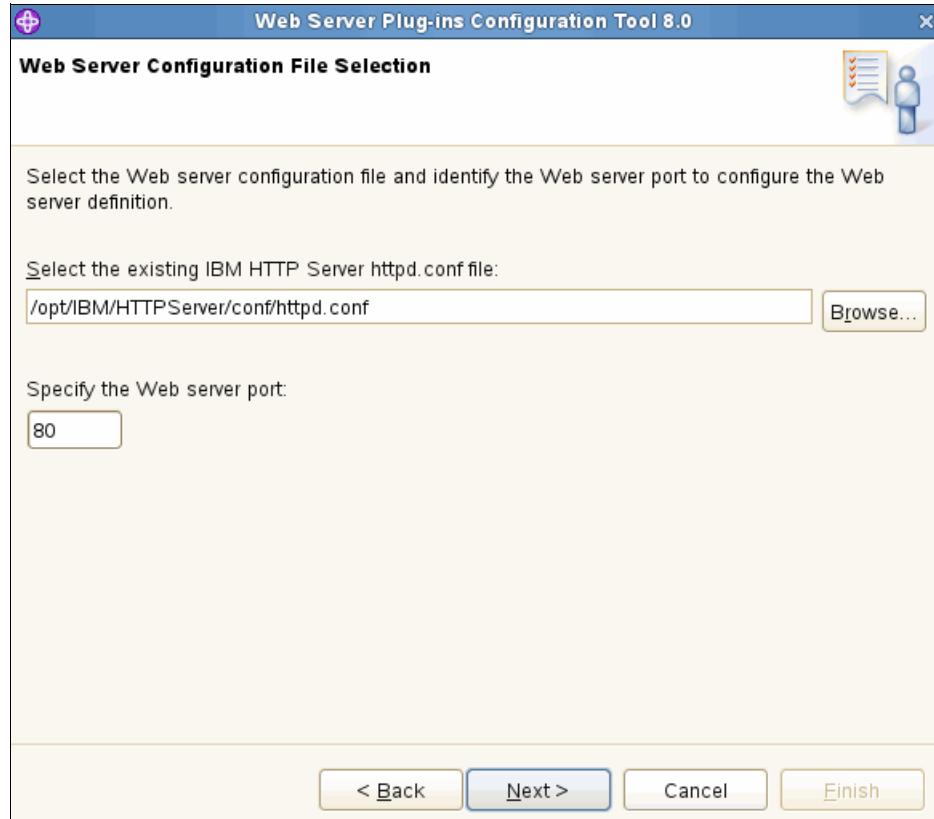


Figure 12-12 Web server configuration file selection

Click **Next**.

7. If you are configuring an IBM HTTP web server plug-in, complete the following steps:
 - a. Enter a port number for the administration server. Keep the default of 8008 or enter a unique port.

- b. Create a user ID for the administration server authentication. In Figure 12-13, the user ID is *ihsadmin* and the Password is *ihsadmin*.

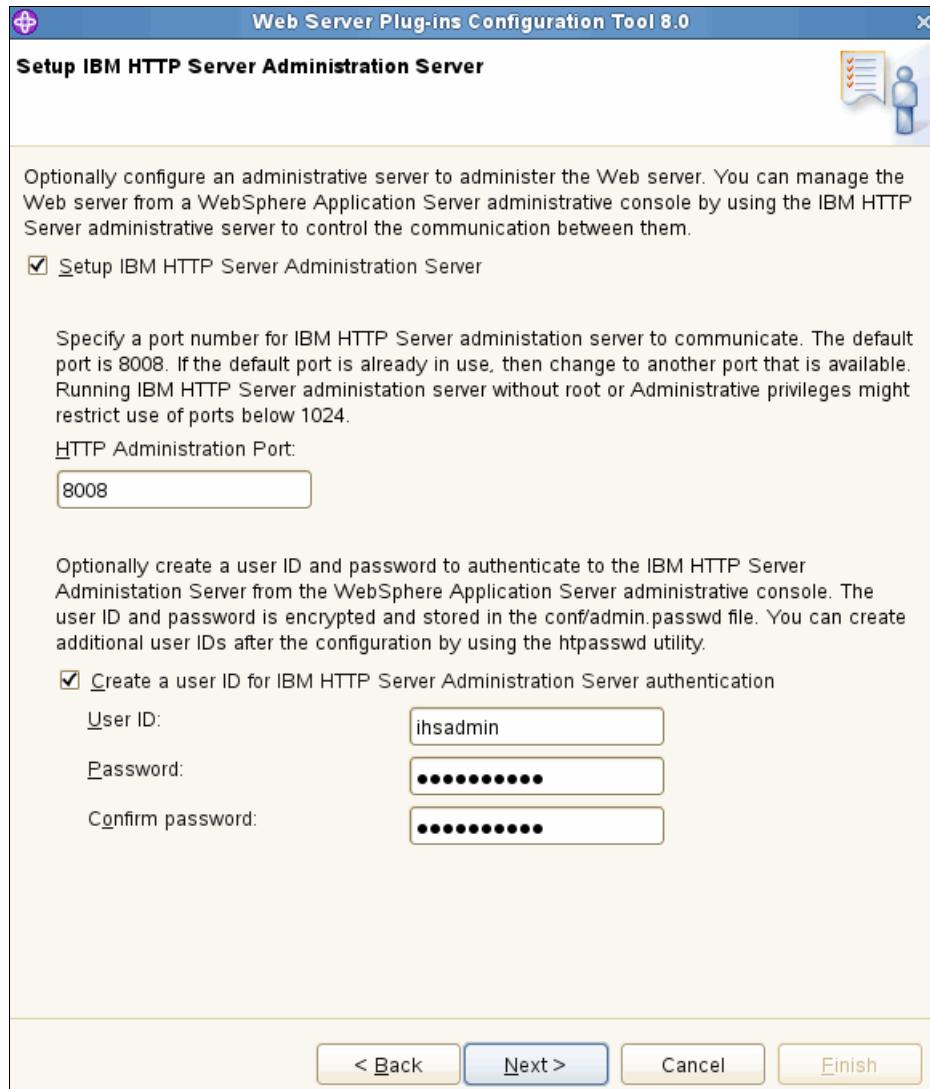


Figure 12-13 IBM HTTP Server administration server settings

Click **Next**.

- c. Provide a unique user ID and group for web server administration. In Figure 12-14, the user ID and Group is *ihs*.

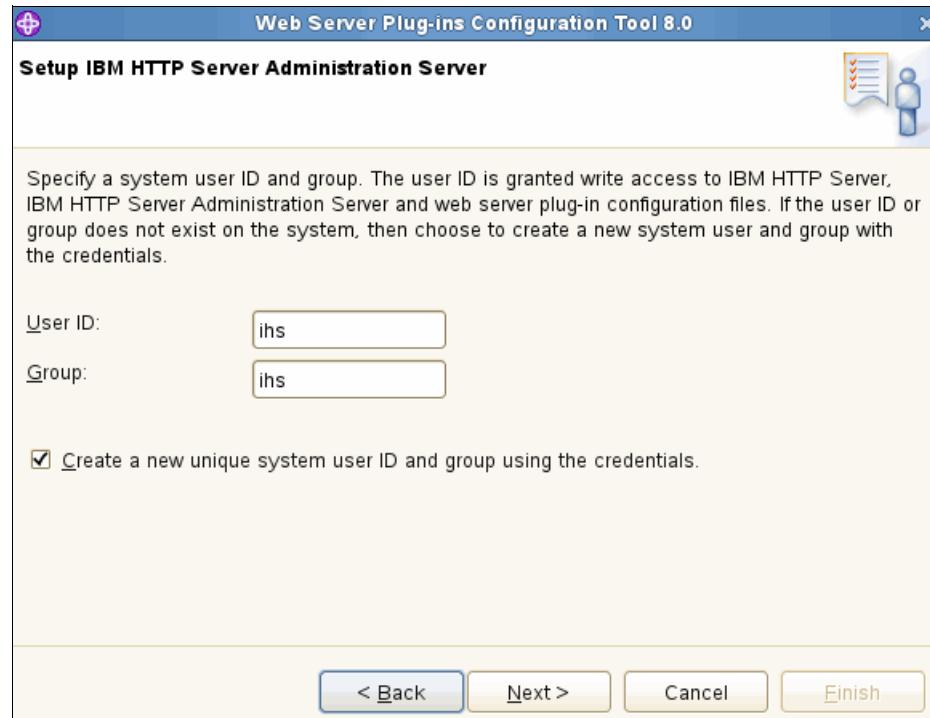


Figure 12-14 IBM HTTP Server administration server setup

Click **Next**.

8. Specify a unique name for the web server definition, as shown in Figure 12-15. Click **Next**.

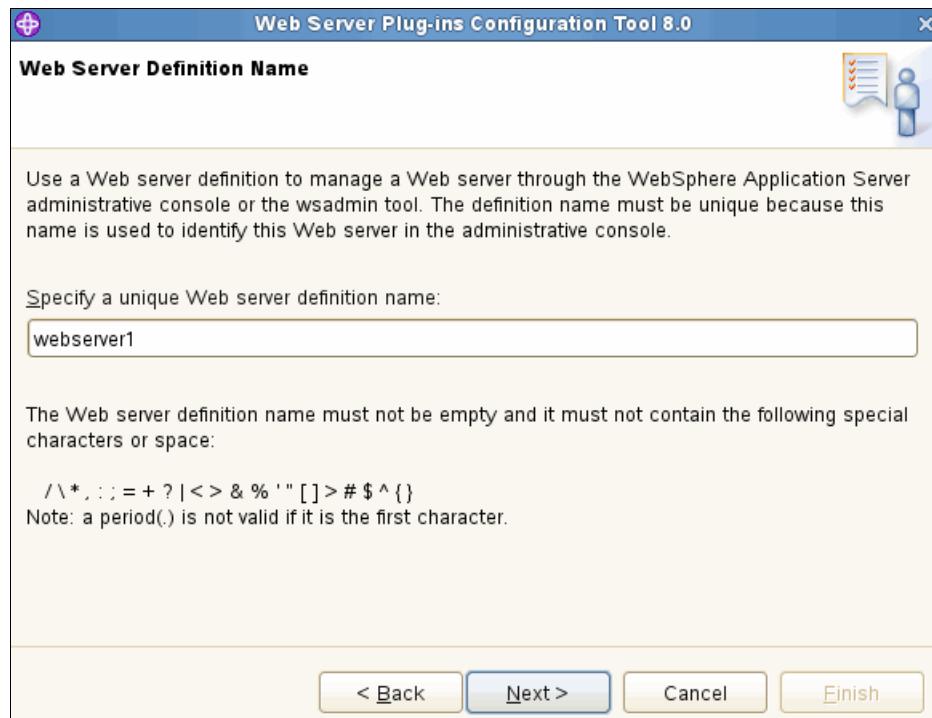


Figure 12-15 Web server definition name

9. Select the remote scenario configuration. Provide a host name or IP address of the application server or deployment manager, as shown in Figure 12-16.

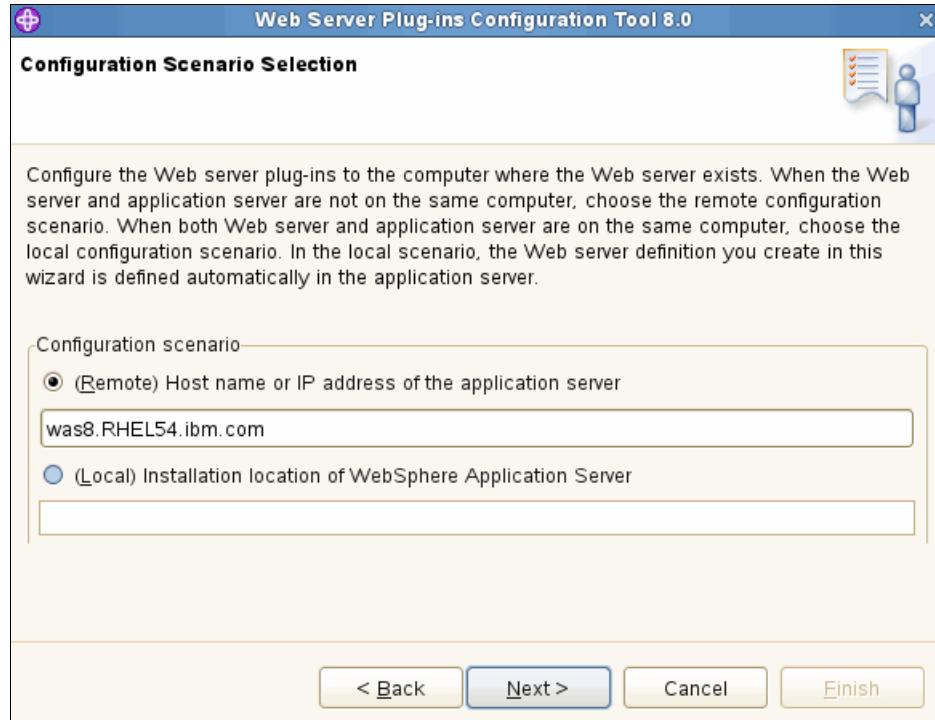


Figure 12-16 Configuration scenario selection

Click **Next**.

10. On the Summary window, review the settings and click **Configure**.

11. When the configuration is complete, clear the **Launch the plug-in configuration roadmap** check box and click **Finish**. Also note the manual configuration step that must be performed, as shown in Figure 12-17.

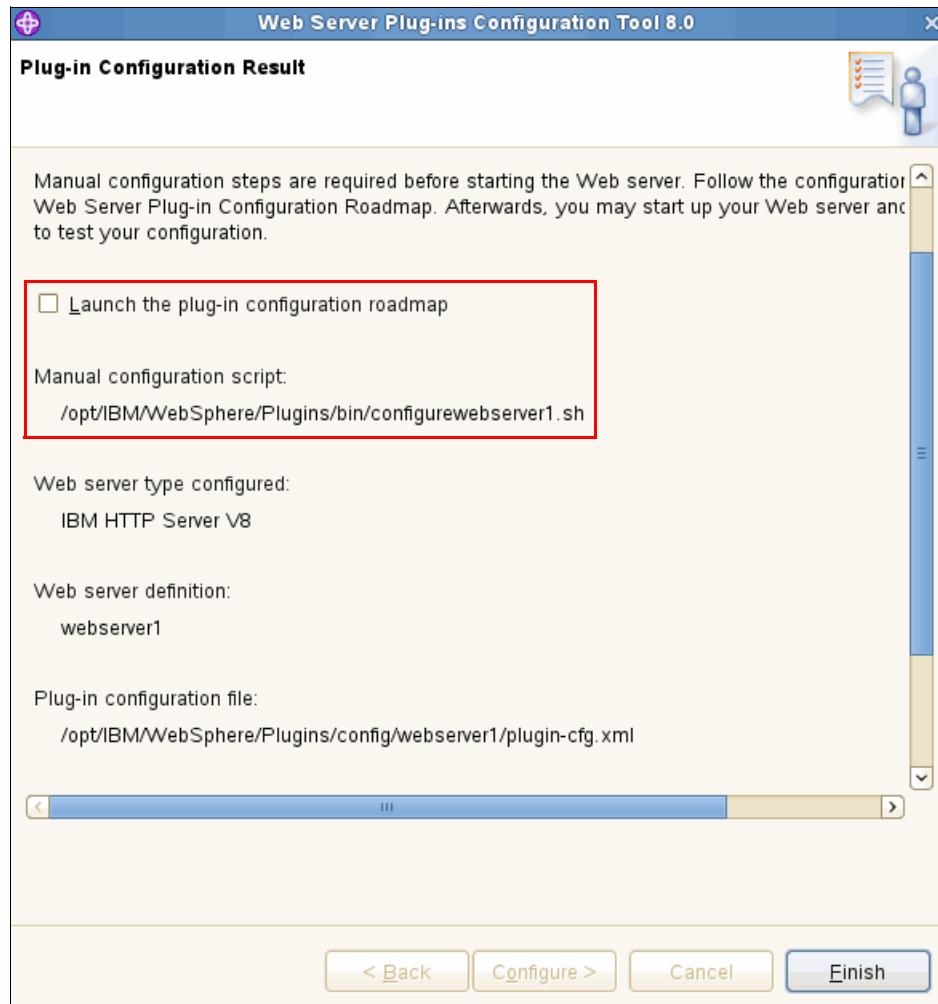


Figure 12-17 Configuration results

12. You can see the configuration file in the WebSphere Customization Toolbox. Exit the WebSphere Customization Toolbox. When the WebSphere Customization Toolbox GUI is used for the plug-in configuration, the selections made are saved and are available in a response file. See Figure 12-18.

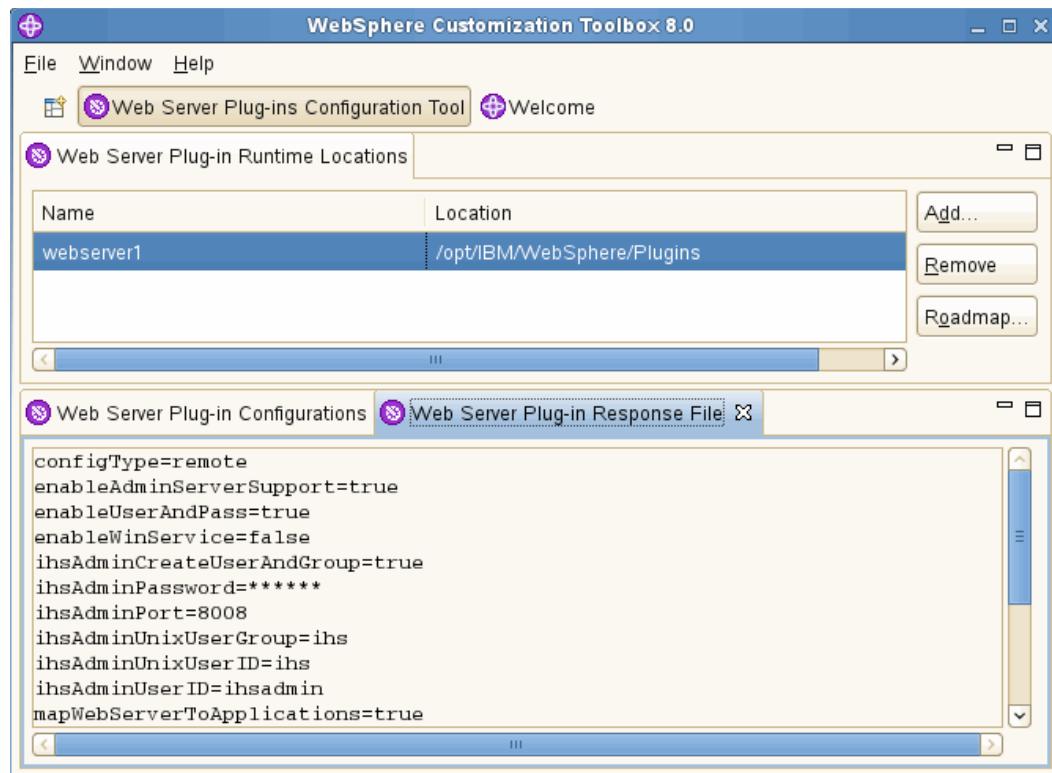


Figure 12-18 Web server plug-in response file

Alternative: An alternative to using the WebSphere Customization Toolbox GUI is the WebSphere Customization Toolbox command-line utility, which can be used with a response file to configure a web server.

For more information about the WebSphere Customization Toolbox command-line utility, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.nd.doc/info/ae/ae/tins_pctl_using.html

13. The Web Server Plug-ins Configuration Tool creates the `configureweb_server_name` script in the `plugins_root/bin/` directory on the machine with the web server. Examine the script and make any needed changes. You may need to compensate for file encoding differences to prevent script failure.

Note: The content of the `configureweb_server_name.bat` script or the `configureweb_server_name.sh` script can be corrupt if the default file encoding of the two machines differs. This scenario is possible when one machine is set up for a double-byte character set (DBCS) locale and the other machine is not.

For more information about this topic, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.nd.doc/info/ae/ae/tins_webplugins_remotesa.html

The Web Server Plug-ins Configuration Tool also creates the `plugin-cfg.xml` file in the `plugins_root/config/web_server_name` directory.

14. Copy the `configureweb_server_name` script in the `plugins_root/bin/` directory to `profile_root/Dmgr/bin`.

Note: If one platform is a system such as AIX or Linux and the other is a Windows platform, copy the script from the `plugins_root/bin/crossPlatformScripts` directory.

15. Run the `configureweb_server_name` script. If administrative security is enabled, provide the `-username` and `-password` arguments or wait until prompted for the credentials.
16. Log in to the administrative console and verify the configuration. Click **System administration** → **Nodes**. See Figure 12-19.

Nodes

Use this page to manage nodes in the application server environment. A node corresponds to a physical computer system with a distinct IP host address. The following table lists the managed and unmanaged nodes in this cell. The first node is the deployment manager. Add new nodes to the cell and to this list by clicking Add Node.

Preferences

Add Node	Remove Node	Force Delete	Synchronize	Full Resynchronize	Stop
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Select	Name	Host Name	Version	Discovery Protocol	Status
You can administer the following resources:					
<input type="checkbox"/>	ihsNode01	was8.RHEL56.ibm.com	Not applicable	TCP	<input type="checkbox"/>
<input type="checkbox"/>	was8CellManager01	was8.RHEL56.ibm.com	ND 8.0.0.0	TCP	<input type="checkbox"/>
<input type="checkbox"/>	was8Node01	was8.RHEL56.ibm.com	ND 8.0.0.0	TCP	<input type="checkbox"/>
<input type="checkbox"/>	was8Node02	was8.RHEL56.ibm.com	ND 8.0.0.0	TCP	<input type="checkbox"/>
Total 4					

Figure 12-19 Nodes listing

12.4 Working with web servers

The introduction of web server definitions to the WebSphere Application Server administrative tools provides the following administrative features:

- ▶ Defining nodes (distributed server environment)
- ▶ Defining and modifying web servers
- ▶ Checking the status of a web server
- ▶ Starting and stopping IBM HTTP Servers
- ▶ Administering IBM HTTP Servers

- ▶ Viewing or modifying the web server configuration file
- ▶ Mapping modules to servers

12.4.1 Manually defining nodes and web servers

A managed node is added to the cell as part of the process when you federate an application server profile or custom profile to the cell. An unmanaged node, however, is not created using a profile. As you have seen, the web server definition script created by the Web Server Plug-ins Configuration Tool defines an unmanaged node for a web server and the web server.

However, there might be times when you need to define or update the definitions using the administrative console.

Adding an unmanaged node to the cell

To add an unmanaged node using the administrative console, complete the following steps:

1. Click **System administration** → **Nodes** in the console navigation tree.
2. Click **Add Node**.
3. Select **Unmanaged node**. See Figure 12-20.

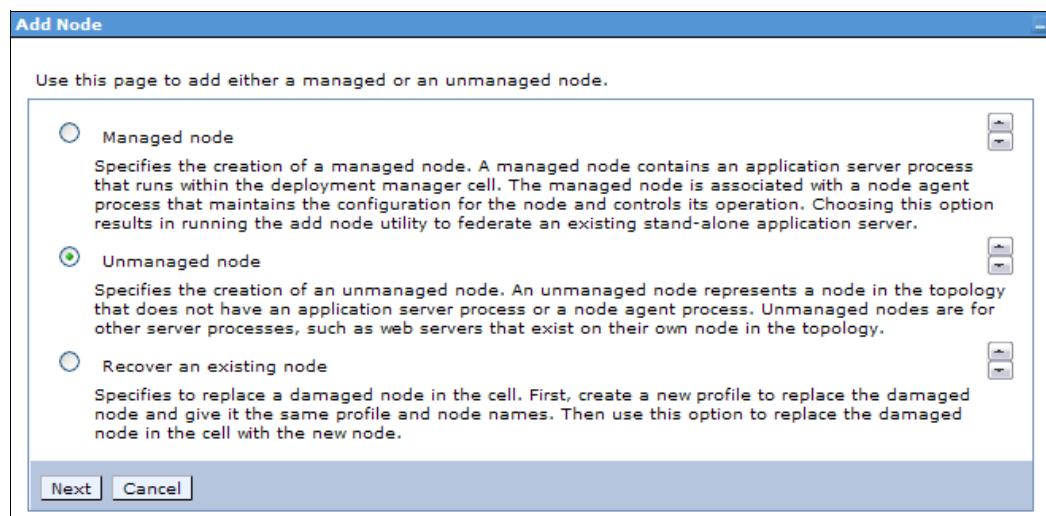


Figure 12-20 Add node selection window

4. Click **Next**.

5. Enter the following values in the General Properties window. See Figure 12-21.

a. Name

Type a logical name for the node. The name must be unique within the cell. A node name usually is identical to the host name for the computer. However, you can make the node name different than the host name.

b. Host Name

Enter the host name of the unmanaged node that is added to the configuration.

c. Platform Type

Select the operating system on which the unmanaged node runs. Valid options are:

- Windows
- AIX
- HP-UX
- Solaris
- Linux
- IBM OS/400®
- z/OS

The screenshot shows the 'Nodes > New...' dialog box. At the top, there is a brief description: 'Use this page to view or change the configuration for an unmanaged node. An unmanaged node is a node defined in the cell topology that does not have a node agent running to manage the process. Unmanaged nodes are typically used to manage web servers.' Below this, there are two tabs: 'Configuration' (which is selected) and 'Additional Properties'. The 'General Properties' section contains three fields with asterisks: 'Name' (set to 'ihsNode02'), 'Host Name' (set to 'was8.RHEL56.ibm.com'), and 'Platform Type' (set to 'Linux'). To the right of these fields, a note says: 'The additional properties will not be available until the general properties for this item are applied or saved.' Below the 'General Properties' section is a 'Custom Properties' section, which is currently empty. At the bottom of the dialog are four buttons: 'Apply', 'OK' (highlighted in yellow), 'Reset', and 'Cancel'.

Figure 12-21 General properties for an unmanaged node

6. Click **OK**. The node is added and the name is displayed in the collection on the Nodes page. See Figure 12-22.

Select	Name	Host Name	Version	Discovery Protocol	Status
You can administer the following resources:					
<input type="checkbox"/>	ihsNode01	was8.RHEL56.ibm.com	Not applicable	TCP	
<input type="checkbox"/>	ihsNode02	was8.RHEL56.ibm.com	Not applicable	TCP	
	was8CellManager01	was8.RHEL56.ibm.com	ND 8.0.0.0	TCP	
<input type="checkbox"/>	was8Node01	was8.RHEL56.ibm.com	ND 8.0.0.0	TCP	
<input type="checkbox"/>	was8Node02	was8.RHEL56.ibm.com	ND 8.0.0.0	TCP	
Total 5					

Figure 12-22 Nodes in a cell

Adding a web server

After the node for the web server has been defined, you can add the web server definition. To add a web server definition, complete the following steps:

1. Click **Servers** → **Web servers**.
2. Click **New**.
3. Select the node and enter the web server name. Using the drop-down menu, select the web server type. See Figure 12-23. Click **Next**.

Use this page to create a new web server.

→ Step 1: Select a node for the Web server and select the Web server type

Step 2: Select a Web server template

Step 3: Enter the properties for the new Web server

Step 4: Confirm new Web server

Select a node for the Web server and select the Web server type

Select a node that corresponds to the Web server you want to add.

Select node

ihsNode02

* Server name

webserver2

* Type

IBM HTTP Server

Next Cancel

Figure 12-23 Defining a web server

4. Select a template. Initially, this template will be the one supplied with WebSphere that is specific to the web server type. After you have defined a web server, you can make it a template for future use. See Figure 12-24.

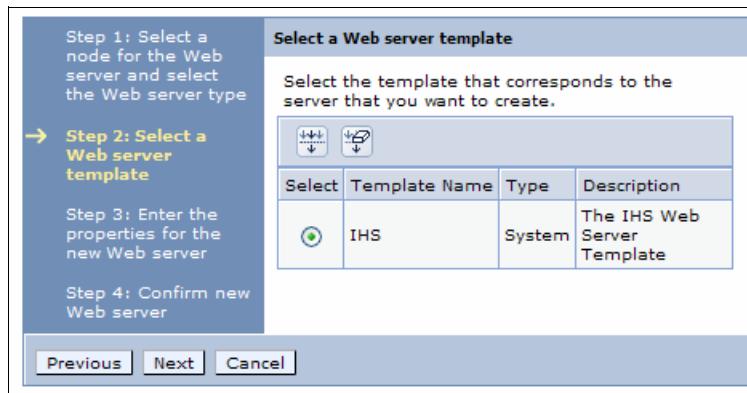


Figure 12-24 Defining a web server - Template

Click **Next**.

5. Enter the properties for the web server. You also need to enter the parameters required for remote administration. See Figure 12-25.

* Port	80
* Web server installation location	/opt/IBM/HTTPServer
* Plug-in installation location	/opt/IBM/HTTPServer/Plugins
Application mapping to the Web server	All
Enter the IBM Administration Server properties.	
* Administration Server Port	8008
* Username	ihsadmin
* Password	*****
* Confirm password	*****
<input type="checkbox"/> Use SSL	

Figure 12-25 Defining a web server - Properties

- Review the options and click **Finish**, as shown in Figure 12-26.

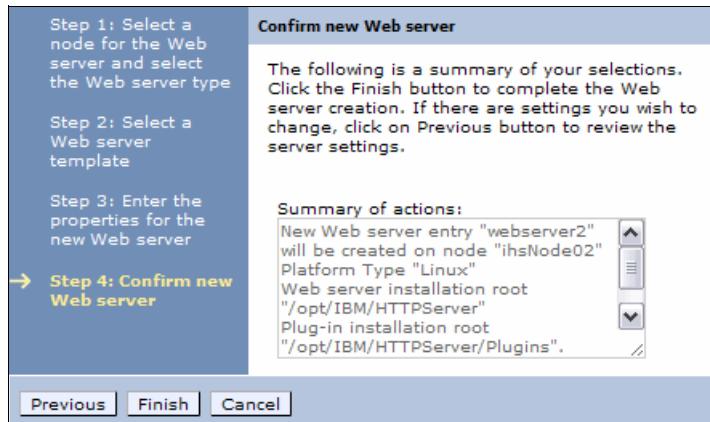


Figure 12-26 Defining a web server - Confirmation

12.4.2 Viewing the status of a web server

The web server status is reflected in the administrative console. To view web servers and their statuses, complete the following steps:

- Click **Servers** → **Web servers**. If a web server is started or stopped using a native command, you might need to refresh the view by clicking the icon to see the new status, as shown in Figure 12-27.

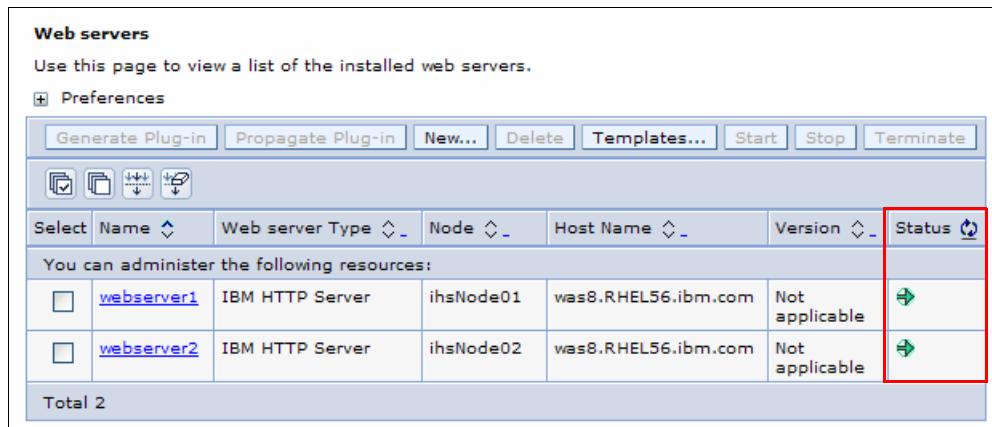


Figure 12-27 Web server status

WebSphere Application Server reports server status using the web server host name and port that you have defined. This is normally port 80. Do not use the remote administration port. If **Use secure protocol** is defined, SSL will be used. See Figure 12-28 on page 471.

12.4.3 Starting and stopping a web server

You can start or stop a web server using either the administrative console or a command window.

From the administrative console

You can start or stop the following web servers from the WebSphere administrative console:

- ▶ All web servers on a managed node
 - The node agent will be used to start or stop the web server.
- ▶ IBM HTTP Server on an unmanaged node
 - The IBM HTTP Server administration must be running on the web server node.

To start or stop a web server from the administrative console, complete the following steps:

1. Click **Servers** → **Web servers**.
2. Select the check box to the left of each web server you want to start or stop.
3. Click **Start or Stop**.

If you have problems starting or stopping an IBM HTTP Server, check the WebSphere console logs (trace) and, if using the IBM HTTP administration server, check the `admin_error.log` file.

If you have problems starting and stopping IBM HTTP Server on a managed node using the node agent, you can try to start and stop the server by setting up the managed profile and issuing the `startserver <IBM HTTP Server> -nowait -trace` command and checking the `startServer.log` file for the IBM HTTP Server specified.

From a command window

You can also use the native startup or shutdown procedures for the supported web server. From a command window, change to the directory of your IBM HTTP Server installed image, or to the installed image of a supported web server.

- ▶ To start or stop the IBM HTTP Server for UNIX platforms, enter one of the following command at a command prompt:
 - `# <ihs_install>/bin/apachectl start`
 - `# <ihs_install>/bin/apachectl stop`
- ▶ To start or stop the IBM HTTP Server on a Windows platform, select the **IBM HTTP Server 8.0** service from the Services window and invoke the appropriate action.

Note: When the web server is started or stopped with the native methods, the web server status on the web servers page of the administrative console is updated accordingly.

12.4.4 IBM HTTP Server remote administration

You can administer and configure IBM HTTP Server V8.0 using the WebSphere administrative console. On a managed node, administration is performed using the node agent. This true of all web server types. However, unlike other web servers, administration is possible for an IBM HTTP Server installed on an unmanaged node. In this case, administration is done through the IBM HTTP administration server. This server must be configured and running. Administration is limited to generation and propagation of the plug-in configuration file.

Remote administration setup

In order for the administrative console to access the IBM HTTP administration server, you must define a valid user ID and password to access the IBM HTTP Server administration server. The user ID and password are stored in the web server's IBM HTTP Server administration server properties.

You can update your IBM HTTP Server administration server properties in the web server definition through the Remote Web server management properties window of the administrative console. Complete the following steps to set or change these properties:

1. Click **Servers** → **Web servers**.
2. Select the name of the web server.
3. Click **Remote Web server management** in the Additional Properties section.
4. Enter the remote web server management information, as shown in Figure 12-28.

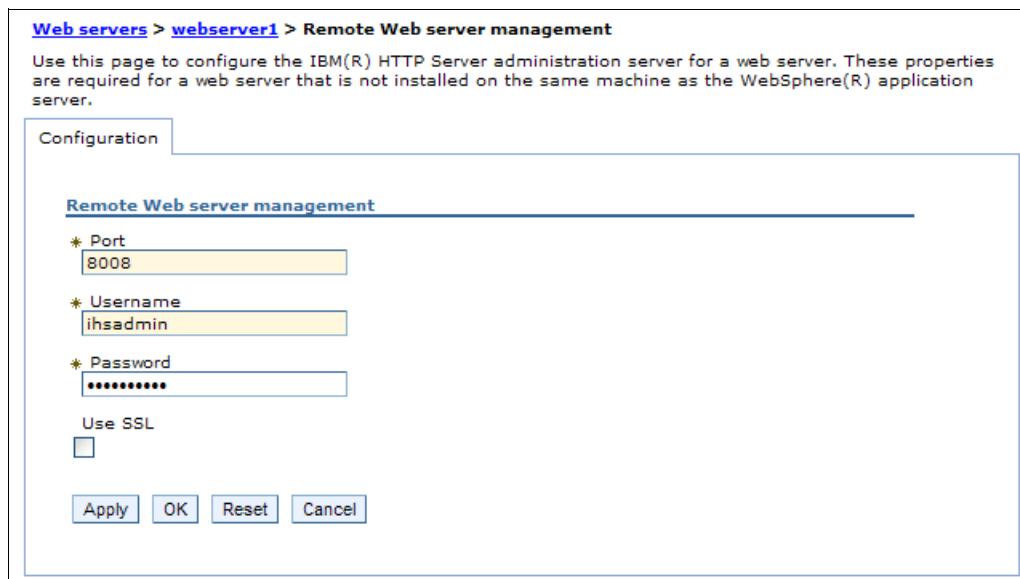


Figure 12-28 IBM HTTP Server remote management properties

- a. Enter the port number for the IBM HTTP Server administration server. The default is 8008.
- b. Enter a user ID and password that are defined to the IBM HTTP administration server. The IBM HTTP administration server user ID and password are not verified until you attempt to connect.
- c. Select the **Use SSL check** box if the port is secure. The default is not set.
5. Click **OK** and save the configuration.

Setting the user ID and password in the IBM HTTP administration server: The IBM HTTP administration server is set, by default, to refer to the following file to get the user ID and passwords to use for authentication:

```
<ihs_install>/conf/admin.passwd
```

To initialize this file with a user ID, use the **htpasswd** command. The following example initializes the file with the user ID webadmin:

```
/opt/IBM HTTP Server/bin>./htpasswd.sh /opt/IBM HTTP Server/conf/admin.passwd" webadmin
```

```
New password: *****
Re-type new password: *****
```

```
Adding password for user webadmin
```

When you are managing an IBM HTTP Server using the WebSphere administrative console, you must ensure that the following conditions are met:

- ▶ Verify that the IBM HTTP Server administration server is running.
- ▶ Verify that the web server host name and port defined in the WebSphere administrative console match the IBM HTTP Server administration host name and port.
- ▶ Verify that the firewall is not preventing you from accessing the IBM HTTP Server administration server from the WebSphere administrative console.
- ▶ Verify that the user ID and password specified in the WebSphere administrative console under Remote Web server management is an authorized combination for IBM HTTP Server administration.
- ▶ If you are trying to connect securely, verify that you have exported the IBM HTTP Server administration server keydb personal certificate into the WebSphere key database as a signer certificate. This key database will be specified by the com.ibm.ssl.trustStore in the sas.client.props file in the profile your console is running. This setup is mainly for self-signed certificates.
- ▶ Verify that the IBM HTTP Server admin_error.log file and the WebSphere Application Server logs (trace.log) do not contain any errors.

Hints and tips

The following list describes hints and tips on starting, stopping, and obtaining the status for the IBM HTTP Server using the WebSphere administrative console:

Viewing or modifying the web server configuration file

The Web Server Plug-ins Configuration Tool automatically configures the web server configuration file with the information necessary to use the plug-in. For example, among the updates made are the lines in Example 12-1 at the bottom of the httpd.conf file.

Example 12-1 Plug-in configuration location defined in httpd.conf

```
LoadModule was_ap22_module /opt/IBM/WebSphere/Plugins/bin/mod_was_ap22_http.so"
WebSpherePluginConfig /opt/IBM/WebSphere/Plugins/config/webserver1/plugin-cfg.xml"
```

Note that the location the web server expects to find the plug-in configuration file is specified in these lines. When you generate the web server plug-in configuration from the managed web server, you need to propagate or copy the generated file to this location.

The web server configuration file is a text file and can be modified or viewed manually with a text editor. You can also view or modify this file using the WebSphere Application Server administrative console.

To view or modify the contents of the web server configuration file in your web browser, complete the following steps:

1. Click **Servers** → **Web servers**.
2. Select the name of the web server.
3. Click **Configuration File** in the Additional Properties section. See Figure 12-29.

The screenshot shows a text editor window with the title bar 'Configuration file'. The main area contains the content of the httpd.conf configuration file. The file starts with a series of '#' comments explaining the file's purpose and structure. It then defines three basic sections: global environment, server parameters, and virtual hosts. A note about ServerRoot is included, followed by a warning against adding a slash at the end of the directory path.

```
# This is the main IBM HTTP server configuration file. It contains the
# configuration directives that give the server its instructions.
# See <URL: http://publib.boulder.ibm.com/httpserv/manual70/> for detailed
# information about the Apache directives.
#
# The instructions provided in this configuration file are only hints or
# reminders. Consult the online docs for definitive information.
#
# The configuration directives are grouped into three basic sections:
# 1. Directives that control the operation of the web server process as a
#    whole (the 'global environment').
# 2. Directives that define the parameters of the 'main' or 'default' server,
#    which responds to requests that aren't handled by a virtual host.
#    These directives also provide default values for the settings
#    of all virtual hosts.
# 3. Settings for virtual hosts, which allow Web requests to be sent to
#    different IP addresses or hostnames and have them handled by the
#    same web server process.
#
# Configuration and logfile names: If the filenames you specify for many
# of the server's control files begin with "/" (or "drive:/\" for Win32), the
# server will use that explicit path. If the filenames do *not* begin
# with "/", the value of ServerRoot is prepended -- so "logs/foo.log"
# with ServerRoot set to "/home/IBM/HTTPServer" will be interpreted by the
# server as "/home/IBM/HTTPServer/logs/foo.log".
#
##### Section 1: Global Environment
#
# The directives in this section affect the overall operation of IBM HTTP
# Server, such as the number of concurrent requests it can handle or where
# it can find its configuration files.
#
#
# ServerRoot: The top of the directory tree under which the server's
# configuration, error, and log files are kept.
#
# Do NOT add a slash at the end of the directory path.
#
```

Figure 12-29 IBM HTTP Server configuration file httpd.conf

4. Type your changes directly into the window and click **OK**. Save the changes.

Note: If you made changes to the configuration file, you need to restart your web server for the changes to take effect.

Viewing web server logs

With remote administration, you can also view the IBM HTTP Server access log and error log. To view the logs, complete the following steps:

1. Click **Servers** → **Web servers**.
2. Select the name of the web server.
3. Click **Log file** in the Additional Properties section.

- Click the **Runtime** tab. See Figure 12-30.

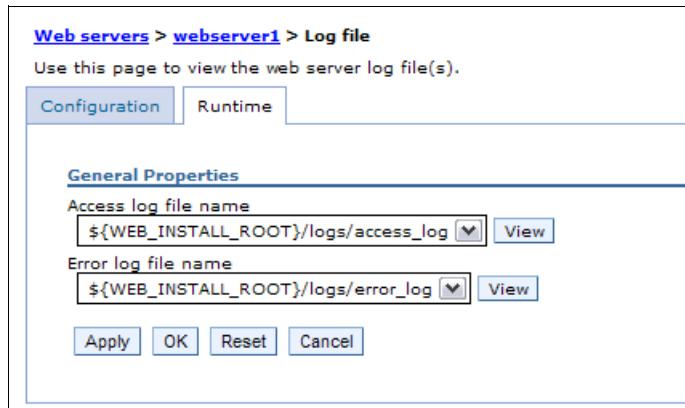


Figure 12-30 Web server Runtime tab for logs

- Click **View** beside the log you want to view. See Figure 12-31.



Figure 12-31 Viewing the error log

12.4.5 Mapping modules to servers

Each module of an application is mapped to one or more target servers. The target server can be an application server, cluster of application servers, or a web server. Modules can be installed on the same application server or dispersed among several application servers. Web servers specified as targets will have routing information for the application generated in the plug-in configuration file for the web server.

This mapping takes place during application deployment. After an application is deployed, you can view or change these mappings. To check or change the mappings, complete the following steps:

- Click **Applications** → **Application Types** → **WebSphere enterprise applications**.
- Click the application for which you want to review the mapping.

3. Click **Manage modules** in the Modules section.
4. The Selecting servers window opens, as shown in Figure 12-32. Examine the list of mappings. Ensure that each Module entry is mapped to all targets identified under Server.

Enterprise Applications > DefaultApplication.ear > Manage Modules

Manage Modules

Specify targets such as application servers or clusters of application servers where you want to install the modules that are contained in your application. Modules can be installed on the same application server or dispersed among several application servers. Also, specify the Web servers as targets that serve as routers for requests to this application. The plug-in configuration file (plugin-cfg.xml) for each Web server is generated, based on the applications that are routed through.

Clusters and servers:

WebSphere:cell=was8Cell01,node=was8Node01,server=server1	WebSphere:cell=was8Cell01,node=was8Node02,server=server2
WebSphere:cell=was8Cell01,node=ihsNode01,server=webserver1	WebSphere:cell=was8Cell01,node=ihsNode02,server=webserver2

Buttons: Remove, Update, Remove File, Export File

Select	Module	URI	Module Type	Server
<input type="checkbox"/>	Increment EJB module	Increment.jar, META-INF/ejb-jar.xml	EJB Module	WebSphere:cell=was8Cell01,node=ihsNode01,server=webserver1 WebSphere:cell=was8Cell01,node=was8Node01,server=server1
<input type="checkbox"/>	Default Web Application	DefaultWebApplication.war, WEB-INF/web.xml	Web Module	WebSphere:cell=was8Cell01,node=ihsNode01,server=webserver1 WebSphere:cell=was8Cell01,node=was8Node01,server=server1

Buttons: OK, Cancel

Figure 12-32 Application module mappings

5. To change a mapping, complete the following steps:
 - a. Select each module that you want mapped to the same targets by placing a check mark in the box to the left of the module.
 - b. From the Clusters and Servers list, select one or more targets. Use the Ctrl key to select multiple targets. For example, to have a web server serve your application, use the Ctrl key to select an application server and the web server together.
6. Click **Apply**.
7. Repeat step 5 until each module maps to the desired targets.
8. Click **OK** and save your changes.
9. Regenerate and propagate the plug-in configuration, if it is not done automatically.

After you have defined at least one web server, you must specify a web server as a deployment target whenever you deploy a web application. If the web server plug-in configuration service is enabled, a web server plug-in's configuration file is automatically regenerated whenever a new application is associated with that web server.

12.5 Working with the plug-in configuration file

The plug-in configuration file (`plugin-cfg.xml`) contains routing information for all applications mapped to the web server. This file is read by a binary plug-in module loaded in the web server. An example of a binary plug-in module is the `mod_ibm_app_server_http.dll` file for IBM HTTP Server on the Windows platform.

The binary plug-in module does not change. However, the plug-in configuration file for the binary module needs to be regenerated and propagated to the web server whenever a change is made to the configuration of applications mapped to the web server. The binary module reads the XML file to adjust settings and to locate deployed applications for the web server.

Example 12-2 shows an excerpt from a generated plug-in configuration file.

Example 12-2 An excerpt from the plugin-cfg.xml

```
<?xml version="1.0" encoding="ISO-8859-1"?><!--HTTP server plugin config file for  
the webserver ITSOCell.wan.webserver1 generated on 2004.10.29 at 03:32:12 PM  
BST-->  
<Config ASDisableNagle="false" AcceptAllContent="false"  
AppServerPortPreference="HostHeader" ChunkedResponse="false" FIPSEnable="false"  
FailoverToNext="false" HTTPMaxHeaders="300" IISDisableNagle="false"  
IISPluginPriority="High" IgnoreDNSFailures="false"  
OS400ConvertQueryStringToJobsCCSID="false" RefreshInterval="60"  
ResponseChunkSize="64" SSLConsolidate="true" TrustedProxyEnable="false"  
VHostMatchingCompat="false">  
    <Log LogLevel="Error"  
Name="c:\opt\WebSphere\Plugins\logs\webserver1\http_plugin.log"/>  
    <Property Name="ESIEnable" Value="true"/>  
    <Property Name="ESIMaxCacheSize" Value="1024"/>  
    <Property Name="ESIInvalidationMonitor" Value="false"/>  
    <Property Name="ESIEnableToPassCookies" Value="false"/>  
    <Property Name="PluginInstallRoot" Value="c:\opt\WebSphere\Plugins\*"/>  
<VirtualHostGroup Name="default_host">  
    <VirtualHost Name="*:9080"/>  
    <VirtualHost Name="*:80"/>  
    <VirtualHost Name="*:9443"/>  
</VirtualHostGroup>  
  
    <ServerCluster CloneSeparatorChange="false" GetDWLMTable="false"  
IgnoreAffinityRequests="true" LoadBalance="Round Robin"  
Name="server1_NodeA_Cluster" PostBufferSize="64" PostSizeLimit="-1"  
RemoveSpecialHeaders="true" RetryInterval="60">  
        <Server ConnectTimeout="0" ExtendedHandshake="false" MaxConnections="-1"  
Name="NodeA_server1" WaitForContinue="false">  
            <Transport Hostname="wan" Port="9080" Protocol="http"/>  
            <Transport Hostname="wan" Port="9443" Protocol="https">  
                <Property Name="keyring"  
Value="c:\opt\WebSphere\Plugins\etc\plugin-key.kdb"/>  
                <Property Name="stashfile"  
Value="c:\opt\WebSphere\Plugins\etc\plugin-key.sth"/>  
            </Transport>  
        </Server>  
</ServerCluster>
```

```

<UriGroup Name="default_host_server1_NodeA_Cluster_URIs">
    <Uri AffinityCookie="JSESSIONID" AffinityURLIdentifier="jsessionid"
Name="/snoop/*"/>
    <Uri AffinityCookie="JSESSIONID" AffinityURLIdentifier="jsessionid"
Name="/hello"/>

</UriGroup>
<Route ServerCluster="server1_NodeA_Cluster"
UriGroup="default_host_server1_NodeA_Cluster_URIs"
VirtualHostGroup="default_host"/>
</Config>

```

The specific values for the UriGroup Name and AffinityCookie attributes depend on how you have assembled your application. When you assemble your application:

- ▶ If you specify **File Serving Enabled**, then only a wildcard URI is generated, regardless of any explicit servlet mappings.
- ▶ If you specify **Serve servlets by class name**, then a URI of the form URI name = <webappuri>/servlet/ is generated.

Both these options apply for both the Name and AffinityCookie attributes.

When the plug-in configuration file is generated, it does not include admin_host in the list of virtual hosts. See “Allowing Web servers to access the administrative console” in the Information Center for information about how to add it to the list at the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.nd.doc/info/ae/ae/tins_configACAccess.html

12.5.1 Regenerating the plug-in configuration file

The plug-in configuration file needs to be regenerated and propagated to the web servers when there are changes to your WebSphere configuration that affect how requests are routed from the web server to the application server. These changes include:

- ▶ Installing an application
- ▶ Creating or changing a virtual host
- ▶ Creating a new server
- ▶ Modifying HTTP transport settings
- ▶ Creating or altering a cluster

The plug-in file can be regenerated manually using the administration tools. You can also set up the plug-in properties of the web server to enable automatic generation of the file whenever a relevant configuration change is made. Refer to “Enabling automated plug-in regeneration” on page 482 for more information.

To regenerate the plug-in configuration manually, you can either use the administrative console, or you can issue the **GetPluginCfg** command.

Generating the plug-in with administrative console

To generate or regenerate the plug-in configuration file, complete the following steps:

1. Click **Servers** → **Web servers**.
2. Select the check box to the left of your web server.
3. Click **Generate Plug-in**.

4. Verify that the generation was successful by looking at the messages. A success message will be accompanied with the location of the generated plug-in configuration file:
`<profile_home>/config/cells/<cell_name>/nodes/<web_server_node>/servers/<web_server>/plugin-cfg.xml`

See Figure 12-33.

Select	Name	Web server Type	Node	Host Name	Version	Status
<input type="checkbox"/>	webserver1	IBM HTTP Server	ihsNode01	was8.RHEL56.ibm.com	Not applicable	
<input type="checkbox"/>	webserver2	IBM HTTP Server	ihsNode02	was8.RHEL56.ibm.com	Not applicable	

Figure 12-33 Web server definitions

- Click the name of the web server. You can view the plug-in configuration file by clicking the **View** button next to the Plug-in configuration file name on the Plug-in properties window of your web server definition, as shown in Figure 12-34. You can also open it with a text editor.

Figure 12-34 Plug-in properties

To use the new `plugin-cfg.xml` file, you must propagate it to the web server system. See 12.5.2, “Propagating the plug-in configuration file” on page 483 for more information.

Regenerating the plug-in with the `GenPluginCfg` command

The `GenPluginCfg` command is used to regenerate the plug-in configuration file. Depending on the operating platform, the command is:

- ▶ On UNIX: `GenPluginCfg.sh`
- ▶ On Windows: `GenPluginCfg.bat`

You can use the `-profileName` option to define the profile of the Application Server process in a multi-profile installation. The `-profileName` option is not required for running in a single profile environment. The default for this option is the default profile. For a distributed server environment, the default profile is the deployment manager profile.

Syntax

The `GenPluginCfg` command reads the contents of the configuration repository on the local node to generate the Web server plug-in configuration file.

The syntax of the `GenPluginCfg` command is as follows:

```
:GenPluginCfg.bat(sh) [options]
```

All options are optional. The options are listed in Table 12-1.

Table 12-1 Options for GenPluginCfg

Option	Description
<code>-config.root <config root></code>	Specifies the directory path of the particular configuration repository to be scanned. The default is the value of CONFIG_ROOT defined in the <code>SetupCmdLine.bat(sh)</code> script.
<code>-profileName <profile></code>	Runs the command against this profile. If the command is run from <code><was_home>/bin</code> and <code>-profileName</code> is not specified, the default profile is used. If it is run from <code><profile_home>/bin</code> , that profile is used.
<code>-cell.name <cell name></code>	Restricts generation to only the named cell in the configuration repository. The default is the value of WAS_CELL defined in the <code>SetupCmdLine.bat(sh)</code> script.
<code>-node.name <node name></code>	Restricts generation to only the named node in the particular cell of the configuration repository. The default is the value of WAS_NODE defined in the <code>SetupCmdLine.bat(sh)</code> script.
<code>-webserver.name <webserver1></code>	Required for creating plug-in configuration file for a given web server.
<code>-propagate yes/no</code>	This option applies only when the option <code>webserver.name</code> is specified. The default is no.
<code>-propagateKeyring yes/no</code>	This option applies only when the option <code>webserver.name</code> is specified. The default is no.
<code>-cluster.name <cluster_name,cluster_name> ALL</code>	Generates an optional list of clusters. It is ignored when the option <code>webserver.name</code> is specified.
<code>-server.name <server_name,server_name></code>	Generates an optional list of servers. It is required for single server plug-in generation. It is ignored when the option <code>webserver.name</code> is specified.
<code>-output.file.name <filename></code>	Defines the path to the generated plug-in configuration file. The default is <code><configroot_dir>/plugin-cfg.xml</code> file. It is ignored when the option <code>webserver.name</code> is specified.
<code>-destination.root <root></code>	Specifies the installation root of the machine on which the configuration is used. It is ignored when the option <code>webserver.name</code> is specified.
<code>-destination.operating.system windows/unix</code>	Specifies the operating system of the machine on which the configuration is used. It is ignored when the option <code>webserver.name</code> is specified.

Option	Description
-force yes	An optional argument to overwrite the existing configuration file. The default is no.
-debug <yes no>	Enables or disables the output of debugging messages. The default is no (debugging is disabled).
-help or -?	Prints the command syntax.

Examples

To generate a plug-in configuration for all of the clusters in a cell, run:

```
GenPluginCfg -cell.name NetworkDeploymentCell
```

To generate a plug-in configuration for a single server, run:

```
GenPluginCfg -cell.name BaseApplicationServerCell -node.name appServerNode
-server.name appServerName
```

To generate a plug-in configuration file for a web server, run:

```
GenPluginCfg -cell.name BaseApplicationServerCell -node.name webserverNode
-webserver.name webserverName
```

When this command is issued without the option **-webserver.name webserverName**, the plug-in configuration file is generated based on topology.

Enabling automated plug-in regeneration

The web server plug-in configuration service by default regenerates the plugin-cfg.xml file automatically. You can view or change the configuration settings for the web server plug-in configuration service. See Figure 12-33 on page 478. To view or change the plug-in generation property, complete the following steps:

1. Click **Servers** → **Web servers**.
2. Click the name of your web server.
3. Select **Plug-in properties** in the Additional Properties section.
4. View or change the **Automatically generate the plug-in configuration file** option, as shown in Figure 12-35.

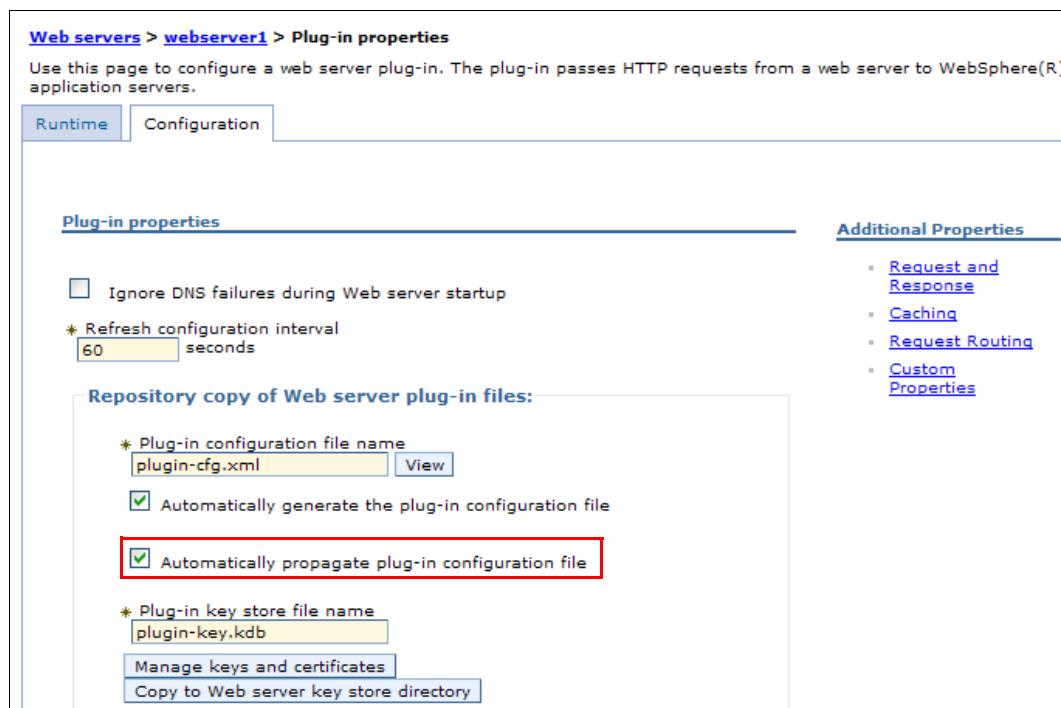


Figure 12-35 Plug-in properties

When selected, the web server plug-in configuration service automatically generates the plug-in configuration file whenever the web server environment changes. For example, the plug-in configuration file is regenerated whenever one of the following activities occurs:

- A new application is deployed on an associated application server.
- The web server definition is saved.
- An application is removed from an associated application server.
- A new virtual host is defined.

Whenever a virtual host definition is updated, the plug-in configuration file is automatically regenerated for all of the web servers.

By default, this option is automatically selected. If you clear the check from the box, you must manually generate a plug-in configuration file for this web server.

12.5.2 Propagating the plug-in configuration file

After a plug-in configuration file is regenerated, it needs to be propagated to the web server.

The configuration service can automatically propagate the plugin-cfg.xml file to a web server machine if it is configured on a managed node, and to an IBM HTTP Server if it is configured on an unmanaged node. For other scenarios, you must manually copy the file to the web server machines.

You can manually propagate the file by copying it from the application server machine to the web server machine, or you can do it from the administrative console.

From a command window

To copy the file from one machine to another machine, complete the following steps:

1. Copy the file from its original location, for example:

```
<profile_home>/config/cells/<cell_name>/nodes/<web_server_node>/servers/<web_se  
rver>/plugin-cfg.xml
```

2. Place the copy in this directory on the remote web server machine:

```
<plug-ins_home>/config/<web_server>
```

From the administrative console

To propagate the plug-in configuration manually from the administrative console, complete the following steps:

1. Click **Servers** → **Web servers**.
2. Select the check box to the left of your web server.
3. Click **Propagate plug-in**.
4. Verify that the propagation was successful by looking at the messages.

If you are in doubt, check whether the plug-in configuration file has been propagated to the web server plug-in location by viewing it.

Activating the new plug-in configuration

The web server binary plug-in module checks for a new configuration file every 60 seconds. You can wait for the plug-in to find the changes, or you can restart the web sever to invoke the changes immediately.

Tip: If you encounter problems restarting your web server, check the http_plugin.log file in <plug-ins_home>/config/<web_server> for information about what portion of the plugin-cfg.xml file contains an error. The log file states the line number on which the error occurred along with other details that might help you diagnose why the web server did not start.

Enable automated plug-in propagation

The web server plug-in configuration service, by default, propagates the plugin-cfg.xml file automatically. To view or change the plug-in propagation property, complete the following steps. See Figure 12-35 on page 482 for further information.

1. Click **Servers** → **Web servers**.
2. Click the name of your web server.
3. Click **Plug-in properties** in the Additional Properties sub section.

4. View or change the **Automatically propagate plug-in configuration file** option.

By default, this option is automatically selected. If you clear the check from the check box, you must manually propagate the plug-in configuration file for this web server.

To verify the automatic propagation was successful, look in the SystemOut.log file of the deployment manager for details.

12.5.3 Modifying the plug-in request routing options

You can specify the load balancing option that the plug-in uses when sending requests to the various application servers associated with that web server.

To view or modify the Request routing, complete the following steps:

1. Click **Servers** → **Web Servers**.
2. Click the name of your web server.
3. Click **Plug-in properties** in the Additional Properties section.
4. Click **Request Routing** in the Additional Properties section, as shown in Figure 12-36.

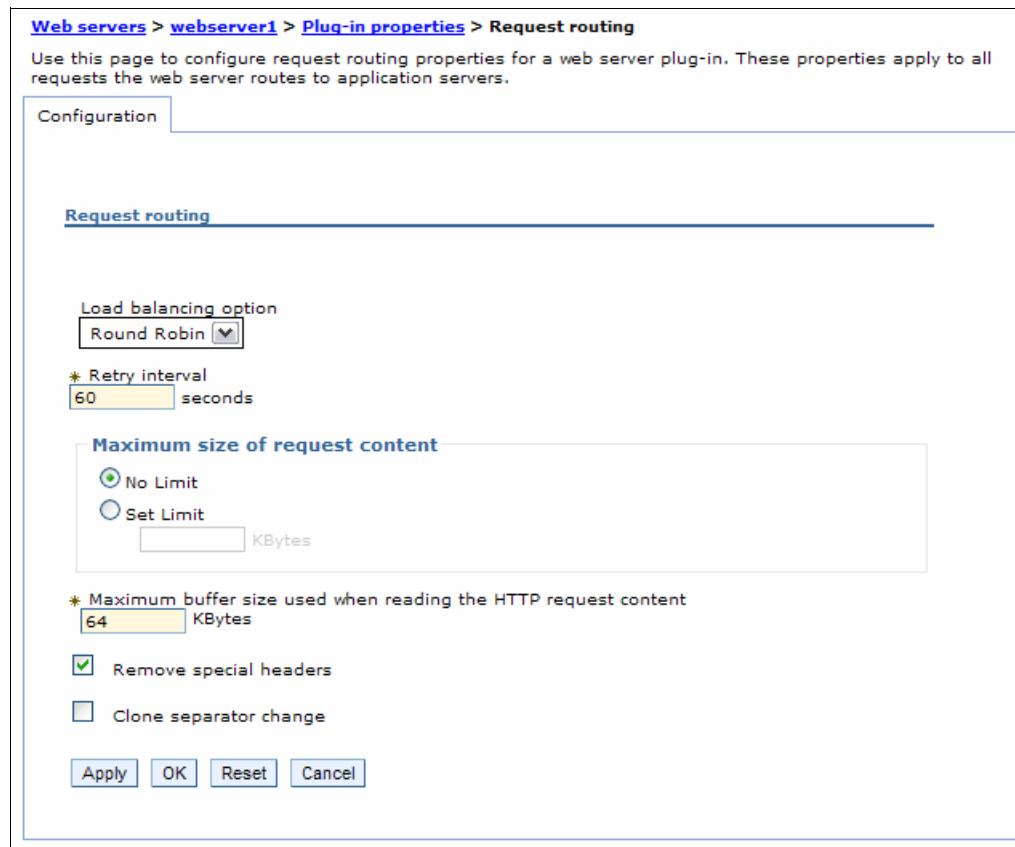


Figure 12-36 Request routing properties

a. Load balancing option

This field corresponds to the LoadBalanceWeight element in the plugin-cfg.xml file. The load balancing options are covered in detail in *WebSphere Application Server V6 Scalability and Performance Handbook*, SG24-6392. The following items are short overviews:

i. Round robin (default)

When using this algorithm, the plug-in selects a cluster member at random from which to start. The first successful browser request will be routed to this cluster member and then its weight is decremented by one. New browser requests are then sent round robin to the other application servers and, subsequently, the weight for each application server is decremented by one. The spreading of the load is equal between application servers until one application server reaches a weight of zero. From then on, only application servers without a weight higher than zero will receive routed requests. The only exception to this pattern is when a cluster member is added or restarted.

ii. Random

Requests are passed to cluster members randomly. Weights are not taken into account as in the round robin algorithm. The only time the application servers are not chosen randomly is when there are requests with associated sessions. When the random setting is used, cluster member selection does not take into account where the last request was handled, which means that a new request could be handled by the same cluster member as the last request.

b. Retry interval

The length of time, in seconds, that should elapse from the time an application server is marked down to the time that the plug-in retries a connection.

This field corresponds to the ServerWaitforContinue element in the plugin-cfg.xml file. The default is 60 seconds.

c. Maximum size of request content

Limits the size of request content. If limited, this field also specifies the maximum number of bytes of request content allowed in order for the plug-in to attempt to send the request to an application server.

This field corresponds to the PostSizeLimit element in the plugin-cfg.xml file. When a limit is set, the plug-in fails any request that is received that is greater than the specified limit.

You can set a limit in kilobytes or no limit. The default is set to no limit for the post size.

d. Maximum buffer size used when reading HTTP request content

The maximum buffer size in kilobytes that is used when the content of an HTTP request is read. If the application server that initially receives a request cannot process that request, the data contained in this buffer is sent to another application server in an attempt to have that application server process the request.

This field corresponds to the PostBufferSize element in the plugin-cfg.xml file. The default is 64 KB.

e. Remove special headers

When enabled, the plug-in will remove any headers from incoming requests before adding the headers the plug-in is supposed to add before forwarding the request to an application server.

This field corresponds to the RemoveSpecialHeaders element in the plugin-cfg.xml file. The plug-in adds special headers to the request before it is forwarded to the application server. These headers store information about the request that will need to be used by the application. Not removing the headers from incoming requests introduces a potential security exposure.

The default is to remove special headers.

f. Clone separator change

When enabled, the plug-in expects the plus character (+) as the clone separator.

This field corresponds to the ServerCloneID element in the plugin-cfg.xml file. Some pervasive devices cannot handle the colon character (:) used to separate clone IDs in conjunction with session affinity. If this field is checked, you must also change the configurations of the associated application servers so that the application servers separate clone IDs with the plus character as well.

12.6 IBM HTTP Server and Web Server Plug-ins for IBM WebSphere Application Server for z/OS

The installation process for IBM HTTP Server and Web Server Plug-ins for IBM WebSphere Application Server for z/OS can be accomplished by running some jobs and setting configurations on the operating system. In this section, we demonstrate how it can be done.

12.6.1 IBM HTTP Server

Before installing IBM HTTP Server for z/OS, you need the following items:

- ▶ Installation Manager

If you do not have Installation Manager installed and operational, refer to 4.2, “IBM Installation Manager installation” on page 143.

- ▶ Product repository

If you do not have your repository available, refer to 4.3.4, “Installing the WebSphere Application Server initial repository” on page 147.

The installation process for IBM HTTP Server is similar to the one used to install WebSphere Application Server, as described in 4.5.1, “Installing using the command line” on page 151. You can find more details at the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=%2Fcom.ibm.websphere.ih.s.doc%2Finfo%2Fihs%2Fihs%2Ftihs_installation_zos_installing.html

After installing IBM HTTP Server for z/OS, you need to configure one instance. For more details, check the Information Center at the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=%2Fcom.ibm.websphere.ih.s.doc%2Finfo%2Fihs%2Fihs%2Ftihs_installihsz.html

Note: If you intend to configure IBM HTTP Server to use port values under 1024, consider reserving the ports using the PORT statement and associating the port to the task name.

When you run the IBM HTTP Server task, several steps will run and numbers will be added to the task names to identify them, so when reserving ports in TCPPARMS, specify your task name followed by an asterisk to guarantee that the web server will be able to bind to the ports.

For more information, refer to *z/OS V1R12 Communications Server IP Configuration Guide* at the following website:

<http://publib.boulder.ibm.com/infocenter/zos/v1r12/index.jsp?topic=/com.ibm.zos.r12.halz002/portaccctrl.htm>

12.6.2 Web Server Plug-ins for IBM WebSphere Application Server for z/OS

Before installing Web Server Plug-Ins for IBM WebSphere Application Server for z/OS, you need the following items:

- ▶ Installation Manager

If you do not have Installation Manager installed and operational, refer to 4.2, “IBM Installation Manager installation” on page 143 for more information.

- ▶ Product repository

If you do not have your repository available, refer to 4.3.4, “Installing the WebSphere Application Server initial repository” on page 147 for more information.

The installation process for Web Server Plug-Ins for IBM WebSphere Application Server for z/OS is similar to the one used to install WebSphere Application Server, as described in 4.5.1, “Installing using the command line” on page 151. You can find more details at the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=%2Fcom.ibm.websphere.installation.zseries.doc%2Finfo%2Fzseries%2Fae%2Ftins_installation_zos_installing_plugins.html

Complete the following steps to configure Web Server Plug-Ins for IBM WebSphere Application Server for z/OS:

1. Create a configuration directory for plug-in, for example, /etc/websrv1.
2. Go to the plug-in installation root directory.

3. Create a runtime location directory for plug-in, which contains configuration information to be used by the plug-in; when running under a web server instance, use the `install_plugin.sh` script. Example 12-3 shows a sample execution where the following parameters were used:

<code>-pluginInstallLocation</code>	The directory where the plug-in code is installed
<code>-pluginRuntimeLocation</code>	The directory where the plug-in configuration will be created.
<code>-wasInstallLocation</code>	The directory where the WebSphere Application Server code is installed.

Example 12-3 Creating the plug-in configuration directory

```
/opt/zWebSphere_Plugins/V8R0/bin: >./install_plugin.sh -pluginInstallLocation  
/opt/zWebSphere_Plugins/V8R0 -pluginRuntimeLocation /etc/websrv1/Plugins  
-wasInstallLocation /opt/zWebSphere/V8R0  
Using the plug-in install location /opt/zWebSphere_Plugins/V8R0.  
Using the plug-in runtime location /etc/websrv1/Plugins.  
Using the WAS install location /opt/zWebSphere/V8R0.
```

```
#=====#
```

Show the `install_plugin.sh` cmdline args

```
Plugin Install Location=/opt/zWebSphere_Plugins/V8R0  
Plugin Runtime Location=/etc/websrv1/Plugins  
WAS Install Location =/opt/zWebSphere/V8R0
```

```
#=====#
```

```
#=====#
```

The `install_plugin.sh` has finished "/etc/websrv1/Plugins" is the Plugin runtime location

```
#=====#
```

4. Go to the `bin` subdirectory from the directory you created in step 3 or from the plug-in installation directory.

- Configure the web server instance to use the plug-in by running the `ConfigureIHSPlugin.sh` script. Example 12-4 shows a sample execution where the following parameters were used:

-plugin.home	The directory where the plug-in code is installed.
-plugin.config.xml	The location where the plug-in configuration file will be created.
-ihs.conf.file	The location of the IBM HTTP Server config file.
-operating.system	The operating system where the configuration is being performed.
-WAS.webserver.name	The web server name that will be defined in the WebSphere Application Server configuration.
-WAS.host.name	The WebSphere Application Server host name or IP address.

Example 12-4 Configuring a web server instance to use the plug-in

```
/SYSTEM/etc/websrv1/Plugins/bin: >./ConfigureIHSPlugin.sh -plugin.home
/etc/websrv1/Plugins -plugin.config.xml
/etc/websrv1/Plugins/config/webserver1/plugin-cfg.xml -ihs.conf.file
/etc/websrv1/conf/httpd.conf -operating.system ZOS -WAS.webserver.name
webserver1 -WAS.host.name WTSC58.ITS0.IBM.COM
WSVR0615W: The user.install.root system property is not set. Some product
classes might not be found.
Buildfile:
/etc/websrv1/Plugins/config/actionRegistry/actions/99SBootStrapPluginsIHS.ant

bootstrapPluginsIHS:

detectCurrentOSFamily:
    Ýecho" Detected current OS family to be: ZOS

setOSFileSeparator:
    Ýecho" file separator is /

defineOSSpecificConfigFlag:

SetBitsDir:
    Ýecho" bitsDir is / PLUGIN64: ${plugin.install.64bit} OS-is-32bit:
${is.thirtytwo}

logStartupProperties:
    Ýmkdir" Created dir: /etc/websrv1/Plugins/logs/config
    Ýtouch" Creating /etc/websrv1/Plugins/logs/config/installIHSPlugin.log

updateLogFile:
updateLogFile:
updateLogFile:
updateLogFile:
updateLogFile:
updateLogFile:
```

```
updateLogFile:  
updateLogFile:  
updateLogFile:  
updateLogFile:  
updateLogFile:  
getPlatformPluginDriver:  
  
logBoth:  
    Ýecho" WAS plugin driver set to:  
    /etc/websrv1/Plugins/bin/mod_was_ap22_http.so  
  
updateLogFile:  
  
chkInstall_Arch:  
  
logBoth:  
    Ýecho" 64 bit directory was located.  
  
updateLogFile:  
  
terminateOnFailure:  
  
checkPlgCfg:  
  
logBoth:  
    Ýecho" 64 bit directory was located.  
  
updateLogFile:  
  
terminateOnFailure:  
  
updatePlgCfg:  
  
logBoth:  
    Ýecho" Installing default plugin-cfg.xml file in directory  
    /etc/websrv1/Plugins/config/webserver1  
  
updateLogFile:  
  
createCfgDir:  
  
logBoth:  
    Ýecho" Creating directory /etc/websrv1/Plugins/config/webserver1  
  
updateLogFile:  
    Ýmkdir" Created dir: /etc/websrv1/Plugins/config/webserver1  
  
changeCfgDirPerms:  
    Ýcopy" Copying 1 file to /etc/websrv1/Plugins/config/webserver1
```

```

updatePlgKeyStore:

logBoth:
    Ÿecho" Installing default keystore files in directory
/etc/websrv1/Plugins/config/webserver1

updateLogFile:
    Ÿcopy" Copying 1 file to /etc/websrv1/Plugins/config/webserver1
    Ÿcopy" Copying 1 file to /etc/websrv1/Plugins/config/webserver1
    Ÿcopy" Copying 1 file to /etc/websrv1/Plugins/config/webserver1
    Ÿcopy" Copying 1 file to /etc/websrv1/Plugins/config/webserver1

updatePluginCfgGroupOwnerShip:

checkConfigFiles:

logBoth:
    Ÿecho" Located config file /etc/websrv1/conf/httpd.conf

updateLogFile:

terminateOnFailure:

chkLogDir:

logBoth:
    Ÿecho" Log directory /etc/websrv1/Plugins/logs/webserver1 does not exist

updateLogFile:

createLogDir:

logBoth:
    Ÿecho" Creating directory /etc/websrv1/Plugins/logs/webserver1

updateLogFile:
    Ÿmkdir" Created dir: /etc/websrv1/Plugins/logs/webserver1

changePluginLogFileOwner:
    Ÿtouch" Creating /etc/websrv1/Plugins/logs/webserver1/http_plugin.log

nobodyIfNotZOS:

appendIHSConfigurationFileLoadModuleEntry:

e2a_for_zos:

logBoth:
    Ÿecho" Commenting out previous LoadModule entries

updateLogFile:

logBoth:
    Ÿecho" Commenting out previous bootstrap entries

```

```

updateLogFile:

appendToFileGivenString:

appendToFileGivenString:

logBoth:
    ÿecho" Appending: LoadModule was_ap22_module
/etc/websrv1/Plugins/bin/mod_was_ap22_http.soto/etc/websrv1/conf/httpd.conf

updateLogFile:

appendToFileGivenString:

logBoth:
    ÿecho" Appending: WebSpherePluginConfig
/etc/websrv1/Plugins/config/webserver1/plugin-cfg.xml to
/etc/websrv1/conf/httpd.conf

updateLogFile:

a2e_for_zos:

ConfigureIHSPlugin:

logBoth:
    ÿecho" Install complete

updateLogFile:

BUILD SUCCESSFUL
Total time: 7 seconds

```

6. If you are not using the standard ports for a web server (80 and 443), change them under a virtual host. At the WebSphere Application Server console, click **Environment** → **Virtual hosts** and define your IBM HTTP Server ports to the proper virtual host.
7. Click **Servers** → **New server** and create the web server definition generated during step 5.
8. Save the configuration and restart your application server.
9. Click **Servers** → **Server Types** → **Web servers**, and select your newly created web server, and click the **Propagate Plug-in** button.
10. Restart the web server.

You can find more details about Web Server Plug-Ins for IBM WebSphere Application Server for z/OS configuration at the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Finfo%2Fzseries%2Fae%2Ftrun_plugin_ihsz.html



Part 3

Managing distributed systems



Performance tuning

Performance tuning is a complex task that spans multiple components and areas with a goal of improving system performance. It is heavily dependent on application architecture, system infrastructure, and the amount of load on your system. WebSphere Application Server V8 delivers a scalable and highly available platform for applications and provides multiple methods for tuning options to optimize runtime performance.

This chapter provides information about methods and provides a list of tunable parameters by which you can improve performance. WebSphere provides an extensive number of tunable parameters. We include those parameters that might have the most noticeable benefits for your system. We use the queue analogy to represent WebSphere resource pools and provide a methodological approach to tuning these pools. We also include methods to tune Java virtual machine (JVM) and other various components.

This chapter includes the following topics:

- ▶ Performance tuning facts
- ▶ Using the queue analogy to tune WebSphere resource pools
- ▶ JVM tuning
- ▶ Other tuning considerations
- ▶ Tools

13.1 Performance tuning facts

Application architecture, system infrastructure, and the amount of load on the system are three essential factors that determine the optimum values for the parameters that we introduce in this chapter. There is no single value for any parameter that will work optimally on all systems. You need to test, collect information, and analyze results to reach to a set of optimum values for your system.

To tune for performance, you need to have a performance test system that is identical to your production system. For the results of the test to be meaningful for the production, the hardware and software on the test system should be identical to the production.

To measure the success of your tests, generate a workload that meets the following characteristics:

- ▶ Measurable: Use a metric that can be quantified, such as throughput and response time.
- ▶ Reproducible: Ensure that the results can be reproduced when the same test is executed multiple times. Execute tests in the same conditions to define the real impacts of the tuning changes. Change only one parameter at a time.
- ▶ Static: Determine whether the same results can be achieved no matter for how long you execute the run.
- ▶ Representative: Ensure that the workload realistically represents the stress to the system under normal operating considerations. Execute tests in a production-like environment with the same infrastructure and same amount of data.

To determine performance targets, you need a clear understanding of your system architecture and requirements. Investigate architectural documents, use cases, and functional and non-functional requirements. With a clear understanding, you can define performance success criterias.

Define your own performance success criterias. Without a goal or target, you cannot determine if the performance campaign was successful. You have to avoid non-figured success criteria, for example, aiming for the “best” performance that you can have. Keep in mind that performance testing can be endless if you do not have target figures to reach. Each time you test, you will find a new bottleneck to solve with a new solution. In the end, you cannot determine whether the test is a success or failure.

Do not attempt to conduct performance tuning on production systems with live load, because there is a high chance that the server will be recycled for the new parameters to take effect. This recycling process can cause downtime for the production environment.

13.2 Using the queue analogy to tune WebSphere resource pools

WebSphere Application Server functions similarly to a queuing network, with a group of interconnected queues that represent various resources. Resource pools are established for the network, web server, web container, EJB container, Object Request Broker (ORB), data source, and possibly a connection manager to a custom back-end system. Each of these resources represents a queue of requests waiting to use that resource. Queues are load-dependent resources. As such, the average response time of a request depends on the number of concurrent clients. The tuning of these queues is an essential task to tune for performance.

As an example, think of an application, consisting of servlets and EJB beans, that accesses a back-end database. Each of these application elements reside in the appropriate WebSphere component (for example, servlets in the web container), and each component can handle a certain number of requests in a given time frame.

A client request enters the web server and travels through WebSphere components to provide a response to the client. Figure 13-1 illustrates the processing path that this application takes through the WebSphere components as interconnected pipes that form a large tube.

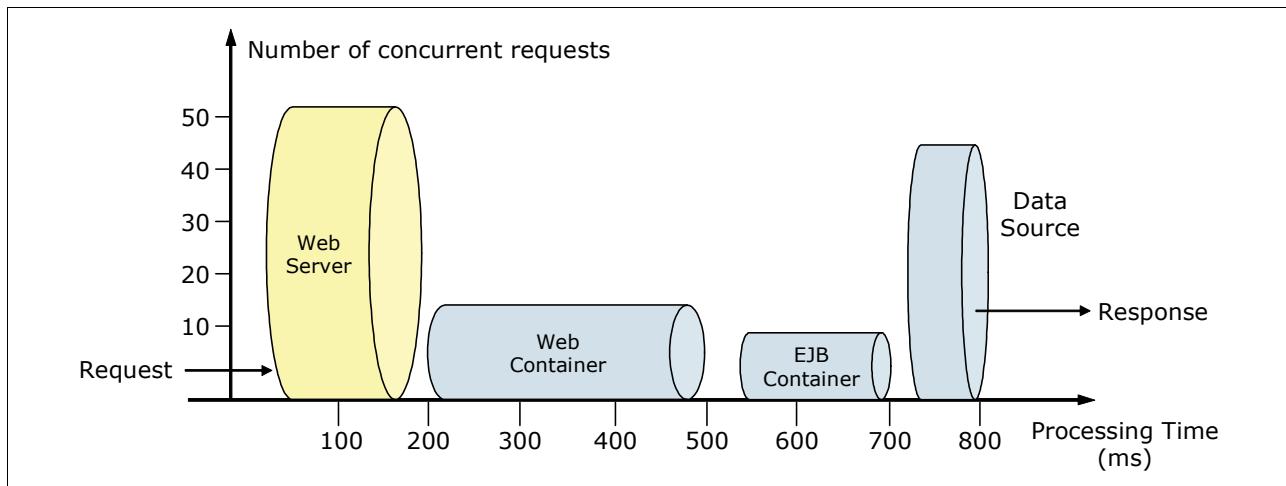


Figure 13-1 Queueing network

The width of the pipes (illustrated by height) represents the number of requests that can be processed at any given time. The length represents the processing time that it takes to provide a response to the request.

Figure 13-1 shows that at any given time, a web server can process 50 requests and a web container can process 18 requests. This difference implies that under peak load requests are queued on the web server side, waiting for the web container to be available. In addition, the EJB container can process nine requests at a given time. Therefore, half of the requests in the web container are also queued, waiting for an ORB thread pool to be available. The database seems to have enough processing power; therefore, requests in the EJB container do not wait for database connections.

Suppose that we have adequate CPU and memory in the WebSphere Application Server system. Then, we can increase ORB pool size to better use the available database connections. Because the web server processes 50 requests at a given time, we can also increase the web container thread pool size to better use the web container threads and to keep up with the increased number of ORB threads.

However, how do you determine the maximum amount of threads and database connections? If requests are queued due to processing differences, do you queue them closer to the data layer or closer to the client for best performance? We provide information about these questions in the sections that follow.

13.2.1 Upstream queuing

The golden rule of resource pool tuning is to minimize the number of waiting requests in WebSphere Application Server queues and to adjust resource pools in a way that resources wait in front of the web server. This configuration allows only requests that are ready to be processed to enter the queuing network. To accomplish this configuration, specify that the queues furthest upstream (closest to the client) are slightly larger and that the queues further downstream (furthest from the client) are progressively smaller. This approach is called *upstream queuing*.

Figure 13-2 shows an example of this type of queuing. When 200 client requests arrive at the web server, 125 requests remain queued in the network because the web server is set to handle 75 concurrent clients. As the 75 requests pass from the web server to the web container, 25 requests remain queued in the web server and the remaining 50 requests are handled by the web container. This process progresses through the data source until 25 user requests arrive at the final destination, the database server. Because there is work waiting to enter a component at each point upstream, no component in this system must wait for work to arrive. The bulk of the requests wait in the network, outside of WebSphere Application Server. This type of configuration adds stability, because no component is overloaded.

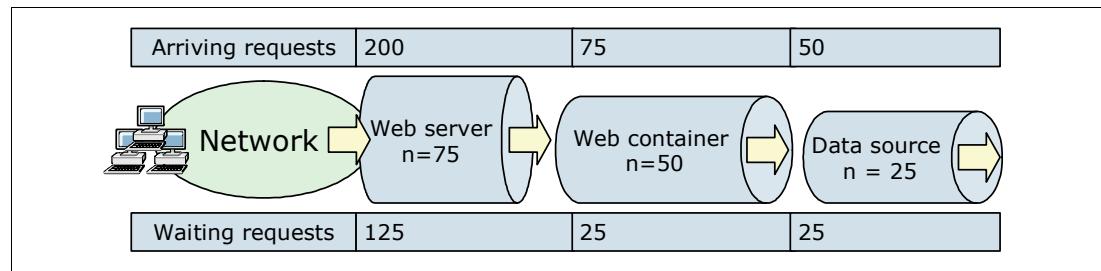


Figure 13-2 Upstream queuing

Because you do not have an infinite amount of physical resources for your system, do not use equal-sized queues. If you had infinite resources, you could tune the system such that every request from the web server has an available application server thread and every application server thread has an available database connection. However, for most real-world systems, this configuration is not possible.

Another important rule for accurate resource tuning is that you need to have a clear understanding of your system infrastructure, application architecture, and requirements. Different systems can have varying access and utilization patterns. For example, in many cases, only a fraction of the requests passing through one queue enters the next queue downstream. However, in a site with many static pages, many requests are fulfilled at the web server and are not passed to the web container.

In this circumstance, the web server queue can be significantly larger than the web container queue. In the previous example, the web server queue was set to 75 requests rather than closer to the value of the Max Application Concurrency parameter. You need to make similar adjustments when different components have different execution times. As the percentage of static content decreases, however, a significant gap in the web server queue and the application server queue can create poorly performing sites overall. Thus, before you begin any tuning activities, understand your system infrastructure, application architecture, and requirements.

For another example, in an application that spends 90% of its time in a complex servlet and only 10% of its time making a short JDBC query, on average 10% of the servlets are using database connections at any time. Thus, the database connection pool can be significantly smaller than the web container pool. Conversely, if much of a servlet execution time is spent making a complex query to a database, consider increasing the pool sizes at both the web container and the data source. Always monitor the CPU and memory utilization for both WebSphere Application Server and the database servers to ensure that the CPU or memory are not being overutilized.

In the following sections, we provide information about how to tune each resource pool, with each pool starting from the furthest downstream, traveling upstream.

13.2.2 Data source tuning

When determining data source queues, consider tuning the following settings:

- ▶ Connection pool size
- ▶ Prepared statement cache size

Connection pool size

When accessing any database, the initial database connection is an expensive operation. WebSphere Application Server provides support for connection pooling and connection reuse. The connection pool is used for direct JDBC calls within the application and for enterprise beans that use the database.

IBM Tivoli Performance Viewer can help you determine the optimal size for the connection pool. Use a standard workload that represents a typical number of incoming client requests, use a fixed number of iterations, and use a standard set of configuration settings. Watch the Pool Size, Percent Used, and Concurrent Waiters counters of the data source entry under the JDBC Connection Pools module. The optimal value for the pool size is the value that reduces the values for these monitored counters. If the Percent Used counter is consistently low, consider decreasing the number of connections in the pool.

Better performance is generally achieved if the value for the connection pool size is set lower than the value for the Max Connections parameter in the web container. Lower settings for the connection pool size (10-30 connections) typically perform better than higher settings (more than 100). On UNIX platforms, a separate DB2 process is created for each connection. These processes affect performance on systems with low memory, causing errors.

Each entity bean transaction requires an additional connection to the database specifically to handle the transaction. Be sure to take this connection into account when calculating the number of data source connections. The connection pool size is set from the Administrative Console by completing the following steps:

1. Click **Resources** → **JDBC Providers** in the console navigation tree.
2. Select the appropriate scope (cell, node, or server), depending on your configuration.
3. Open the JDBC provider configuration by clicking the name of the provider.
4. Select the **Data Sources** entry under Additional Properties.
5. Open the data source configuration by clicking the data source name.
6. Click **Connection pool properties**.
7. Use the Minimum connections and Maximum connections fields to configure the pool size.
8. Save the configuration, and restart the affected application servers for the changes to take effect.

The default values are 1 for Minimum connections and 10 for Maximum connections.

A deadlock can occur if the application requires more than one concurrent connection per thread and if the database connection pool is not large enough for the number of threads. Suppose each of the application threads requires two concurrent database connections and the number of threads is equal to the maximum connection pool size. Deadlock can occur when both of the following statements is true:

- ▶ Each thread has its first database connection, and all connections are in use.
- ▶ Each thread is waiting for a second database connection, and no connections will become available because all threads are blocked.

To prevent the deadlock in this case, the value set for the database connection pool must be at least one higher than the number of waiting threads to have at least one thread complete its second database connection. To avoid deadlock, code the application to use, at most, one connection per thread. If the application is coded to require concurrent database connections per thread, the connection pool must support at least the following number of connections, where T is the maximum number of threads:

$$T * (C - 1) + 1$$

The connection pool settings are directly related to the number of connections that the database server is configured to support. If you raise the maximum number of connections in the pool and if you do not raise the corresponding settings in the database, the application fails and SQL exception errors are displayed in the SystemErr.log file.

Prepared statement cache size

The data source optimizes the processing of prepared statements to help make SQL statements process faster. It is important to configure the cache size of the data source to gain optimal statement execution efficiency. A prepared statement is a precompiled SQL statement that is stored in a prepared statement object. This object is used to efficiently execute the given SQL statement multiple times. If the JDBC driver specified in the data source supports precompilation, the creation of the prepared statement will send the statement to the database for precompilation. Some drivers might not support precompilation, and the prepared statement might not be sent until the prepared statement is executed.

If the cache is not large enough, useful entries are discarded to make room for new entries. In general, the more prepared statements that your application has, the larger the cache should be. For example, if the application has five SQL statements, set the prepared statement cache size to 5, so that each connection has five statements.

Tivoli Performance Viewer can help tune this setting to minimize cache discards. Use a standard workload that represents a typical number of incoming client requests, use a fixed number of iterations, and use a standard set of configuration settings. Watch the PrepStmtCacheDiscardCount counter of the JDBC Connection Pools module. The optimal value for the statement cache size is the setting used to get either a value of zero or the lowest value for the PrepStmtCacheDiscardCount counter.

As with the connection pool size, the statement cache size setting requires resources at the database server. Specifying too large a cache can have an impact on database server performance. Consult your database administrator to determine the best setting for the prepared statement cache size.

Note: The statement cache size setting defines the maximum number of prepared statements cached per connection.

You can set the cache size from the Administrative Console using these steps:

1. Select **Resources** → **JDBC Provider** in the console navigation tree.
2. Select the appropriate scope (cell, node or server), depending on your configuration.
3. Open the JDBC provider configuration by clicking the name of the provider.
4. Select the **Data Sources** entry under Additional Properties.
5. Open the data source configuration by clicking the data source name.
6. Select **WebSphere Application Server data source properties**.
7. Use the Statement cache size field to configure the total cache size.
8. Save the configuration, and restart the affected application servers for the change to take effect.

13.2.3 EJB container

The Enterprise JavaBeans (EJB) container can be another source of potential scalability bottlenecks. The inactive pool cleanup interval is a setting that determines how often unused EJB beans are cleaned from memory. Set this interval too low, and the application spends more time instantiating new EJB beans when an existing instance can be reused. Set the interval too high, and the application has a larger memory heap footprint with unused objects that remain in memory. EJB container cache settings can also create performance issues if not properly tuned for the system.

In the sections that follow, we describe the parameters that you can use to make adjustments that might improve performance for the EJB container.

Cache settings

To determine the cache absolute limit, multiply the number of enterprise beans that are active in any given transaction by the total number of concurrent transactions expected. Then, add the number of active session bean instances. Use Tivoli Performance Viewer to view bean performance information. The cache settings consist of the following parameters:

- ▶ The cache size
The cache size specifies the number of buckets in the active instance list within the EJB container.
- ▶ The cleanup interval
The cleanup interval specifies the interval at which the container attempts to remove unused items from the cache to reduce the total number of items to the value of the cache size.

To change these settings, click **Servers** → **Application servers** → **<AppServer_Name>** → **EJB Container Settings** → **EJB container** → **EJB cache settings**.

The default values are Cache size=2053 buckets and Cache cleanup interval=3000 milliseconds.

ORB thread pool size

Method invocations to enterprise beans are only queued for requests coming from remote clients going through the RMI activity service. An example of such a client is an EJB client running in a separate JVM (another address space) from the enterprise bean. In contrast, no queuing occurs if the EJB client (either a servlet or another enterprise bean) is installed in the same JVM that the EJB method runs on and the same thread of execution as the EJB client.

Remote enterprise beans communicate by using the RMI/IIOP protocol. Method invocations initiated over RMI/IIOP are processed by a server-side ORB. The thread pool acts as a queue for incoming requests. However, if a remote method request is issued and there are no more available threads in the thread pool, a new thread is created. After the method request completes, the thread is destroyed. Therefore, when the ORB is used to process remote method requests, the EJB container is an open queue, due to the use of unbounded threads.

Tivoli Performance Viewer can help tune the ORB thread pool size settings. Use a standard workload that represents a typical number of incoming client requests, use a fixed number of iterations, and use a standard set of configuration settings. Watch the PercentMaxed counter of the Thread Pools module. If the value of this counter is consistently in the double digits, then the ORB might be a bottleneck, and you need to increase the number of threads in the pool.

The degree to which you need to increase the ORB thread pool value is a function of the number of simultaneous servlets (that is, clients) that are calling enterprise beans and the duration of each method call. If the method calls are longer or if the applications spend a lot of time in the ORB, consider making the ORB thread pool size equal to the web container size. If the servlet makes only short-lived or quick calls to the ORB, servlets can potentially reuse the same ORB thread. In this case, the ORB thread pool can be small, perhaps even one-half of the thread pool size setting of the web container.

You can configure the ORB thread pool size from the Administrative Console by completing the following steps:

1. To change these settings, click **EJB cache settings**.
2. Click **Servers** → **Application servers** → <*AppServer_Name*> → **Container Services**.
3. Click **ORB Service** → **Thread Pool**.
4. Use the Maximum Size field to configure the maximum pool size. Note that this setting affects only the number of threads that are held in the pool. (The actual number of ORB threads can be higher.)
5. Save the configuration, and restart the affected application server for the change to take effect.

13.2.4 Web container tuning

Monitor the web container thread pool closely during initial performance runs. This bottleneck is the most common bottleneck in an application environment. Define the web container size considering all the infrastructure chain in close cooperation with the web server number of threads and the number of sessions of the database. If you adjust the number of threads to be too low, the web server threads can wait for the web container. If you adjust the number of threads to be too high, the back end can be inundated with too many requests. For both circumstances, the consequence is an increase of the response time or even a hang.

To route servlet requests from the web server to the web containers, a transport connection between the web server plug-in and each web container is established. The web container manages these connections through transport channels and assigns each request to a thread from the web container thread pool.

Web container thread pool

The web container maintains a thread pool to process inbound requests for resources in the container (that is, servlets and JSP pages).

Tivoli Performance Viewer can help tune the web container thread pool size settings. Use a standard workload that represents a typical number of incoming client requests, use a fixed number of iterations, and use a standard set of configuration settings. Watch the PercentMaxed and ActiveCount counters of the Thread Pools module. If the value of the PercentMaxed counter is consistently in the double digits, the web container can be a bottleneck, and you need to increase the number of threads. Alternatively, if the number of active threads are significantly lower than the number of threads in the pool, consider lowering the thread pool size for a performance gain.

You can configure the web container thread pool size from the Administrative Console by completing the following steps:

1. Click **Servers** → **Application servers** → <*AppServer_Name*>.
2. Click the **Thread Pools** entry under Additional Properties.
3. Click the **WebContainer** entry in the thread pools list of the workspace.
4. Use the Maximum Size field to configure the maximum pool size. Note that in contrast to the ORB, the web container uses threads only from the pool. Thus, this configuration is a closed queue. The default value is 50.
5. Save the configuration, and restart the affected application server for the change to take effect.

Note: Selecting the **Allow thread allocation beyond maximum thread size** option on the Web Container Thread Pool Configuration window allows for an automatic increase of the number of threads beyond the maximum size that is configured for the thread pool. As a result, the system can become overloaded.

HTTP transport channel maximum persistent requests

The maximum persistent requests is the maximum number of requests that are allowed on a single keep-alive connection. This parameter can help prevent denial of service attacks when a client tries to hold on to a keep-alive connection. The web server plug-in keeps connections open to the application server as long as it can, providing optimum performance. A good starting value for the maximum number of requests that are allowed is 100 (which is the default value). If the application server requests are received from the web server plug-in only, increase this value.

You can configure the maximum number of requests that are allowed from the Administrative Console by completing the following steps:

1. Click **Servers** → **Application servers** → <*AppServer_Name*>.
2. Click **Web Container Settings** → **Web container transport chains** under Container Settings.
3. Select the transport chain you want to modify, for example, WCInboundDefault.
4. Click **HTTP Inbound Channel (HTTP #)** in the Transport Channels pane.
5. Enter a value in the Maximum persistent requests field.
6. Click **OK**.
7. Save the configuration, and restart the affected application server for the change to take effect.

13.2.5 Web server tuning

There are several configuration options that impact the performance of the web server, such as the number of concurrent requests, keep-alive settings, or SSL parameters. For the scope of this section, we focus on the number of concurrent requests.

The web server must allow for sufficient concurrent requests to make full use of the application server infrastructure. The web server should also act as a filter and keep users waiting in the network to avoid flooding the servers if more requests than the system can handle are incoming. As a rough initial start value for testing the maximum concurrent threads (one thread can handle one request at a time), use the following setting:

$$\text{MaxClients} = (((\text{TH} + \text{MC}) * \text{WAS}) * 1.26) / \text{WEB}$$

Where:

TH	Number of threads in the web container
MC	MaxConnections setting in the plugin-cfg.xml
WAS	Number of WebSphere Application Server servers
WEB	Number of web servers

Monitoring the web server using tools such as server-status can help tune the maximum concurrent thread processing setting for the web server. Use a standard workload that represents a typical number of incoming client requests, use a fixed number of iterations, and use a standard set of configuration settings. Watch the number of web server threads going to the web container and the number of threads accessing static content locally on the web server. We explain how to enable server-status in 13.5.5, “IBM HTTP server status monitoring page” on page 516.

For additional information about IBM HTTP Server performance tuning, refer to the following websites:

- ▶ <http://www-01.ibm.com/support/docview.wss?uid=swg21167658>
- ▶ http://publib.boulder.ibm.com/httpserv/ihdiag/ih_performace.html

13.2.6 Determining the optimum queue sizes

A simple way to determine the right queue size for any component is to perform a number of load runs against the application server environment at a time when the queues are large, ensuring maximum concurrency through the system.

For example, one approach might be as follows:

- ▶ Set the queue sizes for the web server, web container, and data source to initial values. Set the values as listed in 13.2.7, “Estimating web container and ORB thread pool initial sizes” on page 506 to estimate initial values for the web container and ORB thread pools.
- ▶ Simulate a large number of typical user interactions entered by concurrent users in an attempt to fully load the WebSphere environment. In this context, *concurrent users* means simultaneously active users that send a request, wait for the response, and immediately re-send a new request upon response reception. You can use any stress tool to simulate this workload, such as IBM Rational Performance Tester.
- ▶ Measure overall throughput, and determine at what point the system capabilities are fully stressed (the saturation point).
- ▶ Repeat the process, each time increasing the user load. After each run, record the throughput (requests per second) and response times (seconds per request), and plot the throughput curve.

The throughput of WebSphere Application Server is a function of the number of concurrent requests that are present in the total system. At some load point, congestion will develop due to a bottleneck and throughput will increase at a much lower rate until reaching a saturation point (maximum throughput value). The throughput curve can help you identify this load point.

It is desirable to reach the saturation point by driving CPU utilization close to 100%, because this point gives an indication that a bottleneck is not caused by something in the application. If the saturation point occurs before system utilization reaches 100%, it is likely that another bottleneck is being aggravated by the application.

For example, the application might be creating Java objects that are causing excessive garbage collection mark phase bottlenecks in Java. You might notice these bottlenecks because only one processor is being used at a time on multi-processor systems. On uniprocessor systems, you will not notice the symptom, but will notice only the problems that the symptom is causing.

Figure 13-3 shows an example throughput curve.

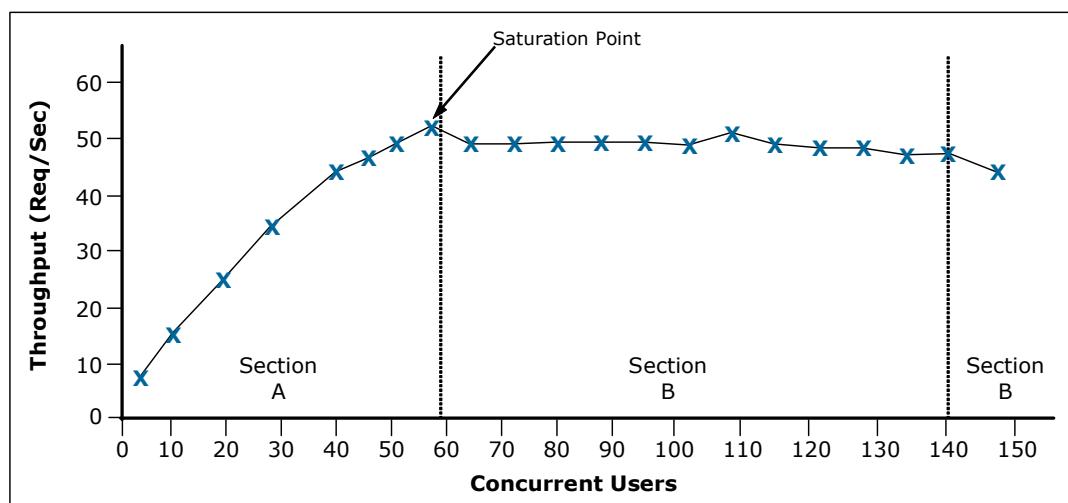


Figure 13-3 Throughput curve

In Figure 13-3, Section A contains a range of users that represent a light user load. The curve in this section illustrates that as the number of concurrent user requests increase, the throughput increases almost linearly with the number of requests. You can interpret this increase to mean that at light loads, concurrent requests face little congestion within the WebSphere Application Server system queues.

In the heavy load zone, Section B, as the concurrent client load increases, throughput remains relatively constant. However, the response time increases proportionally to the user load. That is, if the user load is doubled in the heavy load zone, the response time doubles.

In Section C (the buckle zone), one or more of the system components have become exhausted and throughput degrades. For example, the system might enter the buckle zone if the network connections at the web server exhaust the limits of the network adapter or if the requests exceed operating system limits for file handles.

13.2.7 Estimating web container and ORB thread pool initial sizes

Tuning WebSphere thread pools is a critical activity that has impact on the response time of the applications that run on WebSphere. Although allocating less than an optimum number of threads can result in higher response times, allocating higher than the optimum number can result in over use of system resources, such as CPU, databases, and other resources. A load test can determine the optimum number for thread pool sizes.

When tuning for performance, you are most frequently tuning web container thread and ORB thread pools. As a starting point, you need to know the possible use cases of your applications and the kind of workload that is generated by these use cases. Suppose that you have an online banking application. One use case for a user is to log in, check account details, transfer money to another account, and then log out. You need to estimate how many web and EJB requests are generated by each activity in this use case. (You might also want to gather statistics about how many data source connections this use case uses and for how long it uses these connections.) Then, you need to calculate an estimate of requests for all use cases and determine the total requests per second for your system.

Suppose that your application receives 50 HTTP requests and 30 EJB requests per second. The expected average response time for web requests is 1 second and for EJB requests the response time is 3 seconds.

In this case, you need the following *minimum* number of threads:

- ▶ Web Container Thread Pool: $50/1 = 50$
- ▶ EJB Container Thread Pool: $30/3 = 10$

The *maximum* number of threads that you need depends on the difference between the load in peak time and the average load. Suppose that you have 1.5 times the average load in peak time. Then, you need the following *maximum* number of threads:

- ▶ Web Container Thread Pool = $50 \times 1.5 = 75$
- ▶ EJB Container Thread Pool = $10 \times 1.5 = 15$

13.3 JVM tuning

In this section, we provide information about the JVM parameters that you can tune to increase system performance. You can set the majority of parameters that we explain in this section from the administrative console. Complete the following steps:

1. Navigate to **Servers** → **Application servers** → <**AppServer_Name**>.
2. Under System Infrastructure, expand **Java and Process Management** → **Process definition** → **Java Virtual Machine**.
3. Use the Configuration window to specify the JVM settings.
4. Click **Apply**.

For detailed information about the concepts that we explore in this section, refer to the *Java Diagnostics Guide* at the following website:

<http://publib.boulder.ibm.com/infocenter/javasdk/v6r0/topic/com.ibm.java.doc.diagnostics.60/diag/welcome.html>

13.3.1 Garbage collection policies

Garbage collection has the following phases:

- ▶ **Mark phase**

All live objects are marked in this phase. All reachable objects in the heap are identified, and every other object is treated as garbage. The marking process is also known as *tracing*. The marking phase creates and uses a *mark bit array*, whose bits identify reachable objects in the memory.

- ▶ **Sweep phase**

The sweep phase uses the mark bit array that is created in the mark phase to identify those chunks of heap storage that can be reclaimed for future allocations. These chunks of heap are added to the pool of free space.

The mark and sweep phases can have threads running in parallel or running concurrently. *Parallel* activities run while application threads are halted, and *concurrent* activities run without stopping the application threads.

- ▶ **Compaction phase**

In this phase, the resulting set of objects are compacted to remove the spaces between them. Even if the heap has enough total free space, allocation failures can occur because there is not enough contiguous free space. This state is called *fragmentation* of the Java heap.

Compaction is a remedy for fragmentation, because it moves objects to the beginning of the heap, with no spaces between them, resulting in a contiguous free space for allocations. When an object is moved, the garbage collection changes all the references to that object, which is a complicated and time expensive process that can result in high pause times. By default, compaction is triggered only in certain circumstances, such as not having enough free space to satisfy the allocation request after sweeping.

Setting the **-Xnocompactgc** parameter as a JVM argument prevents compaction and the **-Xcompactgc** parameter forces compaction to occur in every global garbage collection cycle.

The different garbage collection policies are compared by:

- ▶ **Pause time**

In some garbage collection policies, certain activities can acquire exclusive access on the JVM, and all other application threads can pause (such as compaction). The amount of pause time is determined by the size of the heap and by the amount of the garbage objects in the heap.

The JVM uses the following techniques to reduce pause times:

- Concurrent garbage collection
- Generational garbage collection

Pause time has a direct effect on response time of the applications, because lower pause times imply higher response times.

- ▶ **Throughput**

The throughput is the amount of data that is processed by an application in a specific time window. For example, if an application can handle 10 client requests simultaneously and if each request takes one second to process, this site can have a potential throughput of 10 requests per second.

- ▶ Response time

The response time is the period from entering a system at a defined entry point until exiting the system at a defined exit point. In a WebSphere Application Server environment, this time is usually the time that it takes from when a request is submitted by a web browser until the response is received at the web browser.

The sections that follow list the garbage collection policies and provide a means to compare the policies. However, to determine the optimum policy for your system, conduct load tests using different policy options. You can set these policies as a JVM argument using **-Xgcpolicy:policy_name** as follows:

-Xgcpolicy:optthrput

The optthrput policy

This policy includes parallel mark and sweep phases. The collector can use multiple threads to run these two phases. By default, the number of threads that are used is limited by the number of CPUs. Compaction occurs if required. As the name implies, this policy favors a higher throughput than the other policies. Therefore, this policy is more suitable for systems that run batch jobs. However, because it does not include concurrent garbage collection phases, the pause times can be longer than the other policies.

The optavgpause policy

This policy uses concurrent garbage collection to reduce pause times. Concurrent garbage collection reduces the pause time by performing some garbage collection activities concurrently with normal program execution. This method minimizes the disruption that is caused by the collection of the heap. This policy limits the effect of increasing the heap size on the length of the garbage collection pause. Therefore, it is most useful for configurations that have large heaps. This policy provides reduced pause times by sacrificing throughput.

The gencon policy

This policy is the default garbage collection policy for WebSphere Application Server V8.0, and it is the short name for concurrent garbage collection and generational garbage collection combined. This policy includes both approaches to minimize pause times. The idea behind generational garbage collection is splitting the Java heap into two areas, nursery and tenured. New objects are created in a nursery area and, if they continue to be reachable for the tenure age (a defined number of garbage collections), they are moved into the tenured area.

This policy dictates a more frequent garbage collection of only the nursery area, rather than collecting the whole heap as the other policies do. The local garbage collection of the nursery area is called *scavenge*. The gencon policy also includes a concurrent mark phase (not a concurrent sweep phase) for the tenured space, which decreases the pause times of a global garbage collection. The gencon policy can provide shorter pause times and more throughput for applications that have many short-lived objects. It is also an efficient policy against heap fragmentation problems, because of its generational strategy.

The balanced policy

You can use the balanced policy only on 64-bit platforms that have compressed references enabled. This policy involves splitting the Java heap into equal sized areas called *regions*. Each region can be collected independently, allowing the collector to focus on the regions that return the largest amount of memory for the least processing effort. Objects are allocated into a set of empty regions that are selected by the collector. This area is known as an *eden space*.

When the eden space is full, the collector stops the application to perform a partial garbage collection. The collection might also include regions other than the eden space, if the collector determines that these regions are worth collecting. When the collection is complete, the application threads can proceed, allocating from a new eden space, until this area is full. Balanced garbage collection incrementally reduces fragmentation in the heap by compacting part of the heap in every collection. By proactively tackling the fragmentation problem in incremental steps, the balanced policy eliminates the accumulation of work that is sometimes incurred by generational garbage collection, resulting in less pause times.

From time to time, the collector starts a global mark phase to detect if there are any abandoned objects in the heap that are unrecognized by previous partial garbage collections. During these operations, the JVM attempts to use underutilized processor cores to perform some of this work concurrently while the application is running. This behavior reduces any stop-the-world time that the operation might require.

The subpool policy

The subpool policy works on SMP systems (AIX, Linux PPC, and IBM eServer zSeries, and z/OS and i5/OS® only) with 16 or more processors. This policy aims to improve performance of object allocations by using multiple free lists called *subpools* rather than the single free list used by the optavgpause and optthruput policies. Subpools have varying sizes. When allocating objects on the heap, a subpool with a “best fit” size is chosen, as opposed to the “first fit” method used in other algorithms. This policy also tries to minimize the amount of time for which a lock is held on the Java heap. This policy does not use concurrent marking. The subpool policy provides additional throughput optimization on the supported systems.

Comparison of garbage collection policies

Table 13-1 summarizes characteristics and results for different garbage collection policies. Note that some of the results, such as yielding to higher throughput times, is common to more than one policy, because all policies that have this result achieve it by using different algorithms. If your aim is to have higher throughput for your system, you might want to test with all the policies that have this result, as listed in Table 13-1, and come up with the best policy for your system.

Table 13-1 Comparison of garbage collection policies

Policy name	Results
Optthruput	Higher throughput, longer response times for applications with GUI
Optavgpause	<ul style="list-style-type: none"> ▶ Less pause times, shorter response times for applications with GUI ▶ Shorter pause times for large heaps
Gencon	<ul style="list-style-type: none"> ▶ High throughput when application allocates short-lived objects ▶ Shorter response times due to local garbage collection ▶ Effective against heap fragmentation
Balanced	<ul style="list-style-type: none"> ▶ Reduces pause times by incremental compactations ▶ Uses NUMA hardware for higher performance ▶ Dynamically unload unused classes and class loaders on every partial collect ▶ Uses under used cores for the global mark phase
Subpool	Additional throughput optimization on the supported systems

13.3.2 Setting maximum and minimum heap sizes

You can set minimum and maximum heap sizes that are equal, instead of setting a minimum heap size that is smaller than the maximum heap size. This approach prevents heap expansions and compactions that might occur if the minimum heap size is set smaller than the maximum heap size. However, this approach can have the following drawback:

- ▶ Because the minimum heap size is larger, it takes more time for the collections.
- ▶ Because many compactions are omitted, this approach can lead to a more fragmented heap than the minimum less than maximum approach, especially if you are not using generational garbage collection policies.

You can use the following JVM arguments to set the minimum and maximum heap sizes:

-Xms<size>	Sets the initial Java heap size.
-Xmx<size>	Sets the maximum Java heap size.

In you load tests, the best approach is to first set the minimum and maximum values to be equal and then determine the optimum heap size for your system trying different sizes. After you find the optimum number, set the minimum and maximum values around this number, and then test to find the best performing minimum and maximum values. You can compare performance with these different values to see which settings are better for your system.

13.3.3 Sizing the nursery and tenured space when using the gencon policy

The duration of the nursery collect is determined by the amount of data that is copied from the *allocate* space to the *survivor* space, which are two different regions in the nursery space. (We do not provide information about these regions in detail for the scope of this book.) The size of the nursery does not have an increasing effect on scavenges. Instead, increasing the nursery size increases the time between scavenges, which decreases the amount of data that is copied.

The amount of live data that can be copied to the nursery at any time is limited by the amount of transactions that can work concurrently, which is fixed by the container pool settings. Thus, the nursery size can be fixed to an optimum large value. By default, the nursery size is a quarter of the total heap size or 64 MB, whichever is smaller. The nursery can shrink and expand within this size.

You can use the following parameters to define the nursery size:

-Xmns<size>	Sets the initial size of the new area to the specified value. By default, this option is set to 25% of the value of the -Xms option.
-Xmnx<size>	Sets the maximum size of the new area to the specified value. By default, this option is set to 25% of the value of the -Xmx option.
-Xmn<size>	Sets the initial and maximum size of the new area to the specified value. This parameter is equivalent to setting both -Xmns and -Xmnx . If you set either -Xmns or -Xmnx , you cannot set -Xmn .

You can use the following parameters to define the tenured space size:

-Xmos<size>	Sets the initial size of the old (tenure) heap to the specified value. By default, this option is set to 75% of the value of the -Xms option.
-Xmox<size>	Sets the maximum size of the old (tenure) heap to the specified value. By default, this option is set to the same value as the -Xmx option.

-Xmo<size>	Sets the initial and maximum size of the old (tenured) heap to the specified value. Equivalent to setting both -Xmos and -Xmox . If you set either -Xmos or -Xmox , you cannot set -Xmo .
-------------------------	--

For a starting point, you can use **-Xmn** to fix the size of the nursery space to a large enough value. Then, you can treat the tenured space as a Java heap with a non-generational policy, and use the **-Xmos** and **-Xmox** parameters to replace **-Xms** and **-Xmx**.

13.3.4 Using compressed references

The IBM SDK for Java 64-bit stores object references as 64-bit values. The **-Xcompressedrefs** command-line option causes object references to be stored as 32-bit representations, which reduces the 64-bit object size to be the same as a 32-bit object. Because the 64-bit objects with compressed references are smaller than default 64-bit objects, they occupy a smaller memory footprint in the Java heap and improve data locality. Generally, this parameter results in better memory utilization and improved performance. You can perform load testing and compare the results to see whether using this parameter improves performance for your application.

To change to compressed references mode for the JVM, complete the following steps:

1. Click **Environment** → **WebSphere variables**. (You can also enable this variable on a process basis by clicking **Servers** → **Server Types** → **WebSphere application servers** → **server_name** → **Java and process management** → **ProcessDefinition** → **Environment entries**.)
2. Select a scope based on your desired affected environment.
3. Click **New** and specify **IBM_JAVA_OPTIONS** in the name field.
4. Add or append the **-Xcompressedrefs** in the Value field, as shown in Figure 13-4.
5. Click **Apply**.

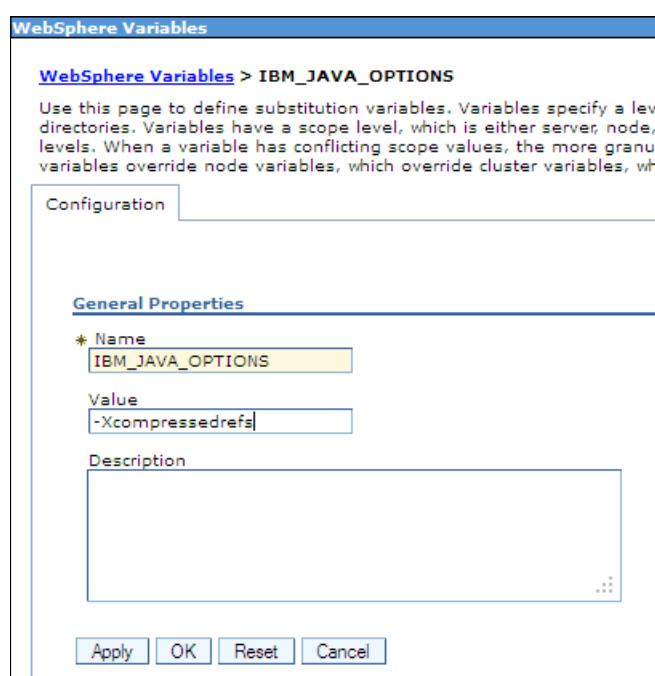


Figure 13-4 Enabling compressed references

6. For the Network deployment configuration, select **Review** and select the **Synchronize changes with nodes** option. Then, click **Save**. Alternatively, click **System administration** → **Nodes**, and click **Synchronize** with the appropriate node or nodes selected.
7. Navigate to **Servers** → **Server types** → **WebSphere Application servers**, and then click **Restart** for the affected server.

13.4 Other tuning considerations

This section describes other tuning considerations that can help and guide you when tuning for performance.

13.4.1 Dynamic caching

The dynamic cache service improves performance by caching the output of servlets, commands, and JavaServer Pages (JSP) files. WebSphere Application Server consolidates several caching activities, including servlets, web services, and WebSphere commands, into one service called the *dynamic cache*. These caching activities work together to improve application performance and share many configuration parameters that are set in an application server's dynamic cache service.

The dynamic cache works within an application server Java Virtual Machine (JVM), intercepting calls to cacheable objects, for example, through a servlet's **service()** method or a command's **execute()** method. The dynamic cache either stores the object's output to or serves the object's content from the dynamic cache. Because J2EE applications have high read/write ratios and can tolerate small degrees of latency in the currency of their data, the dynamic cache can create an opportunity for significant gains in server response time, throughput, and scalability.

Refer to the article found at the following website for detailed information about using the dynamic cache service:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=%2Fcom.ibm.websphere.nd.doc%2Finfo%2Fae%2Fae%2Fcontainer_dyn_admin.html

13.4.2 The pass by reference parameter

Passing EJB beans by value can be expensive in terms of resource use because of the forever remote method call. The parameters are copied onto the stack before the call is made. You can use the pass by reference parameter, which passes the original object reference without making a copy of the object, to avoid this impact.

For EJB 2 or later and 3 or later beans, interfaces can be local or remote. For local interfaces, method calls use the pass by reference parameter by default. If the EJB client and EJB server are installed in the same WebSphere Application Server instance and if the client and server use remote interfaces, specifying the pass by reference parameter can improve performance up to 50%.

Note that using the pass by reference parameters helps performance only when non-primitive object types are being passed as parameters. Therefore, int and floats are always copied, regardless of the call model.

Also, be aware that using the pass by reference parameter can lead to unexpected results. If an object reference is modified by the remote method, the change might be seen by the caller. As a general rule, any application code that passes an object reference as a parameter to an enterprise bean method or to an EJB home method must be scrutinized to determine if passing that object reference results in loss of data integrity or in other problems.

To set this value, use the administrative console to complete the following steps:

1. Click **Servers** → **Application servers** → <*AppServer_Name*> → **Container Services** → **ORB Service**.
2. Select the **Pass by reference** parameter.
3. Click **OK** and then click **Apply** to save the changes.
4. Stop and restart the application server.

13.4.3 Large page support

Many operating system provide the ability to use a larger memory page size than the default memory page size of 4 KB. Having larger page sizes decreases CPU consumption, because the CPU has to manage fewer pages. Therefore, Java applications often benefit from using large pages.

To enable large page utilization, configure the value on your operating system. For more information, refer to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=/com.ibm.websphere.nd.doc/info/ae/ae/tprf_tuneopsys.html

After you configure the operating system, add `-Xlp<size>` as a generic JVM argument to support large pages.

13.4.4 High Performance Extensible Logging

WebSphere Application Server V8.0 provides High Performance Extensible Logging (HPEL), a logging and tracing feature. HPEL is designed to outperform log writing when compared to the basic logging method. It writes logs and traces to a log data repository and a trace data repository in binary format. A log viewer is provided to view, filter, and format the log data and trace data.

Text log file capability: A text log file is also possible, but using this method can impact performance.

HPEL does not replace the existing basic log and trace facility. Instead, it provides flexibility for administrators to manage logging resources, making it easier to work with log and trace content.

Using HPEL provides the following benefits:

- ▶ Log, trace, System.err, and System.out data is stored in collective repositories.
- ▶ Enabling HPEL causes less impact on performance.
- ▶ Better administration of resources used to collect and retain logging information.
- ▶ Enhanced capabilities to work with the logging and trace content.

Refer to the following website for information about how to configure HPEL:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=%2Fcom.ibm.websphere.nd.doc%2Finfo%2Fae%2Fcontainer_hpel_confHPEL.html&resultof=%22hpe1%22

13.4.5 Application tuning

The most important part of your tuning activities is spent on the application. The majority of performance-related problems are related to application design and development implementations. Only a well-designed application, developed with the preferred practices for programming, can give you good throughput and response times. Although environment-related tuning is important to optimize resource use and to avoid bottlenecks, it cannot compensate for a poorly written application.

Review the application code itself as part of the regular application life cycle to ensure that it is using the most efficient algorithms and the most current APIs that are available for external dependencies. For example, take care to use optimized database queries or prepared statements instead of dynamic SQL statements. To help you in this task, you can optimize the application performance using application profiling.

For additional information about the application design considerations, refer to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.nd.multiplatform.doc/info/ae/ae/cprf_appdesign.html

13.5 Tools

This section provides information about the tools that can help and guide you when tuning for performance.

13.5.1 Tivoli Performance Viewer

Tivoli Performance Viewer is included with WebSphere Application Server V8.0 and is used to record and display performance data. Using Tivoli Performance Viewer, you can perform the following tasks:

- ▶ Display PMI data collected from local and remote application servers:
Summary reports show key areas of contention. It also provides graphical and tabular views of raw PMI data.
- ▶ Provide configuration advice through performance advisor section
You can formulate tuning advice from gathered PMI and configuration data.
- ▶ Log performance data
Using Tivoli Performance Viewer, you can log real-time performance data and review the data at a later time.
- ▶ View server performance logs
You can record and view data that is logged using Tivoli Performance Viewer in the Integrated Solutions Console.

Refer to Chapter 15, “Monitoring distributed systems” on page 541 for detailed information about Tivoli Performance Viewer and the underlying monitoring architecture.

13.5.2 Collecting Java dumps and core files using the administrative console

With WebSphere Application Server V8, you can produce a Java dump, Java core, or system dump files directly using the administrative console. These files are useful when you have performance issues that you need to analyze, such as memory, thread, and system behaviors.

To collect Java dump and core files, complete the following steps:

1. Click **Troubleshooting** → **Java dumps and cores**.
2. Select the server or servers.
3. Click **System Dump**, **Java Core**, or **Heap Dump**, as appropriate.

13.5.3 IBM Pattern Modelling and Analysis Tool for Java Garbage Collector

IBM Pattern Modelling and Analysis Tool for Java Garbage Collector (PMAT) is a useful tool that analyzes verbose garbage collection trace. It provides crucial information for garbage collection tuning, such as verbose garbage collection analysis, verbose garbage collection graphics, list of errors, and recommendations.

You can enable verbose garbage collection by selecting the option in the JVM Settings window (refer to 13.3, “JVM tuning” on page 506).

For more information about PMAT and to download the tool, go to the following website:

<http://www.alphaworks.ibm.com/tech/pmat>

13.5.4 IBM Monitoring and Diagnostic tools for Java

IBM Monitoring and Diagnostic tools for Java are available using IBM Support Assistant, a workbench that offers a single point to access these tools. Using IBM Monitoring and Diagnostic tools for Java, you can analyze applications, garbage collection files, Java heap dump files, and Java core files.

For additional information about IBM Monitoring and Diagnostic tools for Java, refer to the following website:

<http://www.ibm.com/developerworks/java/jdk/tools/>

The following sections describes the components of the BM Monitoring and Diagnostic tools for Java.

Health center

Health center allows you to monitor the real-time running applications and provides useful information about memory, class loading, I/Os, object allocations, and the system. This tool can help you to identify application memory leaks, I/O bottlenecks, and lock contentions and can help you to tune the garbage collector. The health center is designed to minimize the performance impact of the monitoring.

Memory analyzer

This tool analyzes the Java heap of a JVM process, identifies potential memory leaks, and provides the application memory footprint. Memory analyzer provides a useful object tree browsing function to focus on the objects interactions and to analyze the memory usage.

Dump analyzer

This tool determines the causes of Java crashes by analyzing the operating system dump. This analysis can be useful to better understand the application failures.

Garbage collection and memory visualizer

This tool helps you analyze and tune the garbage collection, similar to PMAT. It also provides recommendations to optimize the garbage collector and to find the best Java heap settings. Garbage collection and memory visualizer allow you to browse the garbage collection cycles and to better understand the memory behavior of the application.

13.5.5 IBM HTTP server status monitoring page

To monitor IBM HTTP Server, a useful web page called server-status is available. This page is disabled by default, but you can enable it in the httpd.conf configuration file. This web page displays a real-time view of the current IBM HTTP Server state, which includes the following information:

- ▶ The CPU usage
- ▶ The total number of requests served since the server is up
- ▶ The total traffic size since the server is up
- ▶ Some average about the response time
- ▶ The number of requests currently running
- ▶ The number of idle threads
- ▶ And the list of the requests being processed

To enable the server status monitoring page, complete the following steps:

1. Edit the IBM HTTP Server httpd.conf file, and remove the comment character (#) from the following lines in this file:

```
#LoadModule status_module, modules/ApacheModuleStatus.dll,  
#<Location/server-status>  
#SetHandler server-status  
#</Location>
```

2. Save the changes, and restart IBM HTTP Server.

In a web browser, go to: http://your_host/server-status. Click **Reload** to update the status. If the browser supports refresh, go to http://your_host/server-status?refresh=5 to refresh every 5 seconds.

13.5.6 WebSphere performance advisors

When gathering runtime information, the WebSphere performance advisors provide diagnostic advice about the environment. The advisors can determine the current configuration for an application server. Then, by trending the runtime data over time, the advisors provide advice about potential environmental changes that can enhance the performance of the system. The advice is hardcoded into the system and is based on IBM preferred practices for tuning and performance.

The advisors do not implement any changes to the environment. Instead, they identify the problem and allow the system administrator to make the decision as to whether to implement. Perform tests after any change is implemented.

WebSphere provides the following types of advisors:

- ▶ Performance and Diagnostic Advisor
- ▶ Performance Advisor in Tivoli Performance Viewer

Performance and Diagnostic Advisor

This advisor is configured through the Integrated Solutions Console. It writes to the application server log files and to the console while in monitor mode. To minimize the performance impact, configure the server to use HPEL instead of using the `systemOut.log` file.

The interface is configurable to determine how often data is gathered and how often advice is written. It offers advice about the following components:

- ▶ J2C Connection Manager
 - Thread pools
 - LTC Nesting
 - Serial reuse violation
 - Plus various different diagnostic advises
- ▶ Web Container Session Manager
 - Session size with overflow enabled
 - Session size with overflow disabled
 - Persistent session size
- ▶ Web Container
 - Bounded thread pool
 - Unbounded thread pool
- ▶ Orb Service
 - Unbounded thread pool
 - Bounded thread pool
- ▶ Data Source
 - Connection pool size
 - Prepared statement cache size
- ▶ Java virtual machine (JVM)
 - Memory leak detection

If you need to gather advice about items outside this list, use the Performance Advisor in Tivoli Performance Viewer.

Performance Advisor in Tivoli Performance Viewer

This advisor is slightly different from the Performance and Diagnostic Advisor. The Performance Advisor in Tivoli Performance Viewer is invoked only through the Tivoli Performance Viewer interface of the Integrated Solutions Console. It runs on the application server that you are monitoring, but the refresh intervals are based on selecting refresh through the console. Also, the output is routed to the user interface instead of to an application server output log file. This advisor captures data and gives advice about more components. Specifically, this advisor can capture the following types of information:

- ▶ ORB service thread pools
- ▶ Web container thread pools
- ▶ Connection pool size
- ▶ Persisted session size and time
- ▶ Prepared statement cache size
- ▶ Session cache size

- ▶ Dynamic cache size
- ▶ JVM heap size
- ▶ DB2 performance configuration

Running the Performance Advisor in Tivoli Performance Viewer requires resources and can impact performance. Use it with care in production environments.

Refer to Chapter 15, “Monitoring distributed systems” on page 541 for detailed information about performance advisors.



Clustering, workload management, and high availability

Clustering is a fundamental approach for accomplishing high availability. IBM WebSphere Application Server Network Deployment V8 offers a built-in application server clustering function and the high availability (HA) manager for protecting WebSphere singleton services. Clustering application servers provide workload management (WLM) and failover for applications that reside on the application server cluster.

This chapter provides information about the three closely related topics of clustering:

- ▶ Clustering
- ▶ Workload management
- ▶ High availability and failover

This chapter provides an overview of the concepts and configuration steps and investigates how features and components work together to establish high availability and workload management.

14.1 Clustering

A *cluster* is a set of application servers that are managed together and participate in workload management. Application servers participating in a cluster can be on the same node or on different nodes. A Network Deployment cell can contain no clusters or can have many clusters, depending on the need of the administration of the cell. The cluster is a logical representation of the application servers. It is not necessarily associated with any node and does not correspond to any real server process that is running on any node. A cluster contains only application servers and the weighted workload capacity that is associated with those servers.

When creating a cluster, you can select an existing application server as the template for the cluster without adding that application server into the new cluster. (The application server that you use is only a template and is not affected in any way by the cluster creation.) All other *cluster members* are then created based on the configuration of the first cluster member.

Cluster members can be added to a cluster using various methods during cluster creation and afterwards. During cluster creation, you can add one existing application server to the cluster, or you can create and add one or more new application servers to the cluster. You can also add additional members to an existing cluster later. Depending on the capacity of your systems, you can define different weights for the various cluster members. Cluster members are required to have identical application components, but they can be sized differently in terms of weight, heap size, and other environmental factors. Be careful, however, not to change any settings that might result in different application behavior on each cluster member. This concept allows large enterprise systems to belong to a cluster that also contains smaller systems, such as Intel based Windows system servers.

Starting or stopping a cluster starts or stops all cluster members automatically, and changes to the application are propagated to all application servers in the cluster.

Figure 14-1 shows an example of a possible configuration that includes server clusters. Server Cluster 1 has two cluster members on Node B only. Server Cluster 2, which is completely independent of Server Cluster 1, has two cluster members on Node A and three cluster members on Node B. Finally, Node A also contains a free-standing application server that is not a member of any cluster.

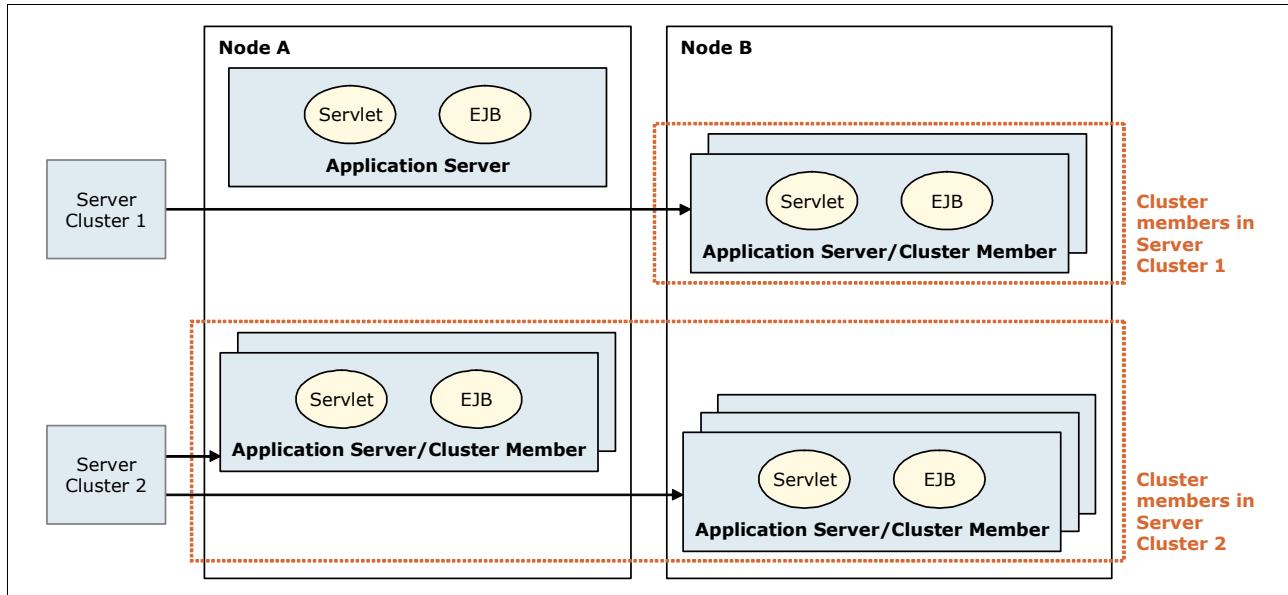


Figure 14-1 Clustering examples

14.1.1 Clustering for scalability and failover

Clustering can be achieved using vertical scaling, horizontal scaling, or a combination of both approaches.

Vertical scaling

In *vertical scaling*, as shown in Figure 14-2, multiple cluster members for an application server are defined on the same physical machine or node, which might allow the machine's processing power to be more efficiently allocated. Even if a single JVM can fully use the processing power of the machine, you might still want to have more than one cluster member on the machine for other reasons, such as using vertical clustering for process availability. If a JVM reaches a table or memory limit (or if there is some similar problem), then the presence of another process provides for failover.

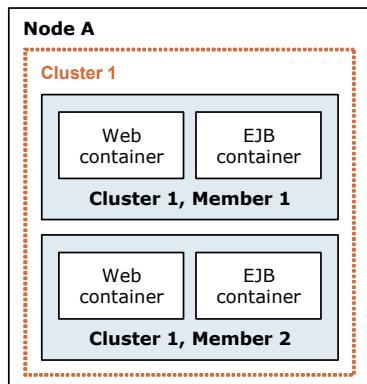


Figure 14-2 Vertical scaling

An accurate way to determine the correct vertical scaling for your environment and application is to tune a single instance of an application server for throughput and performance, then add it to a cluster, and incrementally add additional cluster members. Test performance and throughput as each member is added to the cluster. Always monitor memory usage when you are configuring a vertical scaling topology, and do not exceed the available physical memory on a machine.

In general, 85% (or more) utilization of the CPU on a large server shows that there is little, if any, performance benefit to be realized from adding additional cluster members.

Horizontal scaling

In *horizontal scaling*, as shown in Figure 14-3, cluster members are created on multiple physical machines (or LPARs). This configuration allows a single WebSphere application to run on several machines while still presenting a single system image, making the most effective use of the resources of a distributed computing environment. Horizontal scaling is especially effective in environments that contain many smaller, less powerful machines. Client requests that overwhelm a single machine can be distributed over several machines in the system.

Failover is another important benefit of horizontal scaling. If a machine becomes unavailable, its workload can be routed to other machines that contain cluster members.

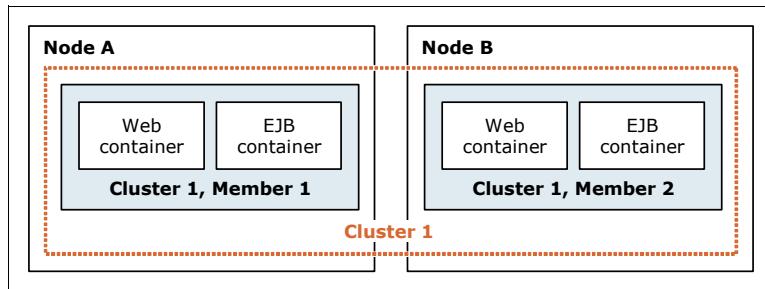


Figure 14-3 Horizontal scaling

Horizontal scaling can handle application server process failures and hardware failures (or maintenance) without significant interruption to client service.

Combining vertical and horizontal scaling

WebSphere applications can combine horizontal and vertical scaling to reap the benefits of both scaling techniques, as shown in Figure 14-4.

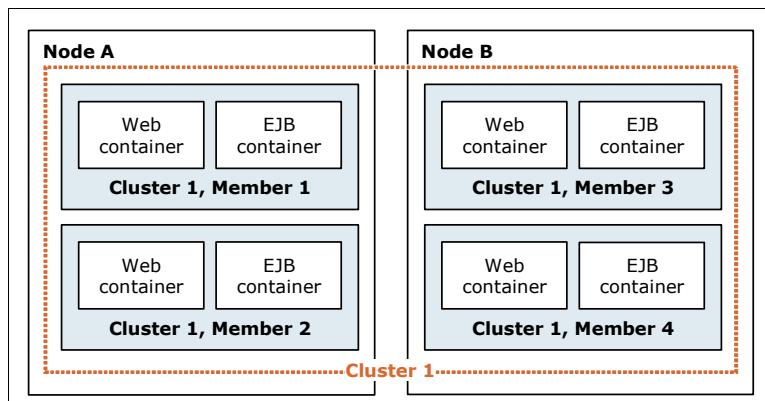


Figure 14-4 Hybrid scaling approach

14.1.2 Creating a cluster

To create a cluster from the administrative console, complete the following steps:

1. Click **Servers** → **Clusters** → **WebSphere application server clusters**, and then click **New**.
2. Enter basic cluster information, as shown in Figure 14-5:
 - a. Enter a cluster name.
 - b. Select one of the following options:
 - The **Prefer local** option routes EJB requests to the same host as the host of the requesting EJB client.
 - The **Configure HTTP session memory-to-memory replication** option creates a replication domain for the cluster.
 - c. Click **Next**.

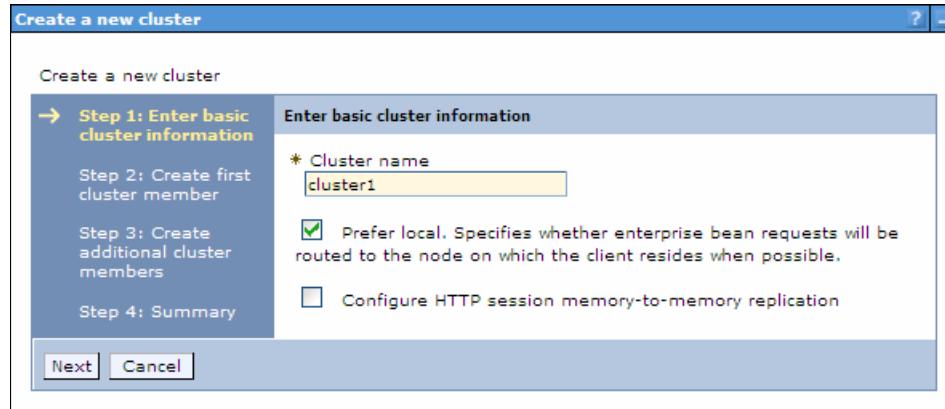


Figure 14-5 Enter basic cluster information

3. Create the first cluster member, as shown in Figure 14-6:
 - a. Enter a member name. Then, select the node, and enter the weight.
 - b. Select the **Generate unique HTTP ports** option to ensure that the server is created with unique port numbers.
 - c. The “Select how the server resources are promoted in the cluster” drop-down menu defines in which scope resources, such as data sources, are initially created in the cluster. Possible options are cluster, server, and both.
 - d. Create the first cluster member based on one of the following options:
 - Using an application server template
 - Using an existing application server
 - Converting an existing server
 - Creating an empty cluster
 - e. Click **Next**.

Create a new cluster

Create first cluster member

The first cluster member determines the server settings for the cluster members. A server configuration template is created from the first member and stored as part of the cluster data. Additional cluster members are copied from this template.

* Member name
member01

Select node
Node01(ND 8.0.0.0)

* Weight
2 (0..20)

Generate unique HTTP ports

Select how the server resources are promoted in the cluster.
Cluster

Select basis for first cluster member:

- Create the member using an application server template.
default
- Create the member using an existing application server as a template.
Cell01/Node01(ND 8.0.0.0)/k1
- Create the member by converting an existing application server.
Cell01/Node01(ND 8.0.0.0)/server1
- None. Create an empty cluster.

Previous Next Cancel

Figure 14-6 Create first cluster member

4. Add additional cluster members, by specifying the following information for each new member:
 - Member name
 - Node
 - Weight

To add a new member, click **Add Member**, as shown in Figure 14-7. Then, repeat steps 3a-e to add other members.

When you are finished adding new members, click **Next**.

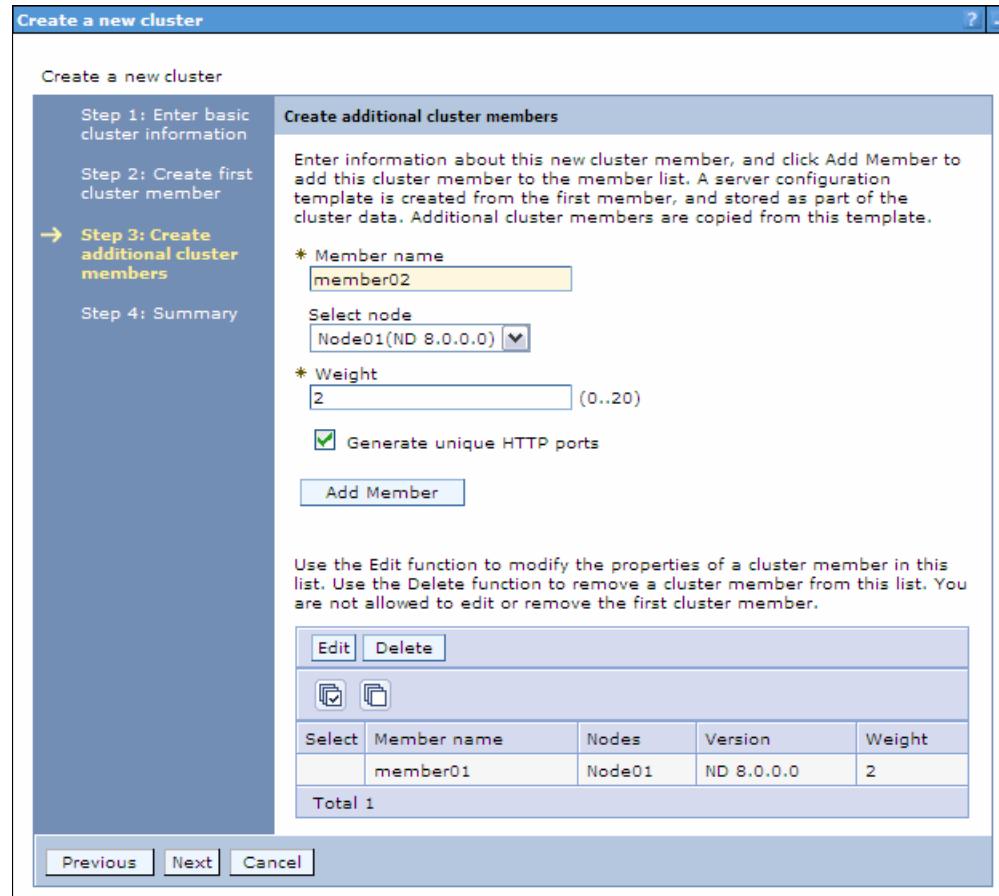


Figure 14-7 Create additional cluster members

5. In the summary window, click **Finish** to create the cluster.

14.2 Workload management

The ability to route a request to any server in a group of clustered application servers allows the servers to share work and to improve throughput of client requests. Requests can be distributed evenly to servers to prevent workload imbalances in which one or more servers has idle or low activity while other servers are overburdened. This load balancing activity is a benefit of workload management.

Thus, this configuration ensures that each machine or server in the configuration processes a fair share of the overall client load that is being processed by the system as a whole. In other words, it is inefficient to have one machine overloaded while another machine is mostly idle. If all machines have roughly the same capacity (for example, CPU power), each machine can process a roughly equal share of the load. Otherwise, consider a provision for the workload to be distributed in proportion to the processing power that is available on each machine.

Using weighted definitions of cluster members allows nodes to have different hardware resources and still participate in a cluster. The weight specifies that the application server with a higher weight is more likely to serve the request faster, and workload management consequently sends more requests to that node.

With several cluster members available to handle requests, it is more likely that failures will not negatively affect throughput and reliability. With cluster members distributed to various nodes, an entire machine can fail without any application downtime. Requests can be routed to other nodes if one node fails. Clustering also allows for maintenance of nodes without stopping application functionality.

This section provides an overview of WebSphere WLM. The available WLM policies and how requests are distributed among available servers is described in great detail in *WebSphere Application Server V6 Scalability and Performance Handbook*, SG24-6392.

14.2.1 Components that can be workload managed

In a typical WebSphere Application Server Topology, the components that we describe in this section can be workload managed.

HTTP requests and web servers

An IP sprayer component, such as the Edge Components Load Balancer or a network appliance, can be used to perform load balancing and workload management functionality for incoming HTTP requests, as shown on Figure 14-8.

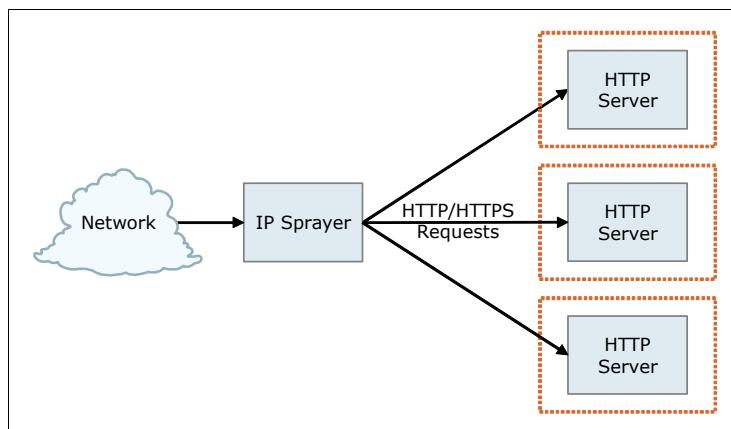


Figure 14-8 HTTP requests and HTTP server WLM

HTTP requests and plug-in

When an HTTP request reaches the HTTP server, a decision must be made. Some requests for static content might be handled by the HTTP server. Requests for dynamic content or static content are passed to a web container that is running in an application server. Whether the request is handled or passed to WebSphere is decided by the WebSphere web server plug-in, which runs in-process with the HTTP server. We refer to this process as *plug-in WLM*, as illustrated in Figure 14-9. For these WebSphere requests, high availability for the web container becomes an important piece of the failover solution.

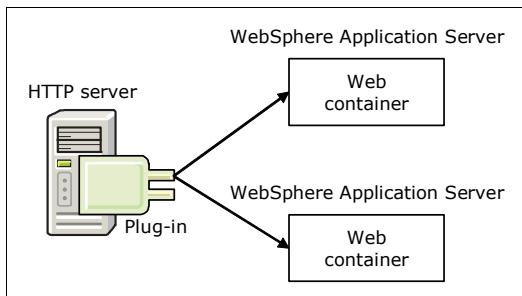


Figure 14-9 Plug-in WLM

WebSphere provides the following load balancing options:

- #### ► Round-robin

This routing is based on the weight that is associated with the cluster members. If all cluster members have identical weights, the plug-in sends equal requests to all members of the cluster, assuming no strong affinity configurations. If the weights are scaled in the range from 0 to 20, the plug-in routes requests to those cluster members with the higher weight value more often. No requests are sent to cluster members with a weight of zero (0) unless no other servers are available. Round-robin is the default load balance policy.

Use the following formula as a guideline for determining routing preference:

% routed to Server1 = weight1 / (weight1+weight2+...+weightn)

Where there are n cluster members in the cluster.

- ## ► Random

The plug-in picks a member of the cluster randomly.

The load balancing options are impacted by the session affinity. After a session is created at the first request, all the following requests have to be served by the same member of the cluster. The plug-in retrieves the application server that serves the previous request by analyzing the session identifier and tries to route to this server. We describe session management concepts in detail in Chapter 25, “Session management” on page 889.

RMI/IOP (EJB) requests

EJB requests can be distributed across multiple EJB containers. When an EJB client makes calls from the web or client container or from outside, the request is handled by the EJB container in one of the clustered application servers. If that server fails, the client request is redirected to another available server. We refer to this process as *EJB WLM*, as illustrated in Figure 14-10.

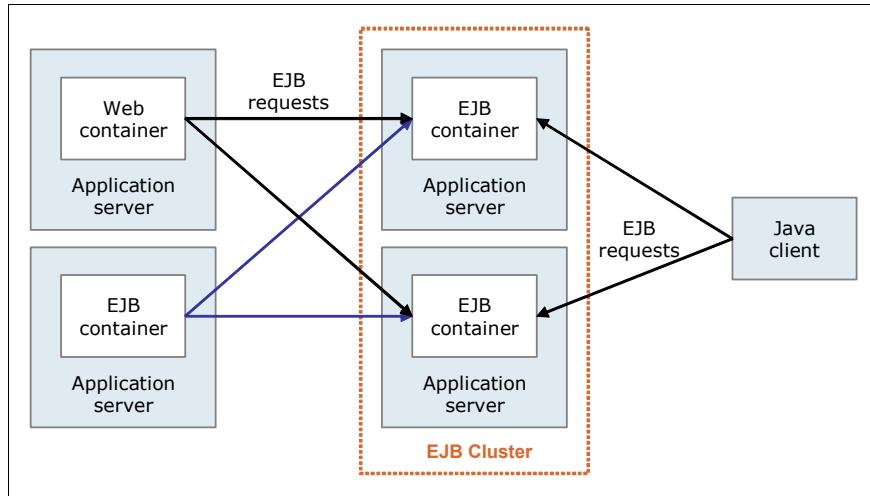


Figure 14-10 EJB WLM

To route the EJB requests, WebSphere provides the following main routing policies:

- ▶ Server weighted round-robin

In this configuration, EJB client requests are routed to available EJB servers in a round-robin fashion based on assigned server weights. The EJB clients can be servlets operating within a web container, stand-alone Java programs using RMI/IOP, or other EJB beans. The server weighted round-robin routing policy ensures a distribution based on the set of server weights that are assigned to the members of a cluster.

For example, if all servers in the cluster have the same weight, the expected distribution for the cluster is that all servers receive the same number of requests. If the weights for the servers are not equal, the distribution mechanism sends more requests to the higher weight value servers than the lower weight value servers.

- ▶ Prefer local

You can also choose that EJB requests are routed preferably to the same host as the host of the requesting EJB client. In this case, only cluster members on that host are chosen (using the round-robin weighted method). Cluster members on remote host are chosen only if a local server is not available.

The following affinity policies can also impact routing:

- ▶ Process affinity

If an EJB is available in the same cluster member as the client, all requests from that client are directed to the EJB in the same JVM process. One of the advantages of this policy is that there is no need for serialization for method calls.

- ▶ Transaction affinity

All the requests from the same transaction are directed to the same cluster member. This policy overwrites all other policies.

14.2.2 WLM benefits

Workload management provides the following benefits to WebSphere applications:

- ▶ It balances client processing requests, allowing incoming work requests to be distributed according to a configured WLM selection policy.
- ▶ It provides failover capability by redirecting client requests to a running server when one or more servers are unavailable. This redirection improves the availability of applications and administrative services.
- ▶ It enables systems to be scaled up to serve a higher client load than provided by the basic configuration. With clusters and cluster members, additional instances of servers can be added easily to the configuration.
- ▶ It enables servers to be maintained and upgraded transparently while applications remain available for users.
- ▶ It centralizes administration of application servers and other objects.

14.3 High availability and failover

High availability is also known as *resiliency*. High availability is the description of the system's ability to tolerate a certain amount of failures and to remain operational. This section provides information about WebSphere Application Server high availability concepts and features.

14.3.1 Overview

High availability (HA) means that your infrastructure continues to respond to client requests no matter what the circumstances are. Depending on the errors or failures, the infrastructure can run in a degraded mode. HA is achieved by adding redundancy in your infrastructure to support the system when failures occur. Availability impacts both performance and scalability. Depending on your needs, you have to define the level of HA for your infrastructure.

The most common method of describing availability is by the “nines” or the percentage availability for the system. For example, 99.9% of system availability represents 8.76 hours of outage in a single year. Table 14-1 shows the level of availability and the calculated downtime per year.

Table 14-1 Availability matrix

Availability %	Downtime per year
99% (two 9s)	87.6 hours
99.9% (three 9s)	8.76 hours
99.99% (four 9s)	56.56 minutes
99.999% (five 9s)	315.36 seconds

You can calculate availability using the following formula, where MTBF is the mean time between failure and MTTR is the maximum time to recovery:

$$\text{Availability} = (\text{MTBF}/(\text{MTBF} + \text{MTTR})) \times 100$$

Keep in mind that the overall infrastructure is available only if all the components are available. For a WebSphere infrastructure that is composed of several components, such as load balancers, HTTP servers, application servers, database servers, and so on, availability is determined by the weakest component.

For most of the environment's components, several degrees of HA implementation exist. The cost of the infrastructure is directly linked to the level of availability. Evaluate the business loss of the infrastructure downtime, and ensure that the business case justifies the costs. Moving a system availability from 99.9% to 99.99% can be expensive. It can also be true that the system is used only during regular business hours on regular working days. This use implies that an availability of 99.9% is more than adequate to meet the operational window.

For additional information about this topic, go to the following website:

http://www.ibm.com/developerworks/websphere/techjournal/0312_polooff/polooff.html#sec1

Because it is likely that the complete environment is made up of multiple systems, the goal is to make the whole system as available as possible by minimizing the number of single points of failure (SPOFs) throughout the system by adding redundancy. Redundancy can be added at different layer, such as hardware, process, and data.

14.3.2 WebSphere Application Server high availability and failover

This section provides information about the WebSphere Application Server HA features. It can help you to understand how the HA features work and can assist you in planning and configuring for HA.

Figure 14-11 represents a typical WebSphere Application Server topology, where a request travels through the load balancer to the HTTP server to the web container and finally to the EJB container.

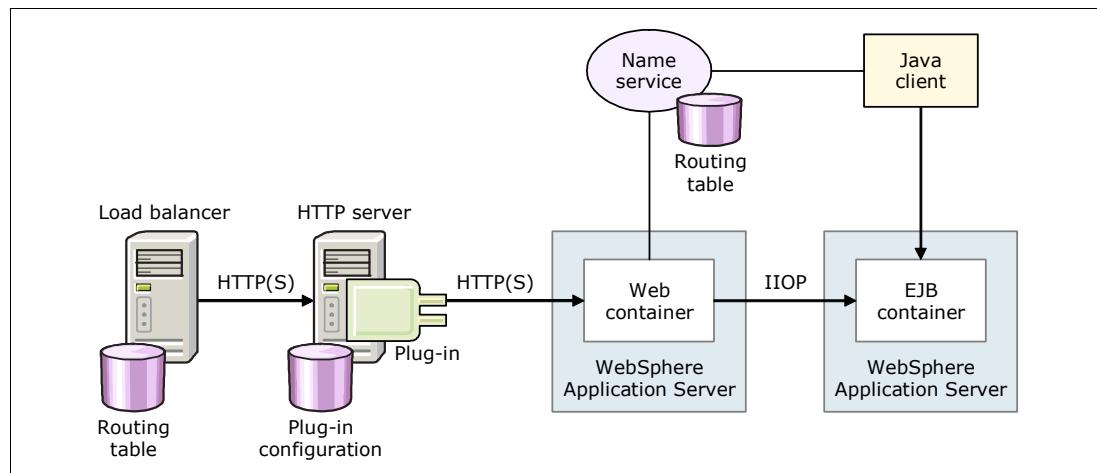


Figure 14-11 WebSphere topology and incoming request

The sections that follow provide information about HA and failover options for components and services that are involved in the processing of the request.

Load balancer high availability and failover

You can configure the Edge Components Load Balancer to run as a active-passive pair. The active instance is the single point of distribution for HTTP requests. If it fails, the passive instance routes the requests.

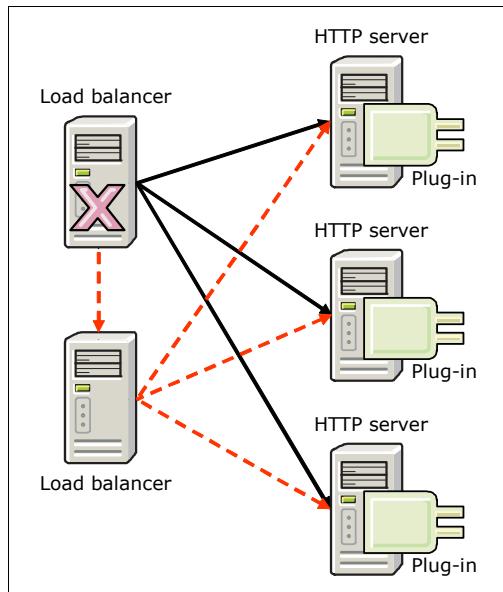


Figure 14-12 Edge Components Load Balancer H

If you use a third-party load balancer, consult with the vendor for load HA features.

Web server high availability and failover

When a web server fails, the Edge Components Load Balancer detects the failure and routes the requests around the failing web server. If there is a session affinity to a server, it still holds, because all web servers have a copy of the plug-in that is capable of routing the requests to servers with affinities.

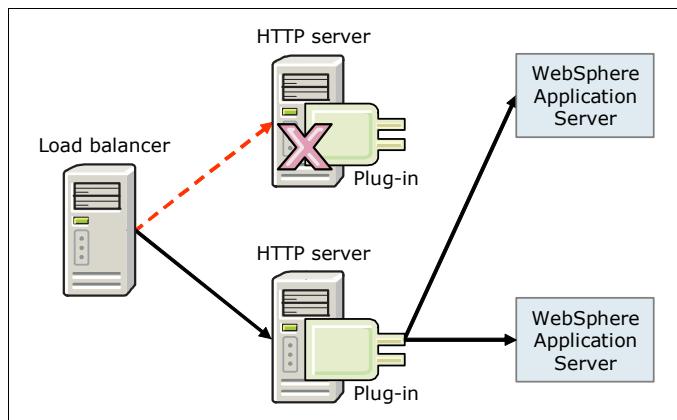


Figure 14-13 Web server H

Web container high availability and failover

A typical WebSphere environment can include several web server instances as well as several WebSphere Application Server instances. Each HTTP server is configured to run the WebSphere web server plug-in. Each request coming into the web server is passed through the plug-in, which uses its configuration information to determine if the request is routed to WebSphere and, if so, to which application server (that is, to which web container) the request is routed (refer to “HTTP requests and plug-in” on page 527 for more information). The communication between the plug-in and the application servers can be either HTTP or HTTPS. The plug-in, which runs in-process with the web server itself, determines to which web container the request is passed. The plug-in also distributes requests around cluster members that are not available.

WebSphere Application Server provides the following mechanisms for web container WLM and failover:

- ▶ Application server clustering creates server process redundancy for failover support. All application servers in a cluster host the same application or applications.
- ▶ The workload management routing technique and the failure detection mechanism is built into the WebSphere web server plug-in. The plug-in can detect a failed web container and mark it down. Then, the plug-in continues to route the request to available server processes.
- ▶ Session management and a failover mechanism provide HTTP session data for redundant server processes.

Satisfactory failover support for web clients can be achieved only by using all three mechanisms, as illustrated in Figure 14-14.

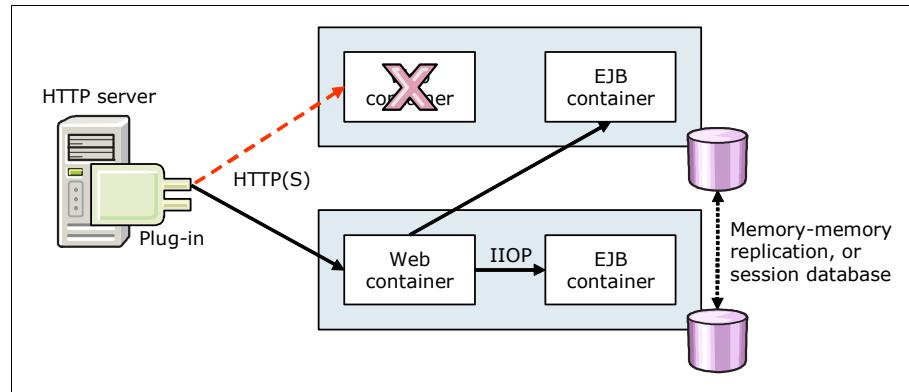


Figure 14-14 Web container HA

EJB container clustering and failover

Many J2EE applications rely on Enterprise JavaBeans (EJB) to implement key business logic. Therefore, providing a resilient and highly available EJB runtime system is a critical task for any EJB container provider. WebSphere Application Server V8 satisfies this requirement for EJB applications by providing an advanced HA solution, which guarantees that EJB requests can be serviced continuously even during various types of failures.

EJB clients can be servlets, JSP pages, a J2EE client, stand-alone Java applications, or other EJB beans. When an EJB client makes calls from within the WebSphere or client container or outside of a container, the request is handled by the EJB container in one of the cluster members. If that cluster member fails, the client request is redirected automatically to another available server, as illustrated in Figure 14-15.

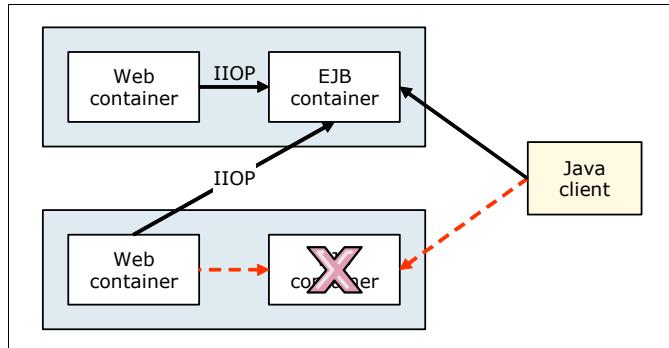


Figure 14-15 EJB container HA

In IBM WebSphere Application Server Network Deployment V8, EJB HA is achieved using a combination of the following WebSphere services:

- ▶ The HA manager
- ▶ The EJB server cluster
- ▶ EJB WLM

The mechanisms for routing workload-managed EJB requests to multiple cluster members are handled on the client side of the application. In WebSphere Application Server V8, this functionality is supplied by a workload management plug-in to the client ORB and the routing table in the Location Service Daemon (LSD) that is hosted in the node agent. The WLM failover support for EJB beans maintains the routing table and modifies the client ORB to redirect traffic in case of a server failure.

EJB failover depends on whether the type of EJB can be workload-managed by the container. Table 14-2 summarizes failover support for different EJB types.

Table 14-2 Summary of EJB types and failover support

EJB type	Component	Failover capable
Entity bean (Option A)	Home CMP bean instance BMP bean instance	Yes No No
Entity bean (Options B and C)	Home CMP bean instance BMP bean instance	Yes Yes Yes
Session Bean	Home Stateless bean instance Stateful bean instance	Yes Yes Yes

EJB WLM failover scenarios

EJB WLM failover can happen in the following scenarios:

- ▶ A failure occurs during EJB processing

This failure triggers an exception. Normally, WebSphere WLM catches the exception and resends the failed request to another application server in the cluster. This method is the normal failover function of WebSphere EJB WLM. If, however, WLM cannot determine whether the transaction completed, it sends an exception to the application.

- ▶ A failure occurs during WLM processing

If the WLM mechanism cannot handle the request, it sends an exception to the application.

Exceptions that cause WLM to fail over

The EJB WLM catches most of the exceptions that can occur during execution. WLM handles these exceptions and decides whether the original request is redirected to another available cluster member. The EJB WLM includes the following exceptions:

- ▶ `org.omg.CORBA.COMM_FAILURE`
- ▶ `org.omg.CORBA_NO_RESPONSE`

These exceptions have a COMPLETION_STATUS of NO, YES, or MAYBE, and these values actually determine whether the request fails over to another available member as follows:

- ▶ With a COMPLETION_STATUS of COMPLETED_NO, automatic failover occurs because the request was not completed. The request is then rerouted to an available cluster member.
- ▶ With a COMPLETION_STATUS of COMPLETED_YES, there is no need to fail over because the transaction completed successfully. Communication errors might have occurred during the marshalling of the answer.
- ▶ With a COMPLETION_STATUS of COMPLETED_MAYBE, WLM cannot verify whether the transaction was completed and, thus, cannot redirect the request. WLM, according to the programming model, sends this exception to the client application. It is the applications' responsibility to handle this exception and to decide whether to retry the request. This status does not mean that something is broken. The application has to check whether the transaction was successful and, depending on the result, should then issue the request.

Exceptions during failover

During the failover process, WLM might face problems and then try to redirect the request again or issue a CORBA exception. If a CORBA exception issues, something unusual occurred while WLM was trying to redirect the original request to another server. There are two reasons for such an exception to occur. Both events are contained in an `org.omg.CORBA.TRANSIENT` exception, with different minors:

- ▶ Communication problem with one of the other servers

The following exception means that an error occurred during the communication and that the client should retry the request:

`TRANSIENT_SIGNAL_RETRY (minor=1229066306)`

- ▶ No reachable server

The following exception means that WLM did not find a cluster member that is able to answer the request:

`NO_IMPLEMENT_NO_USEABLE_TARGET (minor=1229066304)`

This exception is the worst case scenario. The application cannot retry the request because it would fail again. This failure then leads to a message to the user, for example, asking the user to contact the system administrator.

Session persistence

For information about session persistence, refer to 25.8, “Persistent session management” on page 903.

Default messaging provider availability

The service integration bus (SIBus) is used to provide HA to the messaging system process. WebSphere Application Server provides the ability to configure the following policies to achieve message engine HA based on the cluster utilization:

- ▶ **High Availability**

One message engine is created in the cluster and can fail over to any other server in the cluster. The message engine does not fail back to the previous server if this server becomes available again.

- ▶ **Scalability with High Availability**

One message engine is created for each application server on the cluster. Each message engine can fail over to any other server in the cluster. All the messages set for high reliability that were being processed or queued will continue to be processed when the message engine is available on another server. Each message engine can fail back to previous server when this server is available again.

To accomplish the failover seamlessly, the queue information and message data must be stored in a shared location that is reachable by all the members of the cluster, either using an external database or a shared disk environment.

For additional information about the availability policy, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=%2Fcom.ibm.websphere.nd.multiplatform.doc%2Finfo%2Fae%2Fae%2Fcjt1005_.html

Using embedded Derby: If you use embedded Derby as a messaging data store, concurrent access can be a concern. The embedded Derby does not support multiple servers running the Derby engine. Thus, there is no ability to have multiple servers communicating with the same shared file system.

Resources high availability

With WebSphere Application Server V8.0, you can configure failover resources for a data source and connection factory. Resource workload routing improves the availability of the applications. A data source and connection factory can fail over when a default occurs and can fail back when the situation comes back to normal. Only one resource can be used at a time, and the alternate resource is available only when the primary resource fails. To use the alternate resource, you have to create alternate resources for the data source and connection factory. These resources must be identical to the primary resources and must be compatible with applications. Then to finish, add custom properties to configure the availability behavior.

For more details about this topic, refer to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=%2Fcom.ibm.websphere.nd.multiplatform.doc%2Finfo%2Fae%2Fae%2Fcdat_dsfailover.html

Deployment manager high availability

If the deployment manager process fails, the application servers that are managed by the Dmgr process do not fail, and the applications continue to run. Although the Dmgr process has no affects on application availability, consider making this process highly available to be able to use the administrative console and to manage your cell in case a failure occurs.

We present here an active/passive scenario to make the Dmgr process more fault tolerant. This solution is based on shared disks that contain the Dmgr profile. (These disks are accessible by the both active and passive machines.)

The WebSphere binaries are installed on both machines (current and standby). The Dmgr profile is created only one time on a specific file system. The disks of this file system come from a SAN or an NAS and are accessible by the two machines. The profile needs to be install with an IP alias. The failover can be performed by clustering software (such as Power HA) or manually as follows:

- ▶ On the current machine:
 - a. Stop the Dmgr, if it is not already stopped.
 - b. Release the disks.
 - c. Remove the IP.
- ▶ On the standby machine:
 - a. Attach the same disks and retrieve the Dmgr profile.
 - b. Add the same IP.
 - c. Restart the Dmgr process.

After the backup deployment manager starts, you can run it in the same manner as the Dmgr process that failed.

14.3.3 How high availability features work

This section provides information about the WebSphere Application Server components and concepts that facilitate HA.

High availability manager

WebSphere Application Server uses an HA manager to eliminate SPOFs. The HA manager runs key services on available application servers rather than on a dedicated server (such as the deployment manager). It continually polls the core group members to verify that they are active and healthy. The HA manager service runs by default in each server, as shown in Figure 14-16.

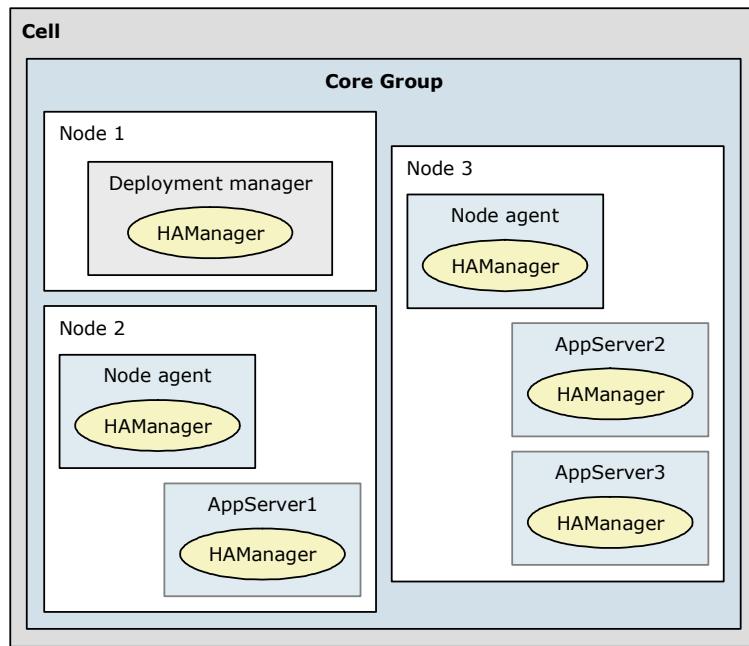


Figure 14-16 Conceptual diagram of a core group

For certain functions (such as transaction peer recovery), the HA manager takes advantage of fault tolerant storage technologies, such as Network Attached Storage (NAS), which lowers the cost and complexity of HA configurations. The HA manager also provides peer-to-peer failover for critical services by maintaining a backup for these services. WebSphere Application Server supports other HA solutions such as Power HA, IBM Parallel Sysplex®, and so on.

An HA manager monitors the application server environment continually. If an application server component fails, the HA manager takes over the in-flight and in-doubt work for the failed server. This process introduces some processing impact, but improves application server availability.

An HA manager focuses on recovery support and scalability in the following areas:

- ▶ Embedded messaging
- ▶ Transaction managers
- ▶ Workload management controllers
- ▶ Application servers
- ▶ WebSphere partitioning facility instances
- ▶ On-demand routing
- ▶ Memory-to-memory replication through Data Replication Service (DRS)
- ▶ Resource adapter management

To provide this focused failover service, the HA manager supervises the JVMs of the application servers that are core group members. The HA manager uses one of the following methods to detect failures:

- ▶ An application server is marked as failed if the socket fails.

This method uses the KEEP_ALIVE function of TCP/IP and is tolerant of poor performing application servers, which might happen if the application server is overloaded, swapping, or thrashing. This method is best for determining a JVM failure if you are using multicast emulation and are running enough JVMs on a single application server to push the application server into extreme processor starvation or memory starvation.

- ▶ A JVM is marked as failed if it stops sending heartbeats for a specified time interval.

This method is referred to as *active failure detection*. When it is used, a JVM sends out one heartbeat, or pulse, at a specific interval. If the JVM does not respond to heartbeats within a defined time frame, it is considered down.

With WebSphere Application Server, you can configure an alternative protocol provider to monitor and manage communication between core group members. In general, alternate protocol providers, such as the z/OS Cross-System Coupling Facility (XCF)-based provider, uses less system resources than the default Discovery Protocol and Failure Detection Protocol, especially during times when the core group members are idle.

In either case, if a JVM fails, the application server on which it is running is separated from the core group, and any services running on that application server are failed over to the surviving core group members.

A JVM can be a node agent, an application server, or a deployment manager. If a JVM fails, any singletons running in that JVM are activated on a peer JVM after the failure is detected. This peer JVM is already running and eliminates the normal startup time, which potentially can be minutes.

This method of detecting failovers actually is a key difference to using operating system-based HA. With this method, the HA manager usually recovers in seconds, and operating system-based solutions take minutes.

When an application server fails, the HA manager assigns the failing application servers work to another eligible application server. Using shared storage for common logging facilities (such as the transaction logs) allows the HA manager to recover in-doubt and in-flight work if a component fails.

Additional resource: The following website provides a testing routine that you can use to determine whether your shared file system is suitable for use with the HA manager:

http://www-01.ibm.com/support/docview.wss?rs=180&context=SSEQTP&ql=transaction+log+failover&uid=swg24010222&loc=en_US&cs=utf-8&lang=en

Core group

A *core group* is an HA domain that consists of a set of processes in the same cell that can establish HA relationships directly. Highly available components can fail over only to another process in the same core group, and replication can occur only between members of the same core group.

A cell must contain at least one core group, although multiple core groups are supported. Each core group contains a core group coordinator to manage its HA relationships and a set of HA policies that are used to manage the highly available components within that core group.

WebSphere Application Server provides one standard core group, the DefaultCoreGroup, that is created during installation. New server instances are added to the default core group as they are created.

In most cases, one core group is sufficient for establishing an HA environment. However, certain topologies require the use of multiple core groups. A basic rule is that all members of a core group require full IP visibility. Therefore, you have to create multiple core groups if you spread the application servers of your cell across different firewall zones.

If you have more than 50 servers in a cell: A large number of application servers in a cell increases the processing impact of core group services and server startup times. Consider creating additional core groups when you have more than 50 servers in a cell.

If you are using a DMZ secure proxy server with dynamic routing, the routing information is exchanged using core groups. In this case, you need to create a tunnel access point group to establish a core group bridge tunnel between the core groups that are running on both sides of the firewall.

The core group contains a bridge service that supports cluster services that span multiple core groups. Core groups are connected by access point groups. A core group access point defines a set of bridge interfaces that resolve IP addresses and ports. It is through this set of bridge interfaces that the core group bridge provides access to a core group.

When moving core group members to new core groups, remember the following information:

- ▶ Each server process within a cell can only be a member of one core group.
- ▶ If a cluster is defined for the cell, all cluster members must belong to the same core group.

Network communication between all members of a core group is essential. The network environment must consist of a fast local area network with full IP visibility and bidirectional communication between all core group members. IP visibility means that each member is entirely receptive to the communications of any other core group member.

High availability groups

HA groups are part of the HA manager framework. An HA group provides the mechanism for building a highly available component and enables the component to run in one of several different processes. An HA group cannot extend beyond the boundaries of a core group.

An HA group is associated with a specific component. The members of the group are the set of processes where it is possible to run that component. A product administrator cannot directly configure or define an HA group and its associated set of members. Instead, HA groups are created dynamically at the request of the components that need to provide a highly available function.

HA groups are dynamically created components of a core group. A core group contains one or more HA groups. However, members of an HA group can also be members of other HA groups, if all of these HA groups are defined within the same core group.

Every HA group has a policy associated with it. This policy is used to determine which members of an HA group are active at a given time. The policies that HA groups use are stored as part of the core group configuration. The same policy can be used by several HA groups, but all of the HA groups to which it applies must be part of the same core group.

Any WebSphere Application Server highly available component can create an HA group for its own usage. The component code must specify the attributes that are used to create the name of the HA group for that component.

For example, establishing an HA group for the transaction manager is achieved by completing the following steps:

1. The code included in the transaction manager component code specifies the attribute type=WAS_TRANSACTIONS as part of the name of the HA group that is associated with this component.
2. The HA manager function includes the default policy Clustered TM Policy that includes type=WAS_TRANSACTIONS as part of its match criteria.
3. Whenever transaction manager code joins an HA group, the HA manager matches the match criteria of the Clustered TM Policy to the HA group member name. In this example, the name-value pair type=WAS_TRANSACTIONS that is included in the HA group name is matched to the same string in the policy match criteria for the Clustered TM Policy. This match associates the Clustered TM Policy with the HA group that was created by the transaction manager component.
4. After a policy is established for an HA group, you can change the policy attributes, such as quorum, fallback, and preferred servers. However, you cannot change the policy type. If you need to change the policy type, create a new policy and then use the match criteria to associate that new policy with the appropriate group.

Match criteria: If you want to use the same match criteria, you must delete the old policy before defining the new policy. You cannot use the same match criteria for two different policies.



Monitoring distributed systems

Being able to measure and monitor system interactions helps IT in providing business continuity. Monitoring capabilities play a key role in successfully managing enterprise systems. In WebSphere Application Server, there are a number of tools that can contribute to an organization's monitoring strategy and provide insights into the performance of the application server.

In this chapter, we provide an introduction to these toolsets.

We cover the following topics:

- ▶ Overview
- ▶ Enabling monitoring infrastructures
- ▶ Viewing the monitoring data
- ▶ Monitoring scenarios
- ▶ IBM Tivoli Composite Application Manager for WebSphere Application Server
- ▶ Additional resources for monitoring

15.1 Overview

IT environments are complex, involving many different servers working together to deliver the electronic functions of business. In a single user interaction, it is typical that information can be retrieved from many systems. Consider the simple distributed WebSphere Application Server environment in Figure 15-1.

The stars in the figure show that even a simple web application request can pass through a whole series of dependent servers to successfully complete a request. JEE is a component based architecture, requiring a request to interact and use 'n' number of these components to complete. Monitoring system components and their performance can become complex, yet are critical to understanding the overall performance of an application.

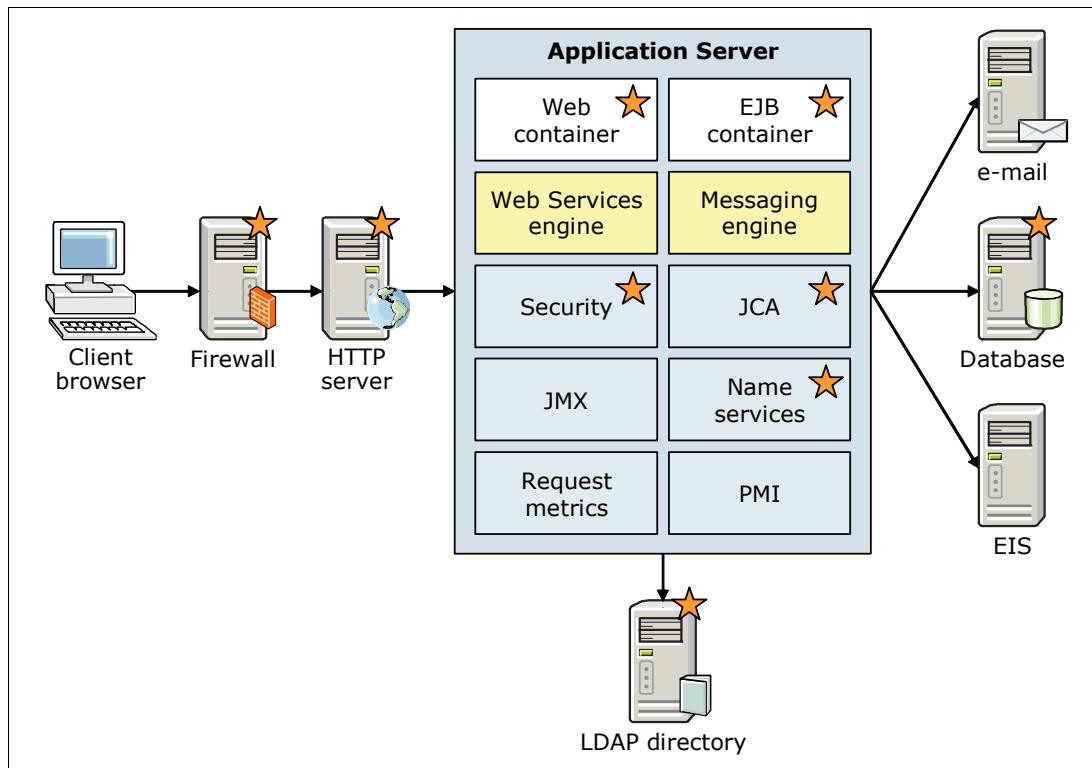


Figure 15-1 Simple web system topology

Monitoring the systems contributes to overall systems management by:

- ▶ Establishing an understanding of the performance baseline and of what runtime behaviors constitute "normal" operations
- ▶ Measuring performance and identifying poorly performing systems and components
- ▶ Identifying service failures, and assisting in root cause identification

WebSphere Application Server monitoring tools rely primarily on information gathered from two core data infrastructures:

- ▶ Performance Monitoring Infrastructure (PMI), which is a collection of statistical agents scattered throughout the application server that gather statistical data on the performance of the application server components. For more information about this topic, go to the following website:
http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.nd.multiplatform.doc/info/ae/ae/cprf_pmidata.html
- ▶ Request metrics, which are primarily a set of timing agents that track a request as it navigates the components of the application server. A key differentiation of request metrics is that they are measured at the request level. The focus of a request metric is to record the time spent by individual requests in different components of the application and at the end of the request provide a record of where time was spent in the request. For more information about this topic, go to the following website:
http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.base.doc/info/aes/ae/tprf_requestmetrics.html

15.1.1 Monitoring scenarios

This chapter demonstrates the use of the system monitoring infrastructures by using different monitoring scenarios, which are summarized in Table 15-1.

Note: These scenarios cover several common uses of the monitoring tools, but it should be understood that many of the different types of data are not explicitly discussed. This chapter provides an introduction to the tool sets and a way for new administrators to get started, and serves as a reminder for experienced administrators about some of the tools that they might not have used in some time.

Table 15-1 Monitoring scenario summary

Monitoring scenario	Chapter reference
How is monitoring data activated and what are the monitoring choices? <ul style="list-style-type: none">▶ PMI data defaults▶ Enabling request metrics	<ul style="list-style-type: none">▶ 15.2, “Enabling monitoring infrastructures” on page 545▶ “PMI defaults and monitoring settings” on page 545▶ “Enabling request metrics” on page 552
What are the tools that I can use for understanding the collected data? <ul style="list-style-type: none">▶ Tivoli Performance Viewer	<ul style="list-style-type: none">▶ 15.3, “Viewing the monitoring data” on page 556
Understanding how application(s) are interacting with a database. This scenario helps identify data that will help answer the following questions: <ul style="list-style-type: none">▶ Is the database responding fast enough?▶ Are there enough connections to the database?▶ Are the connections being returned to the pool?	<ul style="list-style-type: none">▶ 15.4.1, “Database interactions” on page 566

Monitoring scenario	Chapter reference
<p>Understanding JCA connection pool utilization. This scenario helps administrators answer the following questions:</p> <ul style="list-style-type: none"> ▶ What is the response like for the JCA connection pools? ▶ Are there enough connections to support the system interactions? ▶ Are the connections being returned to the pool? 	<ul style="list-style-type: none"> ▶ 15.4.1, “Database interactions” on page 566 ▶ “JCA interactions” on page 567
<p>What about threading resources? Is there sufficient JMS, EJB, and web threads allocated in the server thread pools? This scenario helps administrator understand:</p> <ul style="list-style-type: none"> ▶ How to monitor and improve system related throughput. ▶ Current limits on system concurrency. 	<ul style="list-style-type: none"> ▶ 15.4.2, “Threading resources” on page 567
<p>Monitoring memory allocation and garbage collection. JVM memory tuning is vital to application server performance, and this scenario introduces administrators to tools that assist in making memory tuning choices.</p>	<ul style="list-style-type: none"> ▶ 15.4.3, “JVM memory usage” on page 570
<p>Finding bottlenecks in response time, services, EJB, and web response times. Understanding of component response time helps in the identification of application bottlenecks and performance issues.</p>	<ul style="list-style-type: none"> ▶ 15.4.4, “Request level details” on page 572
<p>Special features of ITCAM for WebSphere in WebSphere Application Server v8.0.</p>	<ul style="list-style-type: none"> ▶ 15.5, “IBM Tivoli Composite Application Manager for WebSphere Application Server” on page 577

The monitoring infrastructure scenarios are demonstrated using the example environment shown in Figure 15-2.

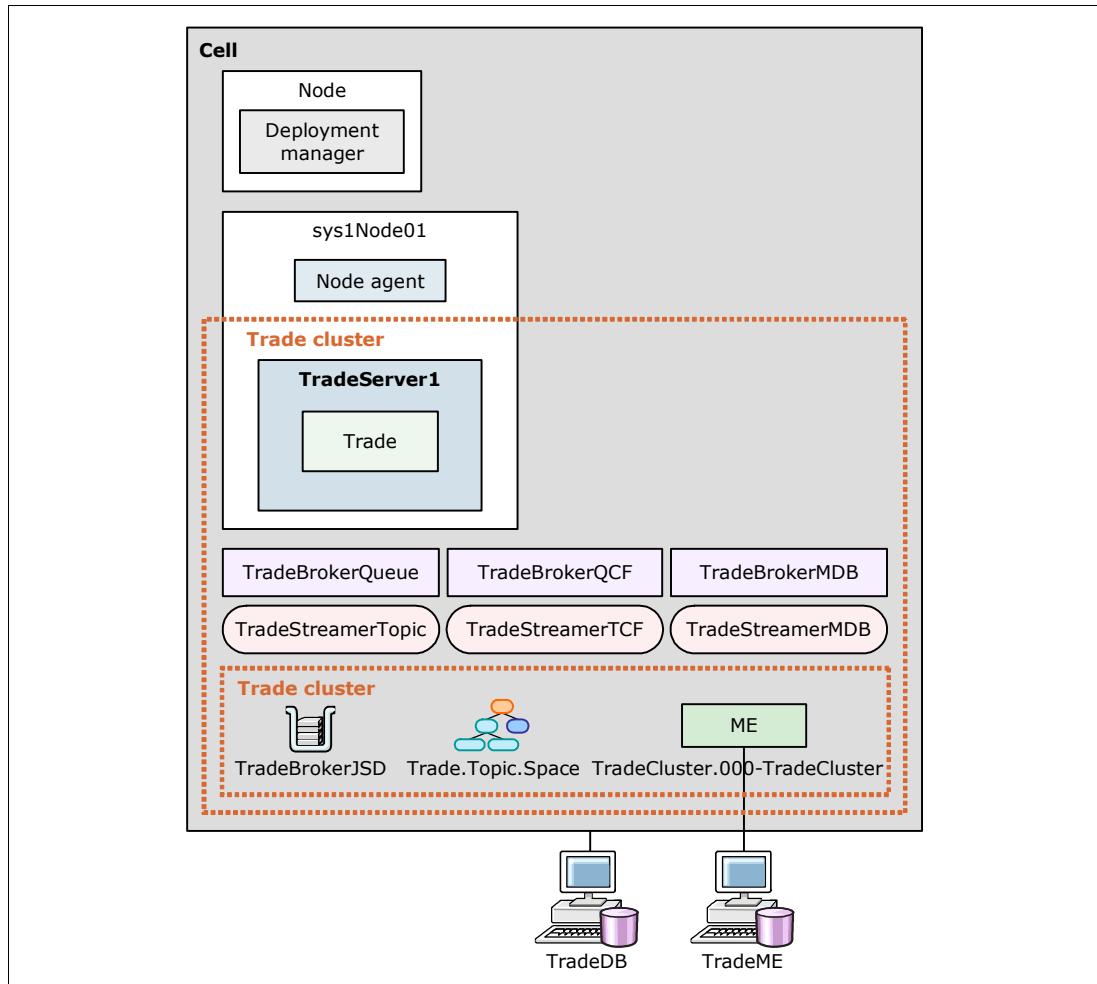


Figure 15-2 Example monitoring environment

Note: We did no special tuning in the test environment. We simply installed the Trade performance application into a base server configuration and changed the default memory.

15.2 Enabling monitoring infrastructures

This section shows you how to enable the PMI monitoring infrastructure and the request metrics that provide the performance data.

15.2.1 PMI defaults and monitoring settings

The enabling of PMI data is managed on a server-by-server basis. In the administrative console, complete the following steps.

1. Click **Monitoring and Tuning → Performance Monitoring Infrastructure (PMI)**.

2. Select the server for which you want to manage the PMI controls. In this example, TradeServer1 is the server. Figure 15-3 shows the PMI configuration window for the server.

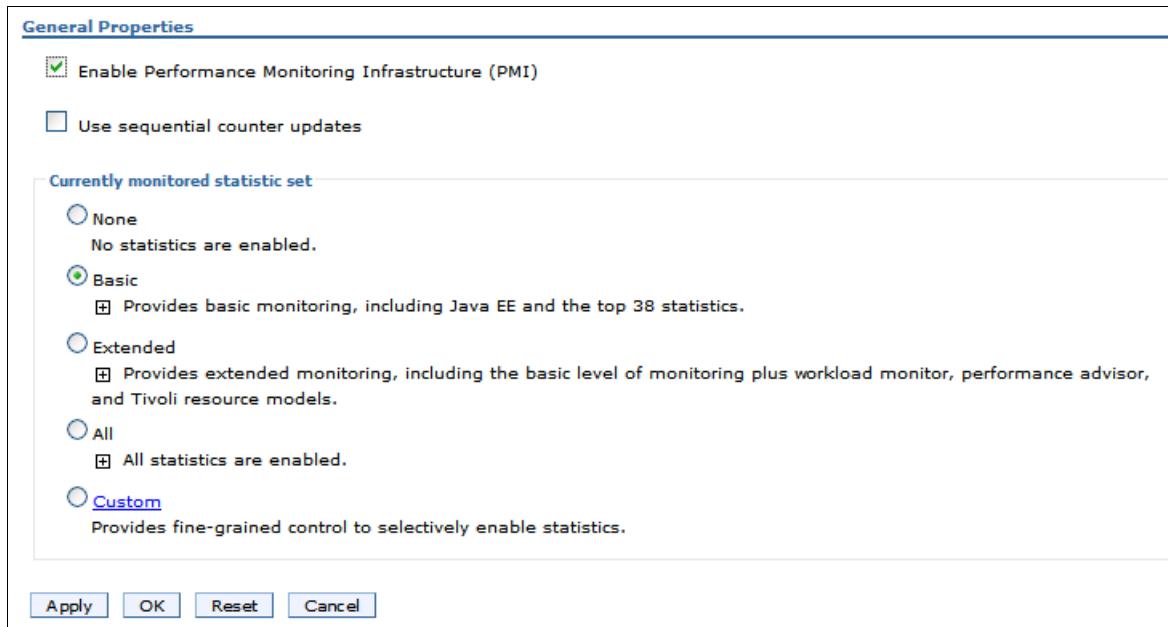


Figure 15-3 Default PMI settings

On this window, note that:

- PMI is enabled by default.
- The default statistical set is the Basic set.

The PMI data can be changed at run time using the settings on the Runtime tab.

Disabling and enabling: Disabling and enabling of PMI data requires a server restart.

The enabling and disabling of PMI is not available on the Runtime tab. However, the monitoring level can be set to None in a server with PMI enabled using the Runtime tab.

Understanding the sets of PMI statistics

The PMI statistic sets represent a group of individual statistical agents. The types of statistics that PMI can collect are classified. Information about these classes can be found in the WebSphere Information Center on the “PMI data classification” page at the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.nd.doc/info/ae/ae/rprf_dataclass.html

Everyone working in system administration knows that every action executed in an environment has a cost. Monitoring is no different, and for PMI, the cost of monitoring is impacted primarily by two factors:

- ▶ The amount of data that is monitored.
- ▶ The impact of individual performance metrics. Not all metrics have the same collection cost.

With PMI, there are multiple sets of statistics that can be enabled, as shown in Figure 15-3 on page 546. These sets of statistics are:

- ▶ None
- ▶ Basic
- ▶ Extended
- ▶ All
- ▶ Custom

None and All are self explanatory, so here we take a closer look at the options provided by Basic, Extended, and Custom.

Basic statistic set

The Basic statistic set is the default setting. The basic setting is configured with the intention of providing an overall understanding of application server health, including statistics as outlined in the JEE specification, as well as other common performance hotspots and key monitoring points for JEE applications. Later, we discuss how to determine the impact and level of a statistic (see “Getting more information about statistics sets” on page 550).

Figure 15-4 shows the list of PMI counters that are active for the basic PMI data level. For details about each counter, refer to “Getting more information about statistics sets” on page 550.

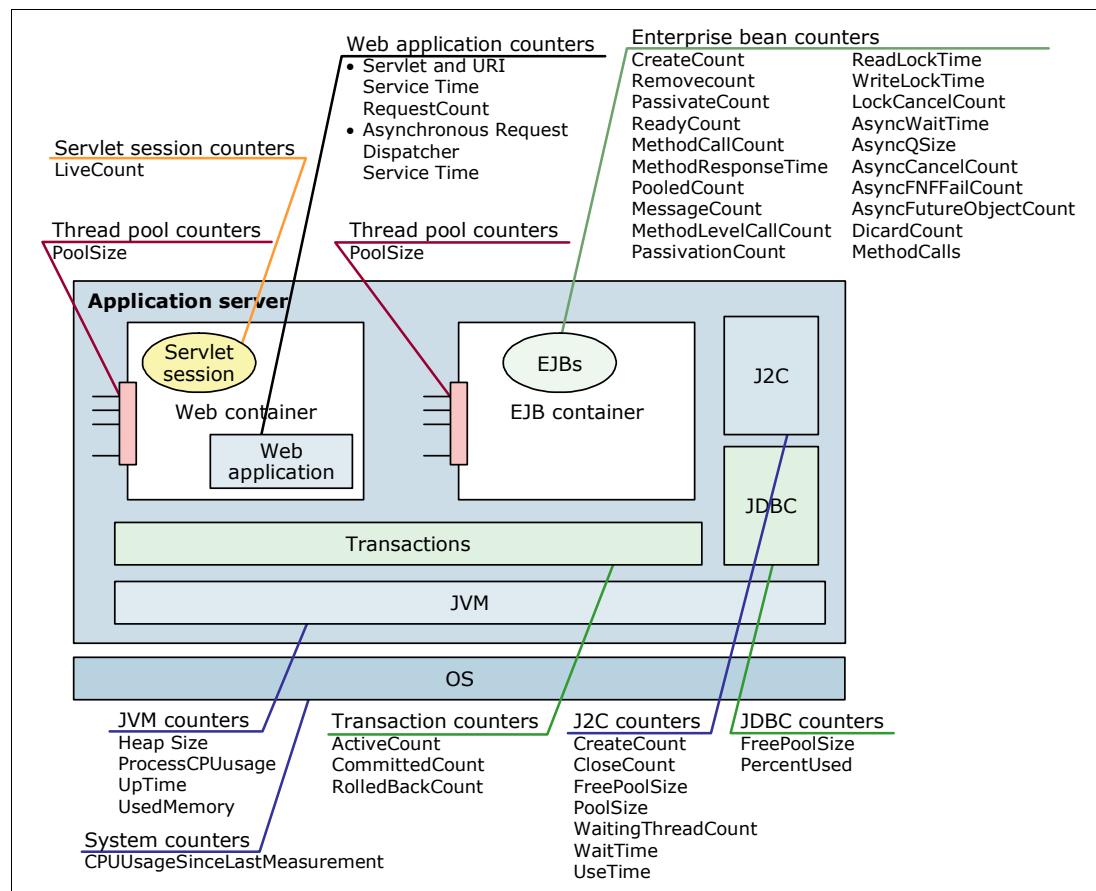


Figure 15-4 PMI basic counters

Extended statistic set

The extended PMI data set has the basic set and some additional statistics with a particular emphasis on statistics that look at the load on the server and the servers response to the load being applied. The statistical agents in the extended set might or might not apply to a JEE application depending on the individual application architecture and environment configurations.

Figure 15-5 shows the extended metrics that are in addition to those of the basic configuration.

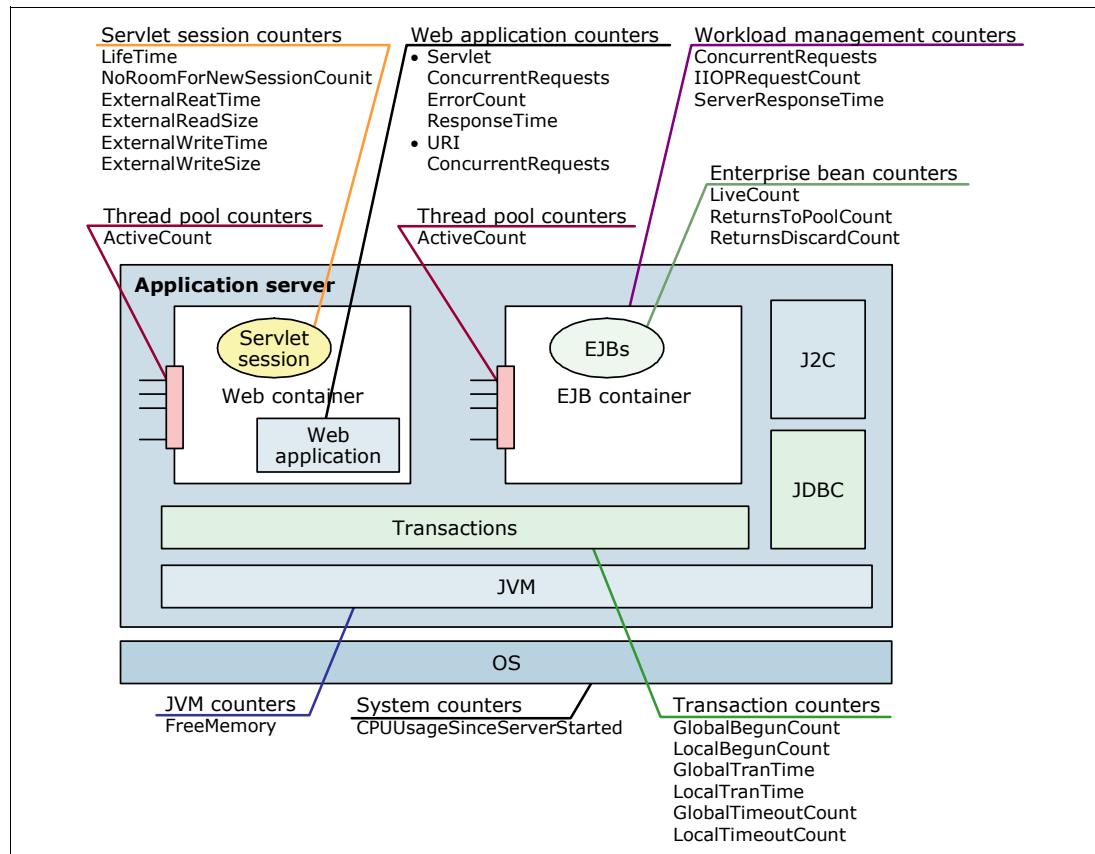


Figure 15-5 PMI extended counters

Custom statistic set

The Custom PMI data collection set allows the administrator to choose the counters that are most appropriate for the application(s) that are deployed on the server. Each counter is individually activated. This is the most powerful configuration, but requires that the administrator spend some time reviewing the available statistical counters and also that the administrator understands the type of counter that is useful for the applications.

For example, consider the counters activated for `ServletSession` if the extended data set is selected. The counters are:

- ▶ LiveCount
- ▶ LifeTime
- ▶ NoRoomForNewSessionCount
- ▶ ExternalReadTime
- ▶ ExternalReadSize

- ▶ ExternalWriteTime
- ▶ ExternalWriteSize

The NoRoomForNewSessionCount counter only applies if the Allow overflow from the web container session management was changed from its default value of true. This attribute is shown in Figure 15-6.

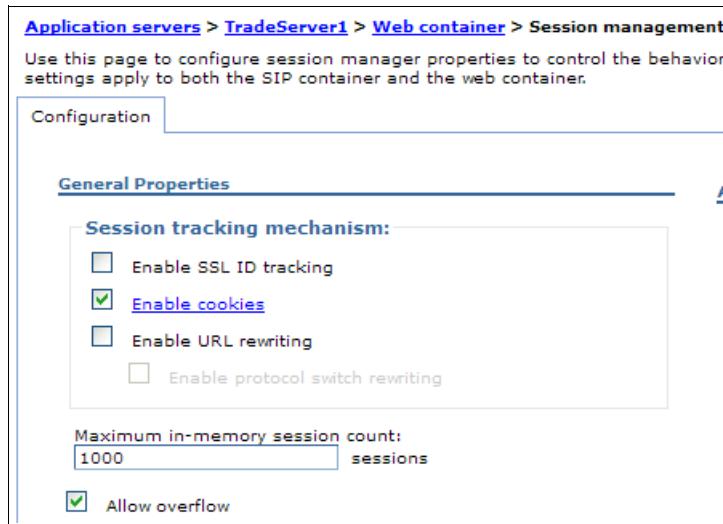


Figure 15-6 Allow overflow default is true

Similarly, the counters related to external session management only apply if session persistence is configured. Hence, the activation of the extended session information does little for an application where the overflow option is not modified and session persistence is not configured.

With a custom metric approach, the administrator could choose to simply add LiveCount and LifeTime counters, as well as perhaps choosing other metrics of interest, such as the TimeoutInvalidationCount, to measure how many sessions are being timed out rather than logged off.

Tip for using custom PMI settings: The counters used for the basic set can be customized to form a baseline for the custom counter activation. They should be supplemented with additional counters that are relevant for the application types that are being deployed.

Impact of PMI

The actual impact of each statistic level varies depending on the particular applications on the server and load that is being executed by the server. In the WebSphere Information Center, each counter has a documented qualitative impact level to indicate the type of impact it will incur (see “Getting more information about statistics sets” on page 550). This is not intended to prevent administrators from using counters with high impact. It is important to remember that the impact is a relative measurement, and the administrator needs to balance the need for the data versus the impact incurred to enable a particular counter temporarily or for the long term.

The approximate impact of the PMI statistic sets are as follows:

- ▶ Basic impact up to 2%.
- ▶ Extended impact up to 3%.

- ▶ All impact of up to 6%.
- ▶ Custom will depend on the counters enabled, but it is reasonable to expect somewhere between 2% - 6%.

Getting more information about statistics sets

The WebSphere Information Center has extensive information to assist the administrator in understanding exactly which metrics are set for a particular level, and to appreciate the potential impacts of using the statistics.

If we consider the number of components that make up an application server, as shown in Figure 15-7, it should be no surprise that there are many PMI counters available to help monitor the application server.

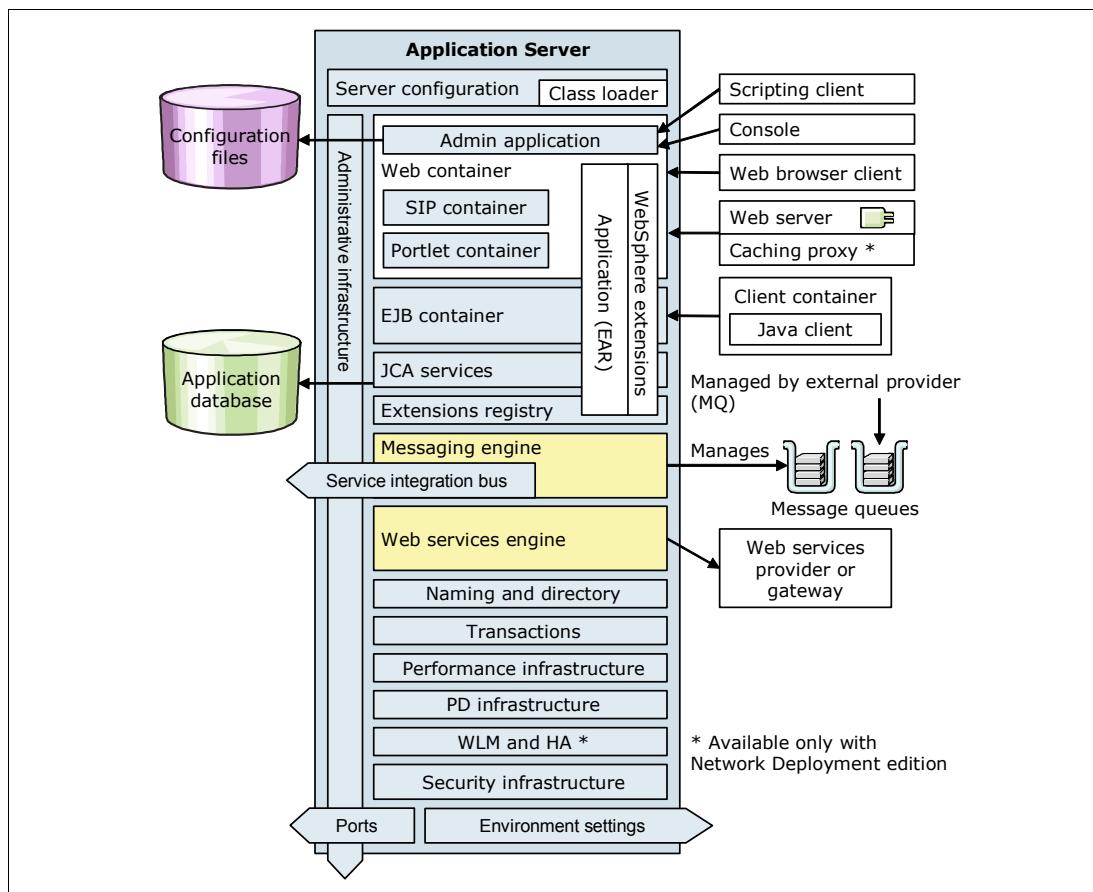


Figure 15-7 Application server components

The WebSphere Information Center provides a summary of PMI counters to help administrators understand the variety of counters that are available in each of the different counter classifications in the application server. A good place to start with gathering information is the article “Enabling PMI data collection” at the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.nd.doc/info/ae/ae/tprf_pmi_encll.html

Figure 15-8 shows the links found at the bottom of this article, taking you to a page with more information about each counter.

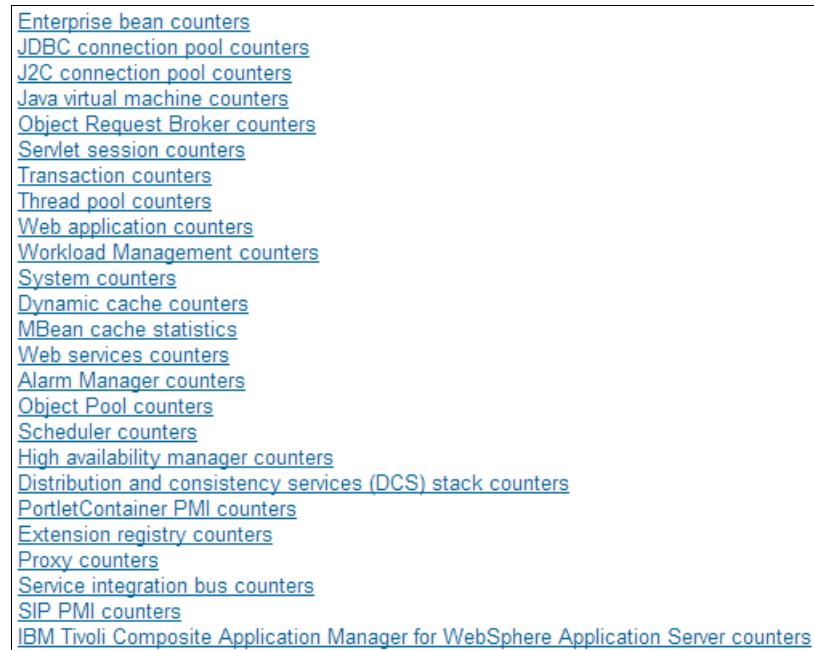


Figure 15-8 PMI counter types

From the links to the counter classifications, it is possible to navigate and view the individual counters that each counter classification contains.

The following topics in the Information Center can provide more information:

- ▶ General PMI data organization, which can be found at the following website:
http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.express.doc/info/exp/ae/rprf_dataorg.html
- ▶ WebSphere Application Server supports the Eclipse framework for extensible applications. A key part of this framework is the implementation of the Extension registry. These counters are only relevant when referring to extensible applications. For more information, go to the following website:
http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.base.iseries.doc/info/iseries/ae/cweb_extensions.html
- ▶ Service integration bus counters, which can be found at the following website:
http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.express.doc/info/exp/ae/rprf_sibcounter.html
- ▶ Proxy counters, which can be found at the following website:
http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=%2Fcom.ibm.websphere.nd.doc%2Finfo%2Fae%2Frprf_proxycounter.html

15.2.2 Enabling request metrics

The enabling of request metrics is a cell wide configuration and when activated, it is activated for all servers in the cell. Complete the following steps to enable request metrics.

1. In the administrative console, click **Monitoring and Tuning** → **Request Metrics** (Figure 15-9).

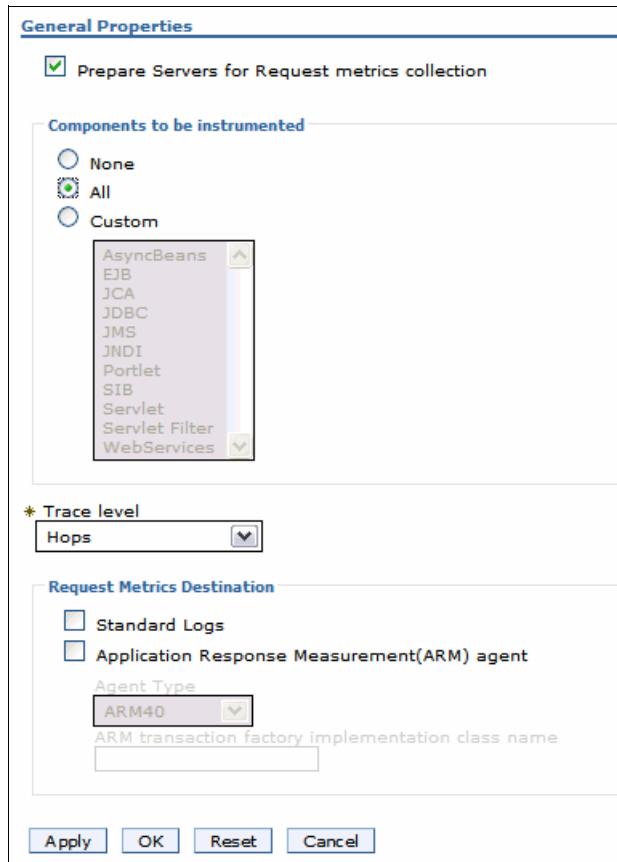


Figure 15-9 Request metrics window

Request metrics are enabled by:

- a. Selecting **Prepare servers for request metrics collection**.
- b. Choosing a monitoring level from the Components to be instrumented section of the window.
- c. Choosing a trace level.
- d. Choosing a destination from the Request Metrics destination section of the Request Metrics window.

When configured, the servers must be restarted for request metrics to be enabled. The servers must also be stopped when disabling request metrics.

Understanding component instrumentation and trace levels

Trace levels and component instrumentation work together to determine if the request is instrumented. The component instrumentation levels are shown in Figure 15-10.

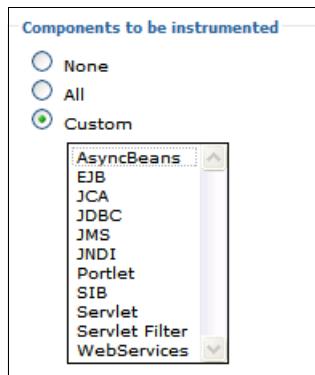


Figure 15-10 Components to be instrumented

If you select **All**, all components will be monitored based on trace level settings.

If you select **Custom**, you can select the components to be monitored. Data will be collected from the components if the trace level also calls for the capturing of data from this component.

Note: When a component is defined as an edge component, meaning the request enters or exits the application server through the component, then this component is instrumented even if it is not selected as part of the custom component listing.

Working in conjunction with the component instrumentation levels is the trace level (Figure 15-11).

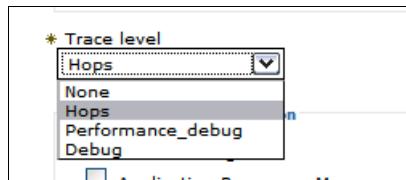


Figure 15-11 Request metric trace levels

The following trace levels are possible:

- ▶ **None:** No instrumentation is generated.
- ▶ **Hops:** Generates instrumentation information about process boundaries only. When this setting is selected, you see the data at the application server level, not the level of individual components, such as enterprise beans or servlets.
- ▶ **Performance_debug:** Generates the data at Hops level and the first level of the intra-process servlet and Enterprise JavaBeans (EJB) call (for example, when an inbound servlet forwards to a servlet and an inbound EJB calls another EJB). Other intra-process calls like naming and service integration bus (SIB) are not enabled at this level.
- ▶ **Debug:** Provides detailed instrumentation data, including response times for all intra-process calls. Note that requests to servlet filters will only be instrumented at this level.

Note: Information about working with instrumentation and trace levels is provided in 15.4.4, “Request level details” on page 572.

Important: Request metrics are checked starting with the HTTP plug-in for some web related settings. The HTTP-plug-in configuration must be regenerated and propagated after enabling request metrics.

Using request metric filters

One final way that can be used to control the request metric instrumentation is to use request metric filters. Filters provide a way to specifically target flows and components to reduce the impact of broad monitoring and to also make it easier to analyze the captured data by reducing the amount data that is captured.

It is important to understand, however, that filters are implemented as edge component filtering, not as intra-component processing, so an EJB filter will not be effective if the EJB is always invoked from a servlet. In this case, it is the URI that needs filtering, not the EJB. Filters should be applied on edge components.

Filters are configured by selecting the **Filters** link from the Additional properties section of the Request metrics window. This action opens the Request Metrics Filter window shown in Figure 15-12.

The screenshot shows the 'Request Metrics > Request Metrics Filter' window. It includes a note about filtering and a 'Preferences' section. Below is a table with columns for Type and Enable status:

Type	Enable
EJB	false
JMS	false
SOURCE_IP	false
URI	false
WEB SERVICES	false

Total 5

Figure 15-12 Request Metrics Filter window

Using filters, fine-grained controls can be applied to the different edge types of EJB, JMS, IP address, URI, and web services. The first step is to specify the filter. An example of each type can be seen by using the administration console to view that type of filter. For example, selecting the **URI** link in Figure 15-12 takes you to the URI window shown in Figure 15-13.

The screenshot shows the 'General Properties' tab of the 'URI' window. It has fields for Type (set to URI), Enable checkbox, and buttons for Apply, OK, Reset, and Cancel. The 'Additional Properties' tab is visible on the right.

Figure 15-13 URI window

The Enable check box must be selected for the filters to be enabled. Then the filters will be used along with the component and trace level settings to determine which components are instrumented.

Note: Enabling filters requires an application server restart.

Select the **Filter Values** link in the Filter window to add or edit filters. This window is also where the default example filters are displayed (Figure 15-14).

The screenshot shows a software interface for managing filter values. At the top, it says "Request Metrics > Request Metrics Filter > URI > Filter Values". Below this, a sub-header reads "Specifies the value of request metrics filter and enablement for the filter type." There is a "Preferences" button. A toolbar with icons for New, Delete, and other operations is visible. A table lists filter entries with columns for Select, Value, and Enable filter. The table shows two entries: "/hitcount" and "/snoop", both set to false. A summary at the bottom indicates "Total 2".

Select	Value	Enable filter
<input type="checkbox"/>	/hitcount	false
<input type="checkbox"/>	/snoop	false

Figure 15-14 Default filter values displayed in filter value window

Each filter type has its own syntax that is appropriate for the type. For example, the EJB filter specifies a method class or package that sets the scope of the filtering. Figure 15-15 shows the example URI filter value supplied for EJB filters.

The screenshot shows a software interface for managing filter values. At the top, it says "Request Metrics > Request Metrics Filter > EJB > Filter Values". Below this, a sub-header reads "Specifies the value of request metrics filter and enablement for the filter type." There is a "Preferences" button. A toolbar with icons for New, Delete, and other operations is visible. A table lists filter entries with columns for Select, Value, and Enable filter. The table shows two entries: "com.yourco.package.Class.method" and "com.yourco.package.Class2.*", both set to false. A summary at the bottom indicates "Total 2".

Select	Value	Enable filter
<input type="checkbox"/>	com.yourco.package.Class.method	false
<input type="checkbox"/>	com.yourco.package.Class2.*	false

Figure 15-15 EJB default filter values

It is possible to use wildcard settings in the filters if desired, as shown in the second entry in Figure 15-15.

Tip: Enabling / disabling a filter requires restarting of the server. It is important to plan the component levels and filters that an application might require to minimize the need to stop and restart servers.

Destination type considerations

The final consideration when configuring the request metrics is where the metrics will be gathered. There are two types of supported destinations:

- ▶ Data can be logged with standard logs. In this configuration, the instrumented components are logged to the SystemOut.log file.
- ▶ Data can also be collated in an Application Response Measurement (ARM) data collector. In this case, the data is normally then moved to a monitoring system for analysis and display (for example, using IBM Tivoli Composite Application Manager for Transactions).

Tip: When configuring ARM agents for use with the application server, follow the installation instructions provided with the specific agent. For more information about ARM agents, see the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.nd.doc/info/ae/ae/cprf_arm.html

Both logging types can be activated at once. Writing to standard logs is not a best practice as a long term monitoring strategy because the impact can be higher than is desirable.

Impact of request metrics

The impact of request metrics can vary significantly based on the components being monitored and the complexity of the request execution within the monitored components. There are no specific metrics on what the impact is, but it is reasonable to assume that request metrics on every request and component might incur more impact than is desired. An organization must consider and plan carefully the interactions that it wants to monitor, then measure the specific impact associated with configuring request metrics for these components.

15.3 Viewing the monitoring data

WebSphere Application Server provides an interface for viewing the monitored data. The interface is the Tivoli Performance Viewer (TPV) found in the administrative console.

15.3.1 Starting TPV monitoring and configuring settings

To work with the TPV from the administrative console, complete the following steps:

1. Click **Monitoring and Tuning** → **Performance Viewer** → **Current Activity**.

2. Select the server(s) that are to be monitored and click the **Start Monitoring** button (Figure 15-16).

Tip: If you are only starting monitoring on a single server, monitoring can be started by simply clicking the server link and navigating to the TPV viewer.

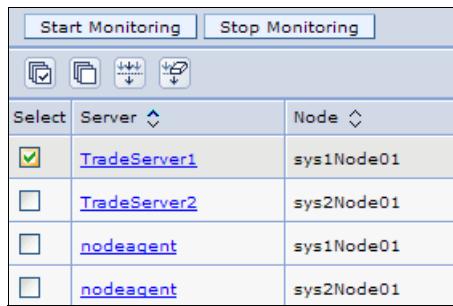


Figure 15-16 Start Server monitoring

After monitoring is started, a message will be returned in the messages section of the window, and the Status column of the server will be updated to *Monitored*.

3. The PMI data can only be observed one server at a time when using a single user session. Select the server name link to navigate to the Tivoli Performance Viewer window (Figure 15-17).

Name	Application	Total Requests	Avg Resp Time (ms)	Total Time (ms)
/account.jsp	DayTrader-EE6#web.war	2	0	0
/displayQuote.jsp	DayTrader-EE6#web.war	30	6.7	201
/marketSummary.jsp	DayTrader-EE6#web.war	2	164.5	329
/order.jsp	DayTrader-EE6#web.war	5	3.2	16
/portfolio.jsp	DayTrader-EE6#web.war	8	1.875	15
/quote.jsp	DayTrader-EE6#web.war	6	75.167	451
/register.jsp	DayTrader-EE6#web.war	1	0	0
/tradehome.jsp	DayTrader-EE6#web.war	2	297	594
/welcome.jsp	DayTrader-EE6#web.war	1	16	16
TradeAppServlet	DayTrader-EE6#web.war	24	510.167	12,244
rsp servlet	ibmasyncrsp.war	0	0	0

Figure 15-17 Tivoli Performance Viewer

The default view for this window shows the Servlet Summary Report pane, which indicates recent servlet activity, as well as the TPV tree navigation pane.

Note: Information about the different ways that data can be viewed is provided in 15.3.2, “Exploring Tivoli Performance Viewer data views” on page 560, but before exploring the data, let us first take a look at the settings menu to examine the User and Logging settings.

- The user settings are reached by expanding the **Settings** category and selecting **User**. This window helps you control how much data is retained and how often data samples are taken in the live system (Figure 15-18).

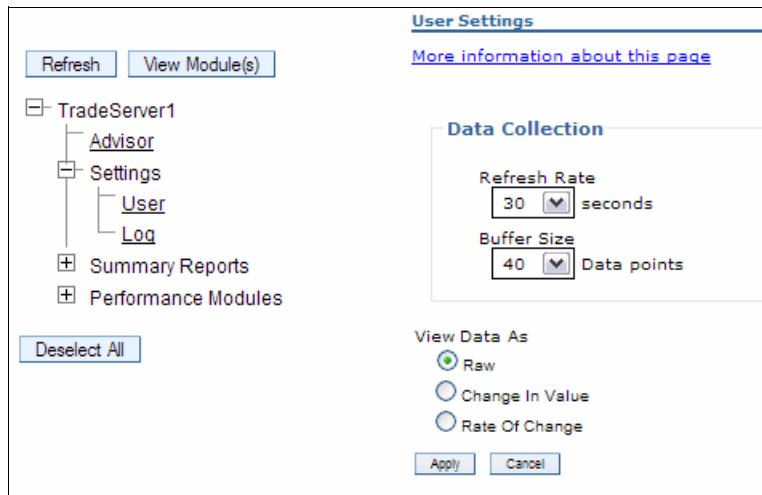


Figure 15-18 User settings

The user settings are significant and can have a direct impact on the performance of the server. The two key configuration settings are in the Data Collection section of the window:

- The refresh rate indicates the interval between data sampling. Higher frequency rates mean that the server will gather and report on statistics more frequently, adding load to the servers that are collecting data.
- The buffer size indicates the number of data points that are kept. More data points simply mean that the PMI data will require more memory.

For a stand-alone server, this means that PMI data at high frequency and high buffer re-initiation will need more processing time and more memory.

In a distributed server model, the load is shared, but some settings might still need to be tuned. The data is collected at the node level and stored in memory on the node agent. Thus, if a node has many servers, the memory requirements of the node agent will need to be adjusted.

Also, in a distributed server configuration, the data is viewed from the deployment manager. Thus, to process the data, the deployment manager should have adequate memory and CPU resources. Consider that more than one administrator might be observing data at once.

- A powerful feature of TPV is the ability to record PMI data and then replay the data later in a different deployment manager as though it were in real time or with options to fast forward and rewind.

Logging is started by clicking the **Start Logging** button shown in the TPV viewing window (Figure 15-17 on page 557). However, before clicking this button, the Log settings must be configured. The Log settings are found by clicking **Settings → Log**, as shown in Figure 15-19.

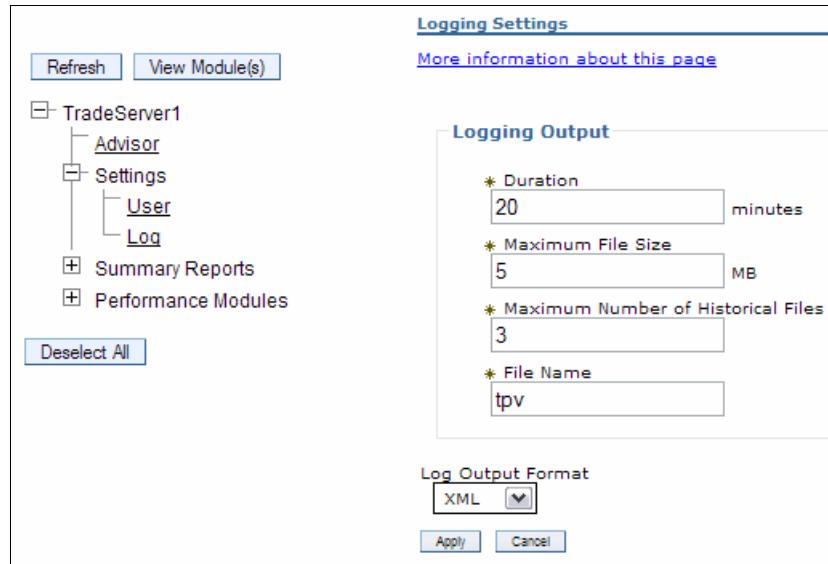


Figure 15-19 Log settings

Examining the log settings that are available, the administrator is faced with several configuration choices:

- Duration:

The logging of PMI data has with it a certain amount of impact (resulting from logging to a file, the buffering of data in memory, and disk usage for log storage). It is not intended to be used as a long term production monitoring strategy. Thus, when logging is enabled, it is configured to be disabled after a period of time.

PMI data logging used in short durations can be used to capture runtime characteristics that might need further investigation or sharing with development and troubleshooting specialists.

- Maximum file size and Maximum number of historical files:

The settings for maximum file size and the number kept that are appropriate for your environment depend on two conditions:

- How much PMI data is enabled.
- The data sampling frequency (as configured in the user settings).

- File name:

The server name and the time at which the log is started is appended to the file name specified to help users identify a log file.

- Log output format:

The other configuration item of consequence is the format type. The default is XML, but the binary format requires a smaller footprint on the disk. If logging larger amounts of data, the binary logging format might be more suitable.

15.3.2 Exploring Tivoli Performance Viewer data views

Tivoli Performance Viewer has three primary types of data that can be viewed:

- ▶ Summary reports
- ▶ Performance modules
- ▶ Advisors

Summary reports

The summary reports provide a general overview in a tabulated format of the current system performance. The reports that are available include:

- ▶ Servlets
- ▶ EJBs
- ▶ EJB Methods
- ▶ Connection Pool
- ▶ Thread Pool

Servlet and EJB summary reports can be useful for identifying the application resources that are the most busy, and to the extent that averages can be used to identify candidates that might warrant further investigation as to their performance. Together with request metrics, results from the summary reports can help identify possible candidates for request metric filters.

The information for Connection Pool and Thread Pool utilization, while useful, can also be easily determined by monitoring the metrics in the performance modules.

Note: For summary reports to be available, the PMI data must be reported at a sufficiently detailed level. For the basic PMI data level, only the Servlets and EJBs reports are available. Higher or custom PMI settings must be specified for the other reports.

Figure 15-20 shows an example of the Servlets report.

Servlets Summary Report				
More information about this page				
<input type="button" value="Start Logging"/>  				
Name	Application	Total Requests	Avg Resp Time (ms)	Total Time (ms)
/PingJsp.jsp	DayTrader-EE6#web.war	5	3.2	16
/PingJspEL.jsp	DayTrader-EE6#web.war	4	31	124
/error.jsp	DayTrader-EE6#web.war	7	6.714	47
/register.jsp	DayTrader-EE6#web.war	1	0	0
/welcome.jsp	DayTrader-EE6#web.war	5	3.2	16
ExplicitGC	DayTrader-EE6#web.war	4	273	1,092
PingServlet	DayTrader-EE6#web.war	3	0	0
PingServlet2Include	DayTrader-EE6#web.war	2	7.5	15
PingServlet2IncludeRcv	DayTrader-EE6#web.war	2	0	0
PingServlet2Servlet	DayTrader-EE6#web.war	1	15	15
PingServlet2ServletRcv	DayTrader-EE6#web.war	1	0	0
PingServletWriter	DayTrader-EE6#web.war	1	0	0
TradeAppServlet	DayTrader-EE6#web.war	10	590.6	5,906
ejb3.PingServlet2Session	DayTrader-EE6#web.war	5	3.2	16
ejb3.PingServlet2TwoPhase	DayTrader-EE6#web.war	1	1,047	1,047

Figure 15-20 Servlets summary report

Figure 15-21 shows an example of the EJBs summary report.

EJBs Summary Report				
More information about this page				
<input type="button" value="Start Logging"/>  				
Name	Application	Method Calls	Avg Resp Time (ms)	Total Time (ms)
DTBroker3MDB	DayTrader-EE6#dt-ejb.jar	0	0	0
DTStreamer3MDB	DayTrader-EE6#dt-ejb.jar	0	0	0
TradeSLSBBean	DayTrader-EE6#dt-ejb.jar	15	959.467	14,392

Figure 15-21 EJB summary

Tip: This summary data can be used to identify EJBs that are invoked often. If the system is working correctly, the slowest bean, on average, might be worth investigating if the time exceeds expected SLAs, and so on.

It can also be useful to identify what work is not happening. In Figure 15-21, there are two message-driven beans that have not been invoked. This situation, in itself, might be unusual and warrant investigation.

Performance modules

Performance modules provide a tracking mechanism for each of the PMI counters that are active. These counters are categorized under their different PMI data classifications. The data can be viewed as a table or graphically displayed. Unlike Version 7, WebSphere now uses dojo technology for plotting the data instead of Scalable Vector Graphics (SVG); the dojo technology provides a better user experience and is more memory and processor efficient.

The performance modules provide a powerful runtime view of the data as it is being recorded to allow the administrator to analyze the current system health.

The data that can be displayed is limited depending on the PMI level that has been configured. The administrator can select one or more metrics for the current PMI level, as shown in Figure 15-22, and then click the **View Modules** button at the top of the window.

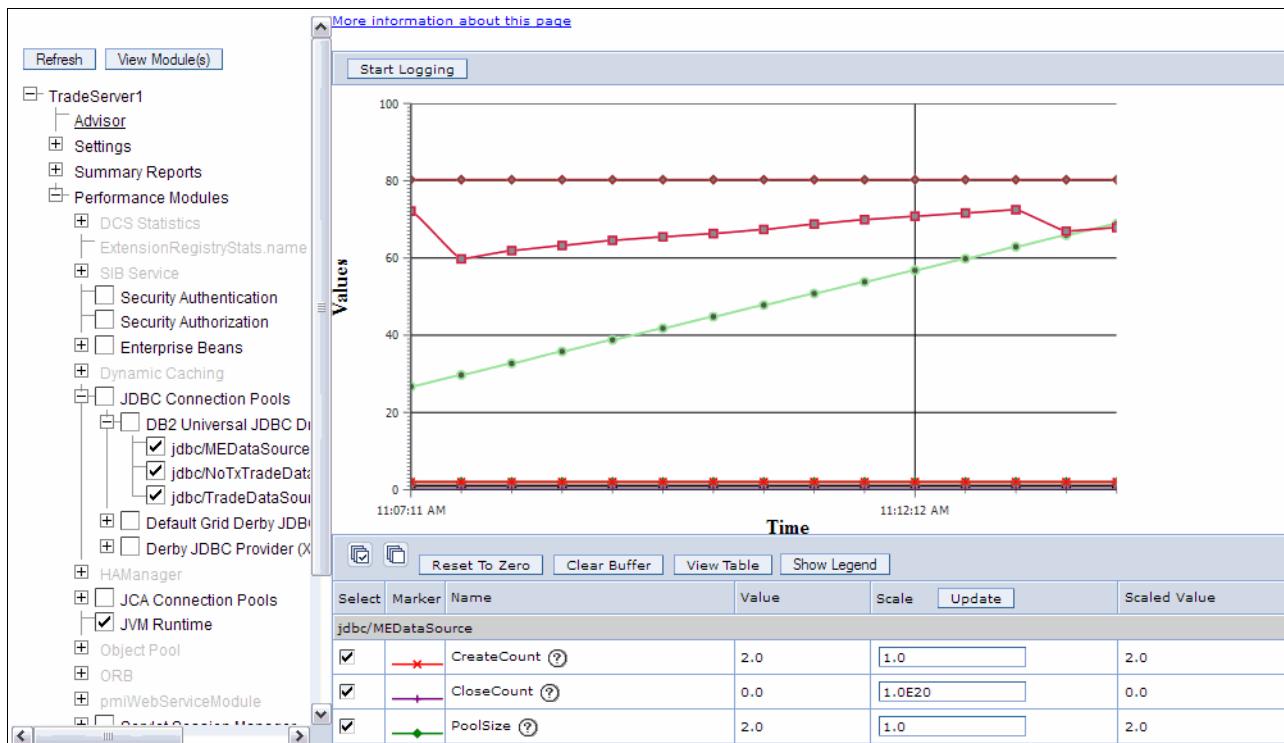


Figure 15-22 Performance modules

After the performance modules are selected, the data window provides a view of the data that is being collected. By default, the graphical view is used. The most recent data point and the graph key for the different counters can be seen under the graph. The graph can then be customized to include or exclude counters from the chosen set of PMI data. To view the data in table format, as shown on Figure 15-23, click the **View table** button. Scales can also be adjusted by changing the scale and clicking the **Update** button.

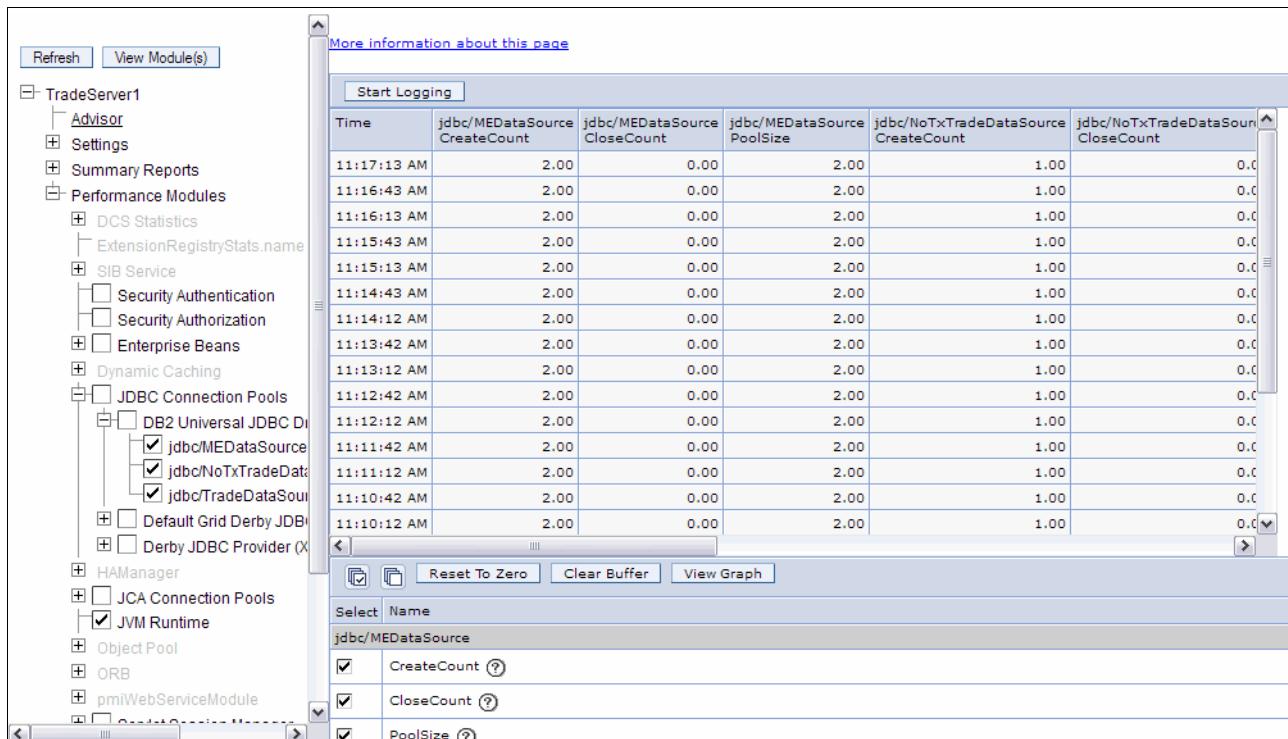


Figure 15-23 Table view

Performance advisors

The last of the TPV data sets contains the TPV performance advisors. These advisors analyze the data using rules that are pre-configured by IBM based on best practice and performance observations. The advisors provide tuning best practices to help improve the performance.

The types of items that TPV will provide advice about include several well known performance hot spots:

- ▶ Object Request Broker service thread pools
- ▶ Web container thread pools
- ▶ Connection pool size
- ▶ Persisted session size and time
- ▶ Data source statement cache size
- ▶ Session cache size
- ▶ Dynamic cache size
- ▶ Java virtual machine heap size
- ▶ DB2 Performance Configuration wizard
- ▶ Connection use violations

Advisors are more of a tuning aid than a monitoring tool set and are not suitable for use in production environments. But advisors can be a useful aid in identifying well known performance hotspots in the current server configurations in testing.

Advisors are best used when:

- ▶ A reasonable load can be driven to the application server utilizing significant CPU (50+%).
- ▶ You want help in tuning a server while establishing initial performance benchmarks.

For more information about the advisors, see the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.nd.doc/info/ae/ae/cprf_whyuseperfadvisors.html

To view the Advisors window, click the **Advisor** link in the TPV menu window. Figure 15-24 shows an example of the Advisors window.

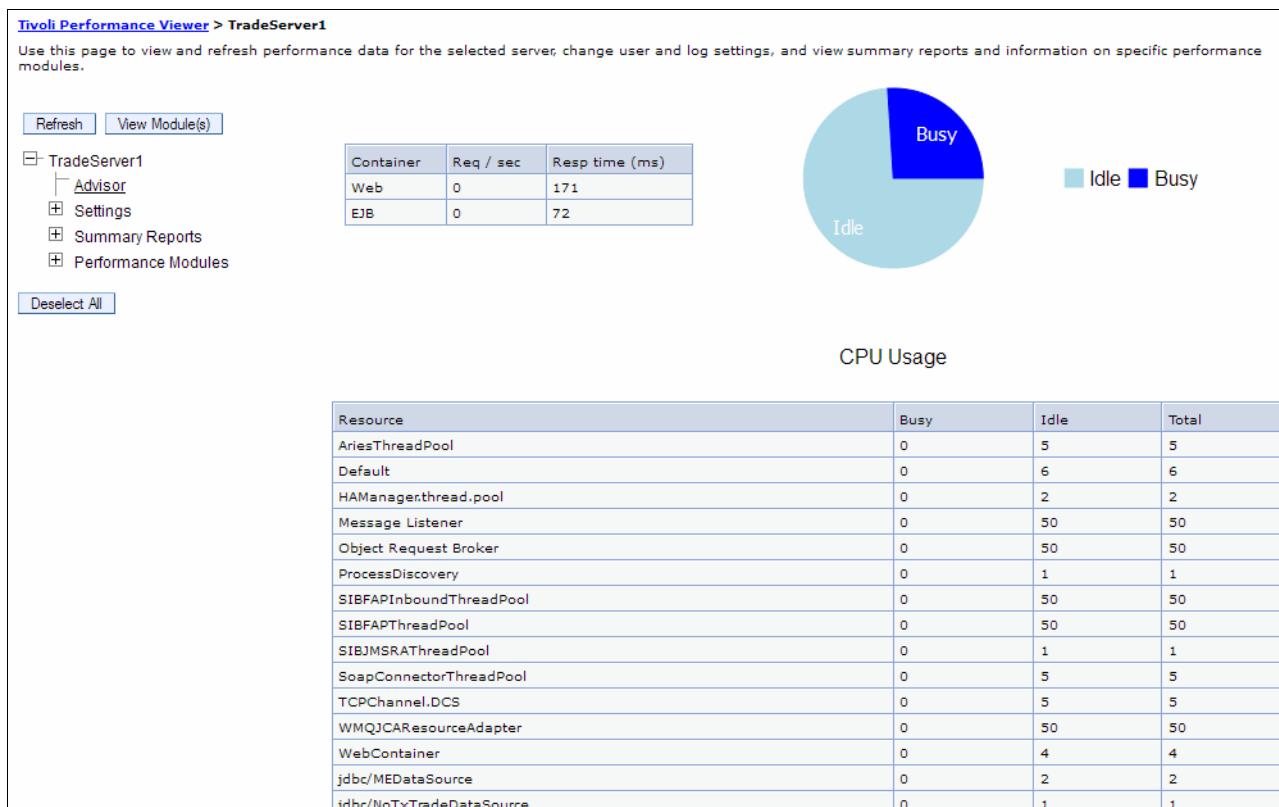


Figure 15-24 Advisors view

At the lower part of the window, the advisor provides advice to the administrator, as shown on Figure 15-25.

The screenshot shows a software interface titled 'Advises view'. At the top, there are buttons for 'Refresh All Advice' and 'Remove Selected Advice'. Below the buttons is a toolbar with icons for refresh, search, and other functions. The main area is a table with columns: 'Select', 'Severity', 'Message', and 'Status'. There are five rows of data:

Select	Severity	Message	Status
<input type="checkbox"/>	Alert	TUNE0221W: Data for memory session ...	Unread
<input type="checkbox"/>	Config	TUNE5003W: The JVM maximum heap siz...	Unread
<input type="checkbox"/>	Config	TUNE5012W: The size of the minimum ...	Unread
<input type="checkbox"/>	Config	TUNE5033W: For data source jdbc/Tra...	Unread
<input type="checkbox"/>	Config	TUNE5033W: For data source jdbc/NoT...	Unread

At the bottom left, it says 'Page: 1 of 4' and 'Total 18'. On the right side of the bottom bar is a small icon.

Figure 15-25 Advises view

From within the advisors view, the different advice statements can be selected and further analyzed by the administrator. Figure 15-26 shows an example of an advice window.

The screenshot shows a detailed view of an advice window. It has several sections:

- Message:** TUNE5033W: For data source jdbc/TradeDataSource, the maximum connection pool size is unusually large. Typically, the connection pool size is no more than 30.
- Severity:** Config
- Description:** In general, a very large connection pool reduces performance, although it might be necessary for some applications.
- User Action:** To change the connection pool properties, open the administrative console and click JDBC Providers > JDBC_provider > Data Sources > data_source > Connection pool properties. See the information center for more information about queueing.
- Detail:** Currently, the size of the minimum connection pool is 10 and the maximum pool size is 50.

At the bottom left is a 'Back' button.

Figure 15-26 Advice window

The Advice window provides a clear description and recommendation to aid the administrator in quickly addressing well understood performance considerations.

Note: More information about advisors is not provided in this chapter. Our focus in this chapter is monitoring, and advisors fall into a tuning toolset more than monitoring.

15.4 Monitoring scenarios

In this section, a variety of monitoring data will be used to provide examples and information about what that data tells an administrator.

Important: Although the statistical data that is observed in PMI and request metrics provides a powerful understanding of what is occurring in the environment, it is important to remember that the counters and statistics are mostly averages. You must always verify that the statistics make sense for your environment. For example, if an EJB seems to be having good response time, but it is throwing and catching an exception shortly after entering the bean, it would still appear to be having excellent response time. Response time alone is not enough to indicate a healthy system.

15.4.1 Database interactions

One of the hotspots for monitoring is interactions with external servers, especially interactions with databases. PMI provides a good set of metrics at even the basic level for these types of interactions.

Consider the data snapshot shown in Figure 15-27. This snapshot was taken in the example environment while the server was under load. You can view this report by clicking **Monitoring and Tuning** → **Performance Viewer** → **Current Activity** → **Performance Modules** → **JDBC Connection Pools**. The monitoring level in effect is the Basic level.

Select	Marker	Name	Value	Scale	Update	Scaled Value
jdbc/TradeDataSource						
<input checked="" type="checkbox"/>		CreateCount (?)	1.0	<input type="text" value="1.0"/>	Update	1.0
<input checked="" type="checkbox"/>		CloseCount (?)	0.0	<input type="text" value="1.0E20"/>	Update	0.0
<input checked="" type="checkbox"/>		PoolSize (?)	1.0	<input type="text" value="1.0"/>	Update	1.0
<input type="checkbox"/>		FreePoolSize (?)	1.0	<input type="text" value="1.0"/>	Update	1.0
<input type="checkbox"/>		WaitingThreadCount (?)	0.0	<input type="text" value="1.0E20"/>	Update	0.0
<input type="checkbox"/>		PercentUsed (?)	0.0	<input type="text" value="1.0E20"/>	Update	0.0
<input type="checkbox"/>		UseTime (?)	38.53416	<input type="text" value="1.0"/>	Update	38.53416
<input type="checkbox"/>		WaitTime (?)	0.0	<input type="text" value="1.0E20"/>	Update	0.0
JDBC Connection Pools						
<input checked="" type="checkbox"/>		CreateCount (?)	4.0	<input type="text" value="1.0"/>	Update	4.0
<input checked="" type="checkbox"/>		CloseCount (?)	0.0	<input type="text" value="1.0E20"/>	Update	0.0
<input checked="" type="checkbox"/>		PoolSize (?)	4.0	<input type="text" value="1.0"/>	Update	4.0
<input type="checkbox"/>		FreePoolSize (?)	3.0	<input type="text" value="1.0"/>	Update	3.0
<input type="checkbox"/>		WaitingThreadCount (?)	0.0	<input type="text" value="1.0E20"/>	Update	0.0
<input type="checkbox"/>		PercentUsed (?)	2.0	<input type="text" value="1.0"/>	Update	2.0
<input type="checkbox"/>		UseTime (?)	63.33555	<input type="text" value="1.0"/>	Update	63.33555
<input type="checkbox"/>		WaitTime (?)	0.0	<input type="text" value="1.0E20"/>	Update	0.0

Figure 15-27 Basic PMI for database

From a monitoring perspective, this snapshot provides several useful statistics that can help the administrator understand if the database interaction is healthy. For measuring runtime health, the following counters are beneficial:

- ▶ PercentUsed indicates if the database connections are being overutilized or underutilized. Depending on the application and capacity of the database, it is typically not a good sign if the database is 100% utilized. If a connection pool becomes 100% utilized all of the time after the system has been tuned, either the database might need more capacity to support more connections, or some type of error is occurring. For example, the application might have a connection leak. Utilization is a good indicator that database connections need some attention.
- ▶ Similarly, it is not unreasonable to have waiting threads on the data source, because the resource is shared. From a monitoring perspective, it is a combination of the waiting thread count and the wait time that makes an interesting combination. If the wait time and waiting thread count grow over time, this is an indication that the database might be responding more slowly than desired, or there are insufficient connections available to support the load. How long is too long a wait time depends on application service level agreements and whether wait times occur all of the time or only on occasion under exceptional load.
- ▶ UseTime can help understand how much time is spent communicating with the database and can help to indicate if database response is degrading.

If the administrator believes there are response time errors with the database, request metrics can be useful in diagnosing with which components the application is spending its time.

JCA interactions

JCA is the more generic form of the JDBC, and is used with standard adapters that comply with the JCA standard. The adapter can have connection pooling that is the same as used with JDBC. Probably the most common JCA adapter other than JDBC is the one used for JMS connections, whether connecting to the service integration bus or connecting to an external JMS provider, such as WebSphere MQ.

Hence, when working with JCA or JMS connections, the same indicators used for the JDBC and discussed in 15.4.1, “Database interactions” on page 566 are relevant, with the exception of counters that are JDBC specific. When monitoring, it is beneficial in particular to monitor the waiting threads, their wait time, and the current pool size.

15.4.2 Threading resources

Assuming that there is enough CPU and memory, it is reasonable to surmise that thread pools along with resource connections are a major factor in understanding the limits of throughput on the application server. The various thread pools in the application server control the entry points for requests into the system. If the pool is exhausted, then requests to the system are queued and have to wait. From a monitoring perspective, it is preferable that thread pools are not constantly exhausted and running at their maximum.

Before monitoring the thread pools, the administrator needs to first be aware of what types of thread pools their application uses. The following application topologies provide some insight into what needs to be considered in understanding which thread pools might be important to application runtime throughput.

In the first example, consider a clustered application server environment where the deployed application has both web and EJB components deployed together in the JVM (Figure 15-28). For performance reasons, WebSphere Application Server will use process affinity when invoking the EJBs, sending requests from the web container to the EJB container in the same JVM.

In this scenario, threads are not swapped and the requests are executed on web container threads. Even though EJBs are extensively used in the application, the ORB thread pool would not be used in this application topology. Web container threads in this topology control the concurrency of the application. This is the topology used by the sample application and is common to many JEE applications.

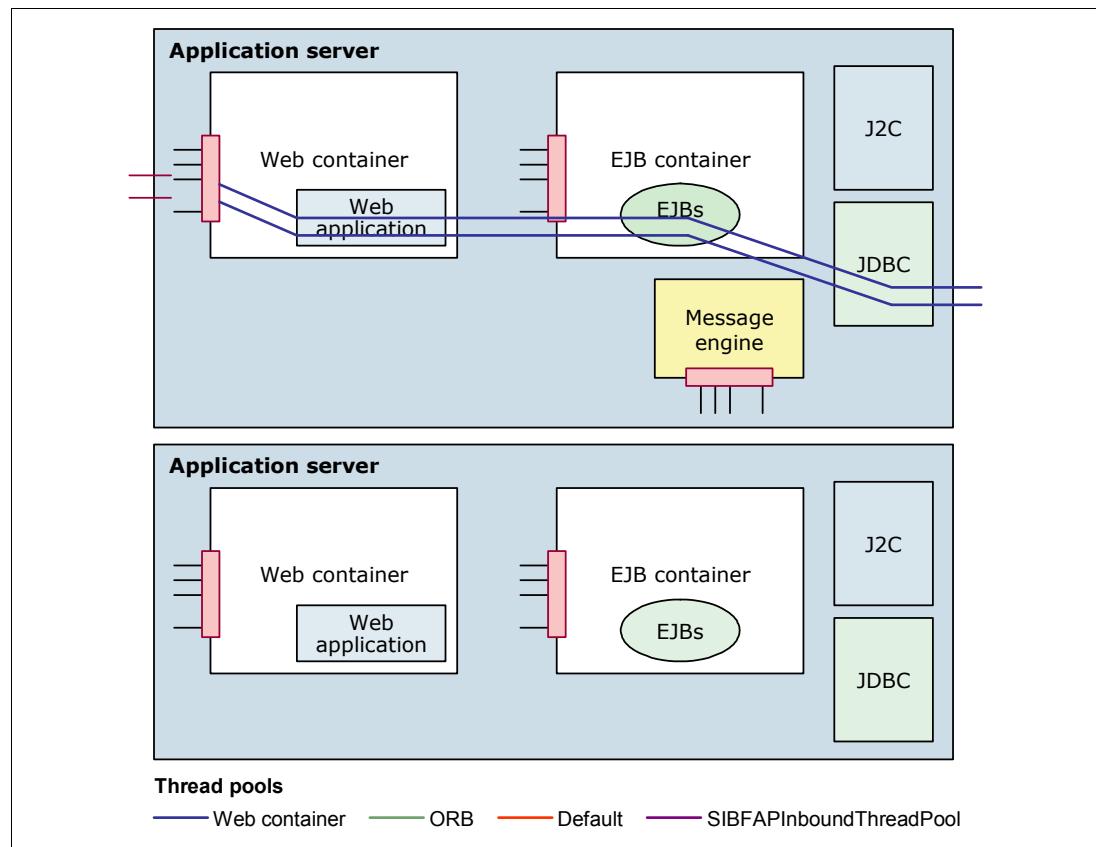


Figure 15-28 Web application

But what if the EJB components were deployed in a separate JVM, as shown in Figure 15-29?

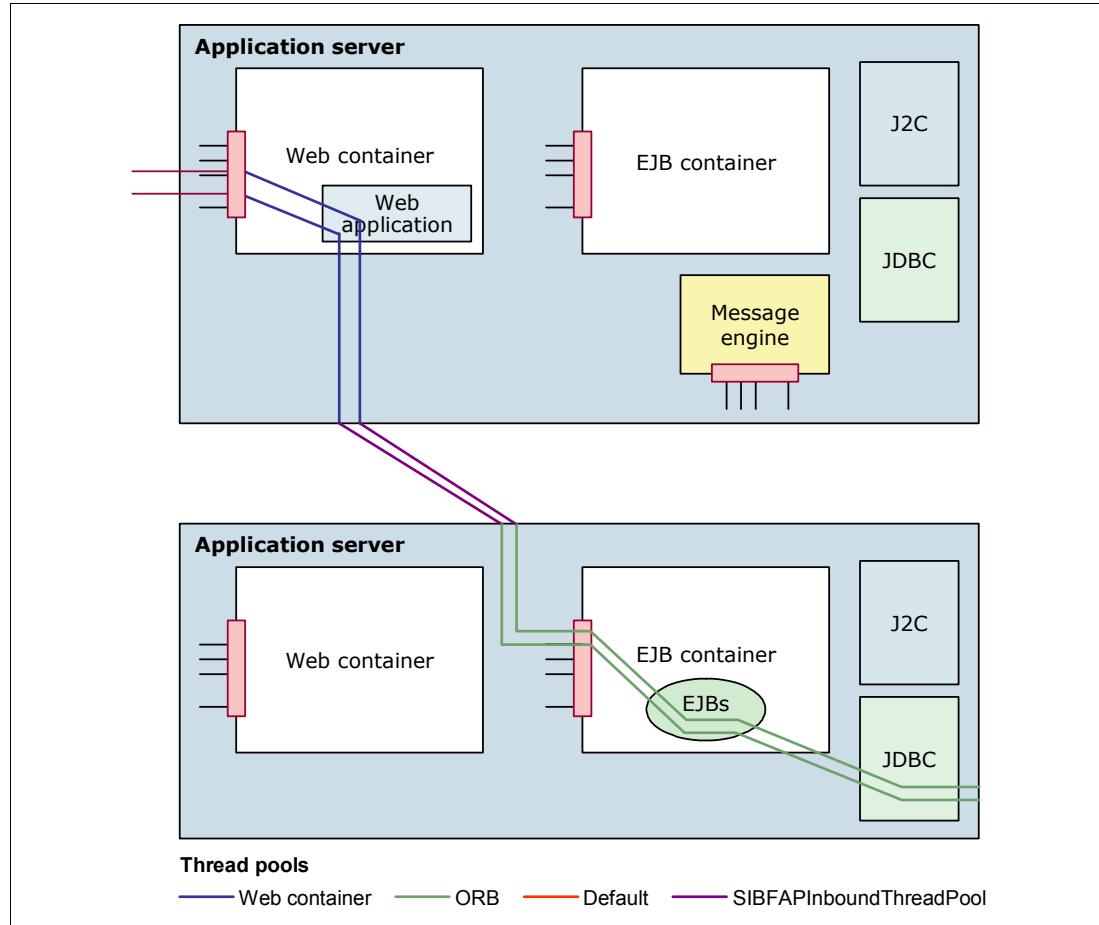


Figure 15-29 Distributed application model

In such a scenario, the ORB thread pool in each application server would become equally important as the web container thread pool. Request throughput would now be controlled by the web container, the ORB thread pool, and other parameters such as memory and connections to external resources.

Different environment resources use different thread pools and a key consideration to understand as an administrator is what are the components with which the deployed applications will interact, and what is the likely environmental impact.

Tivoli Performance Viewer provides a number of resources that can assist you with understanding ThreadPool utilization, and you should consult the Thread Counters to learn more.

Figure 15-30 shows a graphic where the red line represents the web container active thread pool count available when PMI is activated at the basic level. The report is viewed by clicking **Monitoring and Tuning** → **Performance Viewer** → **Current Activity** → **Performance Modules**. Expand the **Thread Pools** section and select **Web container**.

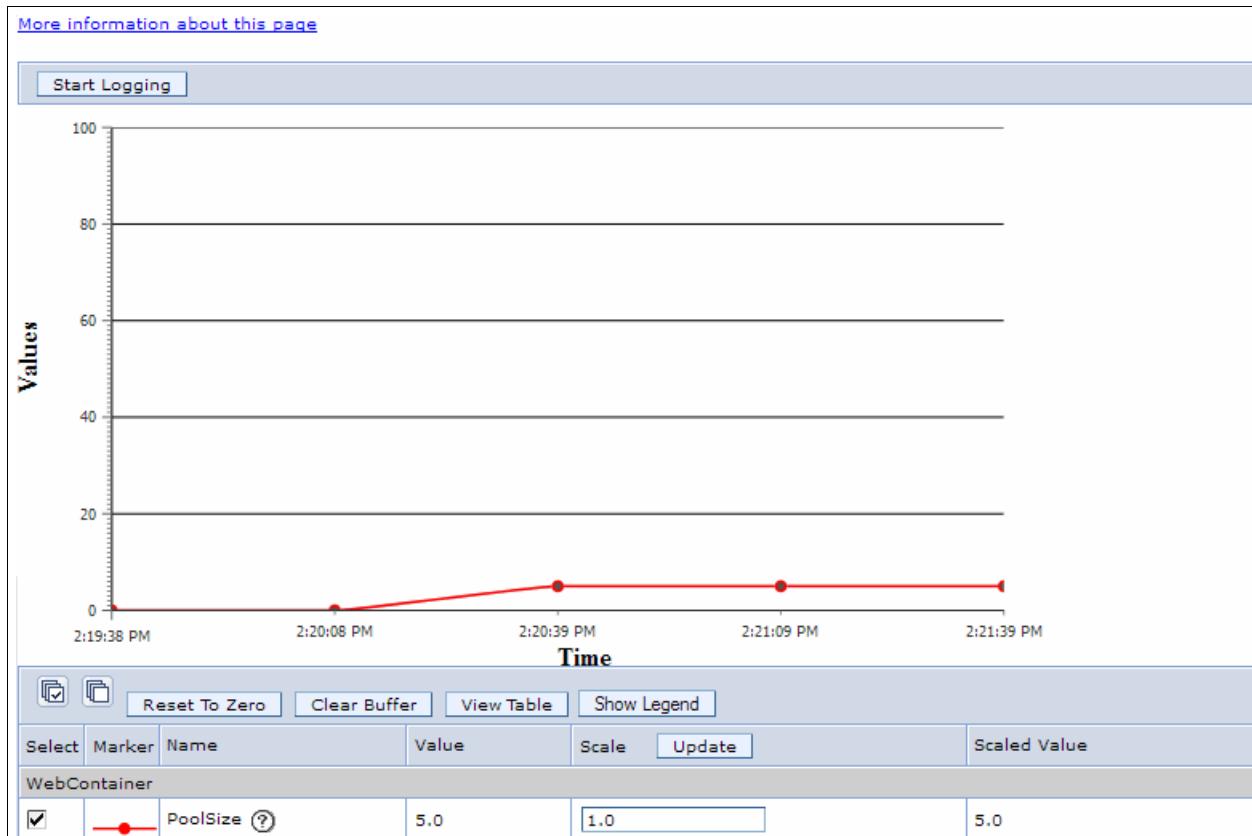


Figure 15-30 Thread Pool counters

15.4.3 JVM memory usage

Environmentally, the way the applications and application servers use memory is important to the overall server performance. As a performance hotspot, it should be no surprise that the application server PMI data provides some basic level monitoring capabilities.

The JVM Used memory counter can be monitored and displayed in the TPV graph. Figure 15-31 shows an example of this data.

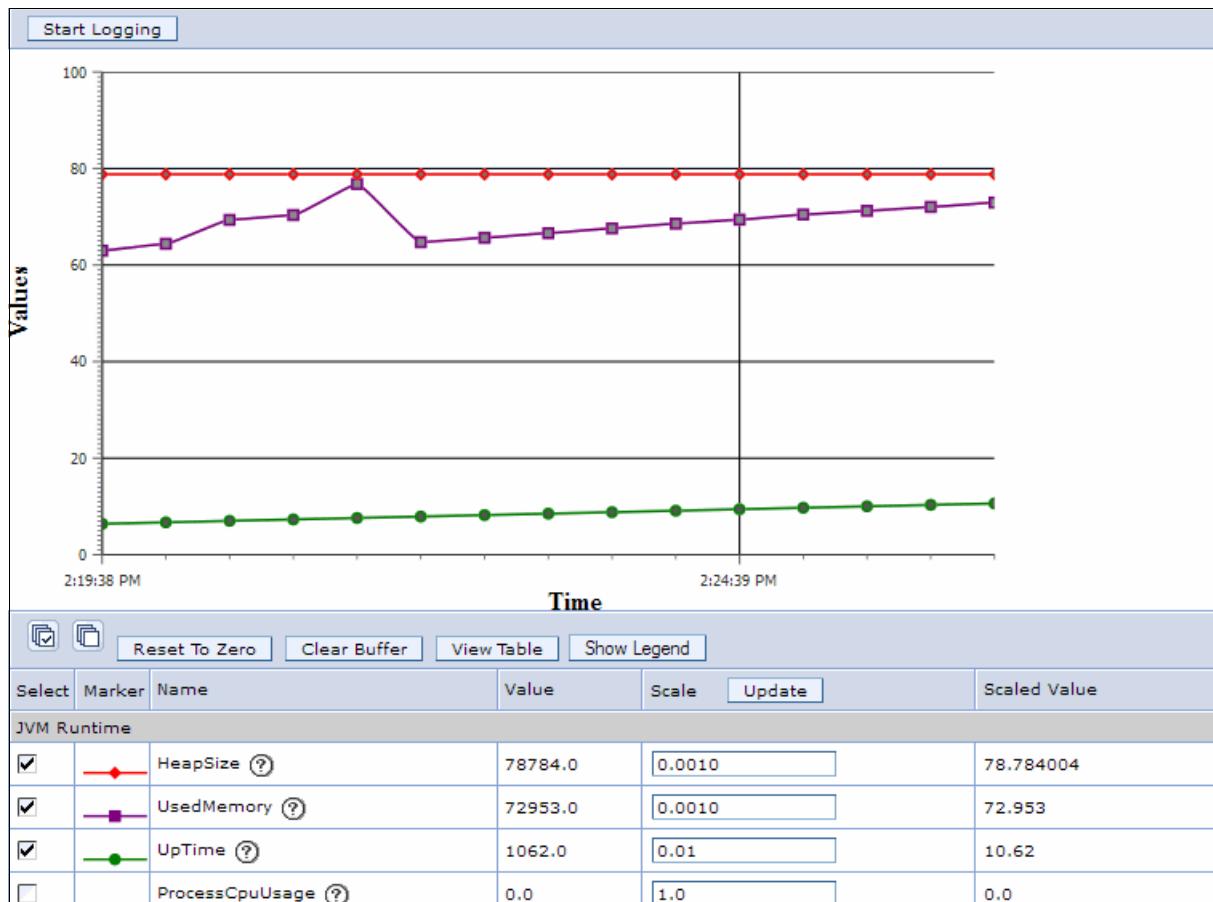


Figure 15-31 JVM memory usage

The report can be viewed by clicking **Monitoring and Tuning** → **Performance Viewer** → **Current Activity** → **Performance Modules**, and then clicking **JVM Runtime**.

This graphical view is a good way to visually check that memory is not constantly growing and is stabilized over time by garbage collection. It also gives a good indication of how frequently garbage collection might be occurring.

However, when tuning memory sizes, use verbose garbage collection and a combination of the application server support tools available in IBM Support Assistant for the analysis of the memory usage.

Note: TPV provides a good representation of what is occurring with the JVM memory, but the IBM Garbage Collection and Memory Visualizer tool and the IBM Pattern Modelling and Analysis for Java Garbage Collector tool provide more insightful help with memory related configurations to assist tuning memory. For example, you can be provided with advice on heap sizes and garbage collection algorithms based on the patterns observed in the garbage collection log.

These tools are available as add-ons to IBM Support Assistant. Use IBM Support Assistant setup wizards to download these and other Support tools. For more information, refer to the following websites:

- ▶ IBM Support Assistant:
<http://www.ibm.com/software/support/isa/>
- ▶ Complete list of ISA available add-ons:
<http://www.ibm.com/support/docview.wss?uid=swg27013116>
- ▶ IBM Education Assistant module for The IBM Garbage Collection and Memory Visualizer:
http://publib.boulder.ibm.com/infocenter/ieduasst/v1r1m0/index.jsp?topic=/com.ibm.iea.was_v7/was/7.0/ProblemDetermination/WASv7_GCMVOverview/player.html

15.4.4 Request level details

When examining performance data, especially when tuning a server, it can be useful to have more detailed data. This situation is particularly true when first trying to pin down and explore bottlenecks. Request metrics can provide this lower level information, showing where time is spent in a request.

Manually creating an application server perceived response time graph (Figure 15-32) can help illustrate the response time of each component as the request is processed. The request metrics at the outer most or edge component are taken, as well as all inner components, as each one handles the request. Each new component represents a narrower view excluding the work done by the components that proceed it.

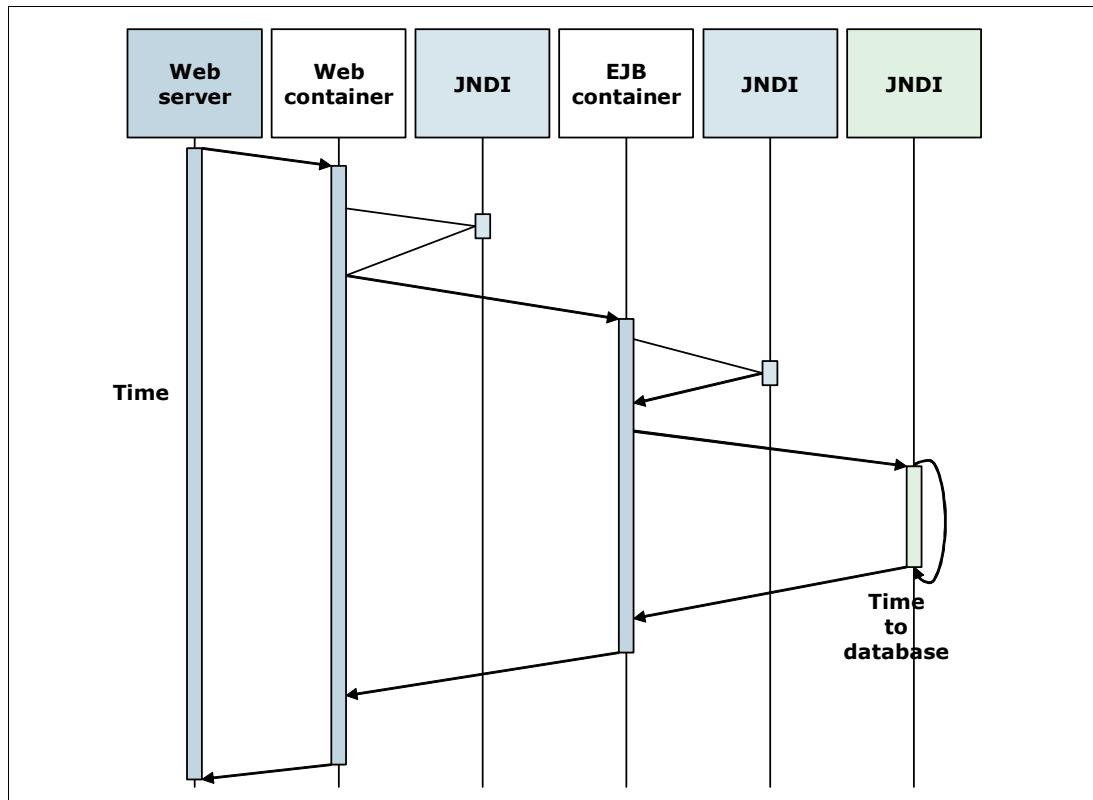


Figure 15-32 Time view of request metrics

Tip: The metrics, when printed to the standard logs, are printed from right to left with the innermost statistic reported first.

Request metrics are ideal for helping to identify the slowest and worst performing request types and can assist in identifying areas where performance can be improved. As a second phase, data from more detailed monitoring for the request type can be broken down to provide insight on where the bulk of time is being spent and provide insight into what areas of the code warrant investigation.

Unlike PMI, there is no built-in tool for request metric analysis, so for these examples, the data will be extracted from the logs. IBM has tooling available in the Tivoli monitoring suites, but it is not part of the application server offering.

Example 15-1 shows the request metrics captured for a single request in the standard log file. In this example, request metrics are enabled with a trace level of hops and all components instrumented. As you can see, it is quite verbose even with a small amount of detail active.

Example 15-1 Simplified PMI data example

```
[7/15/11 11:49:56:343 BRT] 00000030 PmiRmArmWrapp I  PMRM0003I:  
parent:ver=1,ip=9.44.168.198,time=1310740042828,pid=9588,reqid=2,event=1 -  
current:ver=1,ip=9.44.168.198,time=1310740042828,pid=9588,reqid=6,event=1 type=JDBC
```

```

detail=javax.resource.spi.ManagedConnectionFactory.matchManagedConnections(Set, Subject,
ConnectionRequestInfo) elapsed=0
[7/15/11 11:49:56:390 BRT] 00000030 PmiRmArmWrapp I PMRM0003I:
parent:ver=1,ip=9.44.168.198,time=1310740042828,pid=9588,reqid=2,event=1 -
current:ver=1,ip=9.44.168.198,time=1310740042828,pid=9588,reqid=7,event=1 type=JDBC
detail=javax.resource.spi.ManagedConnection.getConnection(Subject, ConnectionRequestInfo) elapsed=0
[7/15/11 11:49:56:421 BRT] 00000030 PmiRmArmWrapp I PMRM0003I:
parent:ver=1,ip=9.44.168.198,time=1310740042828,pid=9588,reqid=2,event=1 -
current:ver=1,ip=9.44.168.198,time=1310740042828,pid=9588,reqid=8,event=1 type=JDBC
detail=javax.resource.spi.XAResource.start(Xid, int) elapsed=0
[7/15/11 11:49:56:734 BRT] 00000030 PmiRmArmWrapp I PMRM0003I:
parent:ver=1,ip=9.44.168.198,time=1310740042828,pid=9588,reqid=2,event=1 -
current:ver=1,ip=9.44.168.198,time=1310740042828,pid=9588,reqid=9,event=1 type=JDBC
detail=java.sql.Statement.executeQuery(String) elapsed=297
...

```

Lots of lines deleted for simplification

```

...
[7/15/11 11:50:02:031 BRT] 00000030 PmiRmArmWrapp I PMRM0003I:
parent:ver=1,ip=9.44.168.198,time=1310740042828,pid=9588,reqid=2,event=1 -
current:ver=1,ip=9.44.168.198,time=1310740042828,pid=9588,reqid=75,event=1 type=JDBC
detail=java.sql.PreparedStatement.executeQuery() elapsed=516
[7/15/11 11:50:02:078 BRT] 00000030 PmiRmArmWrapp I PMRM0003I:
parent:ver=1,ip=9.44.168.198,time=1310740042828,pid=9588,reqid=2,event=1 -
current:ver=1,ip=9.44.168.198,time=1310740042828,pid=9588,reqid=76,event=1 type=JDBC
detail=javax.resource.spi.XAResource.end(Xid, int) elapsed=0
[7/15/11 11:50:02:078 BRT] 00000030 PmiRmArmWrapp I PMRM0003I:
parent:ver=1,ip=9.44.168.198,time=1310740042828,pid=9588,reqid=2,event=1 -
current:ver=1,ip=9.44.168.198,time=1310740042828,pid=9588,reqid=77,event=1 type=JDBC
detail=javax.resource.spi.XAResource.commit(Xid, boolean) elapsed=0
[7/15/11 11:50:02:093 BRT] 00000030 PmiRmArmWrapp I PMRM0003I:
parent:ver=1,ip=9.44.168.198,time=1310740042828,pid=9588,reqid=2,event=1 -
current:ver=1,ip=9.44.168.198,time=1310740042828,pid=9588,reqid=78,event=1 type=JDBC
detail=javax.resource.spi.ManagedConnection.cleanup() elapsed=0
[7/15/11 11:50:02:109 BRT] 00000030 PmiRmArmWrapp I PMRM0003I:
parent:ver=1,ip=9.44.168.198,time=1310740003343,pid=5576,reqid=21,event=1 -
current:ver=1,ip=9.44.168.198,time=1310740042828,pid=9588,reqid=2,event=1 type=URI
detail=/daytrader/scenario elapsed=7656

```

If you do not have tools, a simple editor with good search and parsing capabilities can be used to filter the request based on reqid or other fields.

The timing data can be plotted manually, as illustrated in Figure 15-33. Graphing complex applications might be an easy way to visualize where time is being spent. Plotting data manually can take some time. Interaction diagrams can be substituted and the timing plotted on the diagrams.

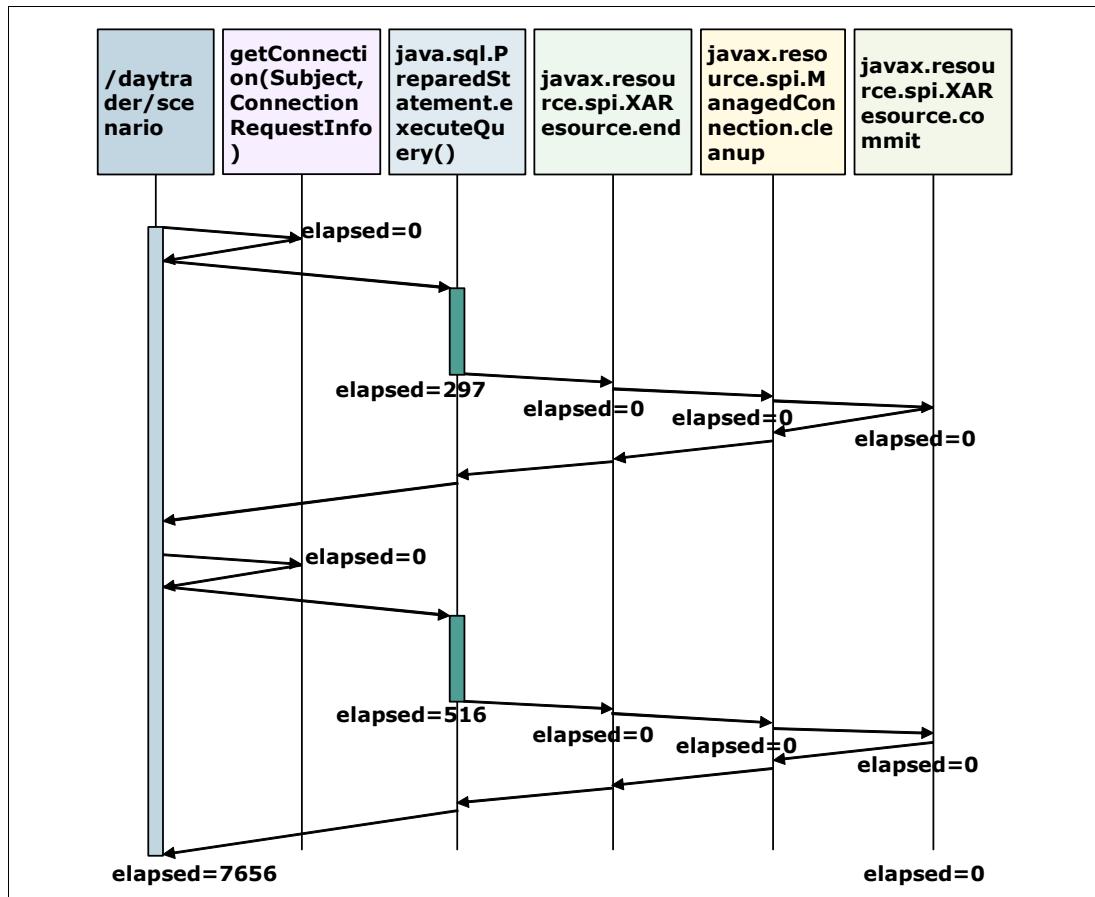


Figure 15-33 Hops time graph for trade application

In this example, the request takes 7656 milliseconds. Looking at all of the details in the log file, you can see exactly where any piece of total time is being spent. Of note in this configuration is that only the edge metrics are captured, the edge being the JDBC data and the original servlet URI request.

Whatever representation is preferred, the key understanding that is required is that request metrics provide a useful mechanisms for the analysis of where time is spent in requests that are being executed.

As more detail is added, the picture becomes more complete. Instead of a trace level of hops, you can use trace level debug to gather more details (Figure 15-34).

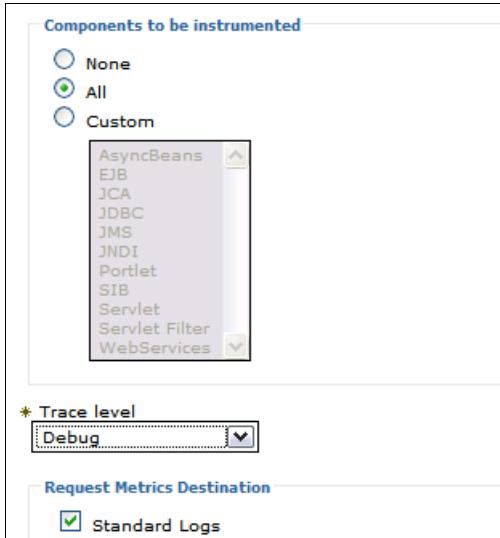


Figure 15-34 Request metrics configuration panel

Example 15-2 shows metrics captured at a trace level of debug.

Example 15-2 Additional component types now recorded

```
[7/15/11 15:27:52:609 BRT] 00000030 PmiRmArmWrapp I PMRM0003I:  
parent:ver=1,ip=9.44.168.198,time=1310754399656,pid=7108,reqid=4,event=1 -  
current:ver=1,ip=9.44.168.198,time=1310754399656,pid=7108,reqid=11,event=1 type=JDBC  
detail=javax.resource.spi.ManagedConnectionFactory.createManagedConnection(Subject,  
ConnectionRequestInfo) elapsed=250  
[7/15/11 15:27:52:625 BRT] 00000030 PmiRmArmWrapp I PMRM0003I:  
parent:ver=1,ip=9.44.168.198,time=1310754399656,pid=7108,reqid=4,event=1 -  
current:ver=1,ip=9.44.168.198,time=1310754399656,pid=7108,reqid=12,event=1 type=JDBC  
detail=javax.resource.spi.ManagedConnectionFactory.matchManagedConnections(Set, Subject,  
ConnectionRequestInfo) elapsed=0  
[7/15/11 15:27:52:625 BRT] 00000030 PmiRmArmWrapp I PMRM0003I:  
parent:ver=1,ip=9.44.168.198,time=1310754399656,pid=7108,reqid=4,event=1 -  
current:ver=1,ip=9.44.168.198,time=1310754399656,pid=7108,reqid=13,event=1 type=JDBC  
detail=javax.resource.spi.ManagedConnection.getConnection(Subject, ConnectionRequestInfo)  
elapsed=0  
[7/15/11 15:27:52:640 BRT] 00000030 PmiRmArmWrapp I PMRM0003I:  
parent:ver=1,ip=9.44.168.198,time=1310754399656,pid=7108,reqid=4,event=1 -  
current:ver=1,ip=9.44.168.198,time=1310754399656,pid=7108,reqid=14,event=1 type=JDBC  
detail=javax.resource.spi.XAResource.start(Xid, int) elapsed=0  
[7/15/11 15:27:52:859 BRT] 00000030 PmiRmArmWrapp I PMRM0003I:  
parent:ver=1,ip=9.44.168.198,time=1310754399656,pid=7108,reqid=4,event=1 -  
current:ver=1,ip=9.44.168.198,time=1310754399656,pid=7108,reqid=15,event=1 type=JDBC  
detail=java.sql.Statement.executeQuery(String) elapsed=203  
...  
Lines deleted  
...  
[7/15/11 15:27:54:843 BRT] 00000030 PmiRmArmWrapp I PMRM0003I:  
parent:ver=1,ip=9.44.168.198,time=1310754399656,pid=7108,reqid=4,event=1 -  
current:ver=1,ip=9.44.168.198,time=1310754399656,pid=7108,reqid=56,event=1 type=JDBC  
detail=java.sql.PreparedStatement.executeQuery() elapsed=31
```

```

[7/15/11 15:27:54:906 BRT] 00000030 PmiRmArmWrapp I PMRM0003I:
parent:ver=1,ip=9.44.168.198,time=1310754399656,pid=7108,reqid=4,event=1 -
current:ver=1,ip=9.44.168.198,time=1310754399656,pid=7108,reqid=57,event=1 type=JDBC
detail=javax.resource.spi.ManagedConnection.getConnection(Subject, ConnectionRequestInfo)
elapsed=0
[7/15/11 15:27:54:921 BRT] 00000030 PmiRmArmWrapp I PMRM0003I:
parent:ver=1,ip=9.44.168.198,time=1310754399656,pid=7108,reqid=4,event=1 -
current:ver=1,ip=9.44.168.198,time=1310754399656,pid=7108,reqid=58,event=1 type=JDBC
detail=java.sql.PreparedStatement.executeUpdate() elapsed=0
[7/15/11 15:27:54:921 BRT] 00000030 PmiRmArmWrapp I PMRM0003I:
parent:ver=1,ip=9.44.168.198,time=1310754399656,pid=7108,reqid=4,event=1 -
current:ver=1,ip=9.44.168.198,time=1310754399656,pid=7108,reqid=59,event=1 type=JDBC
detail=javax.resource.spi.XAResource.end(Xid, int) elapsed=0
[7/15/11 15:27:54:937 BRT] 00000030 PmiRmArmWrapp I PMRM0003I:
parent:ver=1,ip=9.44.168.198,time=1310754399656,pid=7108,reqid=4,event=1 -
current:ver=1,ip=9.44.168.198,time=1310754399656,pid=7108,reqid=60,event=1 type=JDBC
detail=javax.resource.spi.XAResource.commit(Xid, boolean) elapsed=16
[7/15/11 15:27:54:937 BRT] 00000030 PmiRmArmWrapp I PMRM0003I:
parent:ver=1,ip=9.44.168.198,time=1310754399656,pid=7108,reqid=4,event=1 -
current:ver=1,ip=9.44.168.198,time=1310754399656,pid=7108,reqid=61,event=1 type=JDBC
detail=javax.resource.spi.ManagedConnection.cleanup() elapsed=0
[7/15/11 15:27:54:968 BRT] 00000030 PmiRmArmWrapp I PMRM0003I:
parent:ver=1,ip=9.44.168.198,time=1310754399656,pid=7108,reqid=2,event=1 -
current:ver=1,ip=9.44.168.198,time=1310754399656,pid=7108,reqid=4,event=1 type=EJB
detail=org.apache.geronimo.samples.daytrader.ejb3.TradeSLSBBean.login elapsed=3484

```

For more information about these techniques, see the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.nd.multiplatform.doc/info/ae/ae/tprf_requestmetrics.html

15.5 IBM Tivoli Composite Application Manager for WebSphere Application Server

IBM Tivoli Composite Application Manager (ITCAM) for WebSphere is shipped with WebSphere Application V8. After being installed, it is embedded in the application server and can be configured to monitor applications performance providing real-time status information in WebSphere console. More information about ITCAM for WebSphere Application Server can be found at the following website:

http://publib.boulder.ibm.com/infocenter/tivihelp/v24r1/index.jsp?topic=/com.ibm.itcamfad.doc_7101/ecam.html

ITCAM for WebSphere can easily be integrated with ITCAM for Application Diagnostics to provide deep dive diagnostics data, real-time monitoring, and management. ITCAM for Application Diagnostics requires additional licensing. More information can be found at the following website:

http://publib.boulder.ibm.com/infocenter/tivihelp/v24r1/index.jsp?topic=/com.ibm.itcamfad.doc_7101/planning_an_installation/overview_of_itcam_for_application_diagnostics.html

15.5.1 Installing the data collector

The first step in enabling ITCAM is to install and configure a data collector for the monitored servers. ITCAM for WebSphere can be installed from WebSphere media through IBM Installation Manager. For information about the installation, refer to Chapter 2, “Getting started with ITCAM for WebSphere Application Server”, in the *ITCAM for WebSphere Application Server 7.2* manual at the following website:

http://publib.boulder.ibm.com/infocenter/tivihelp/v24r1/topic/com.ibm.itcamfad.doc_7101/itcam_ecam_installation_72.pdf

15.5.2 Configuring IBM Tivoli Composite Application Manager for WebSphere metrics

The last step of the installation process contains a check box to start the configuration tool that configures the servers for data collection. The configuration tool can be started immediately after installation using this option, or you can start it after installation.

The TPV settings for the servers can also be adjusted to include ITCAM data during the configuration.

Complete the following steps in the data configuration wizard:

1. Click **Next** on the data collector configuration window.
2. Select **Configure application servers for data collection** (Figure 15-35) and click **Next**.

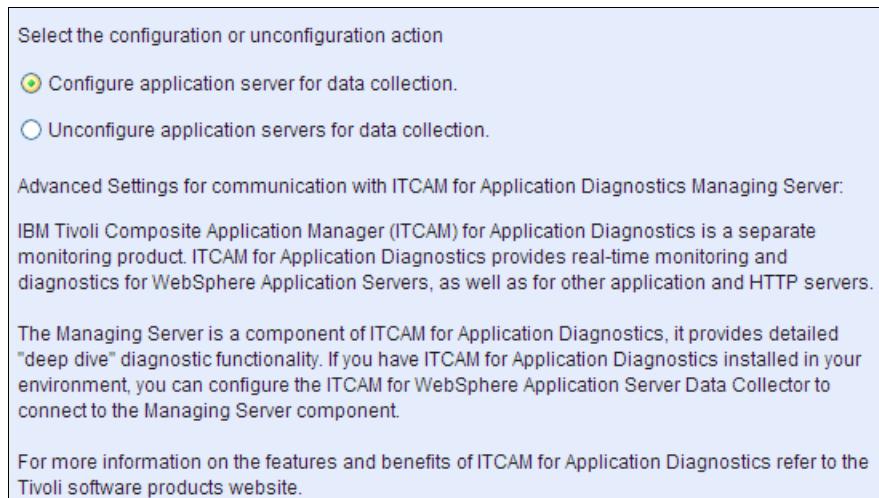


Figure 15-35 Configure application servers for data collection

3. Select the node for which the collector is being configured (Figure 15-36). Click **Next**.

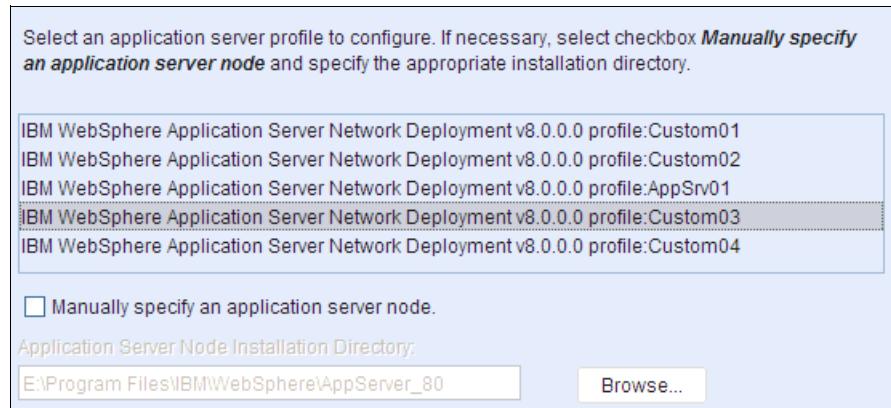


Figure 15-36 Data collector configuration

4. Click **Next** on the directory home window.
5. Enter the administration port details (Figure 15-37). For Network Deployment, these details need to be the information for the deployment manager configuration. Then click **Next**.

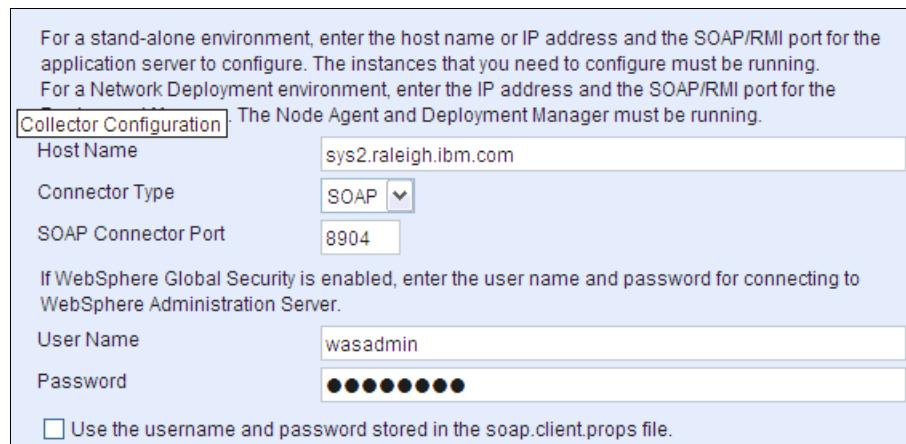


Figure 15-37 Enter administration port details

6. Specify which servers will have data collection configured (Figure 15-38). Click **Next**.

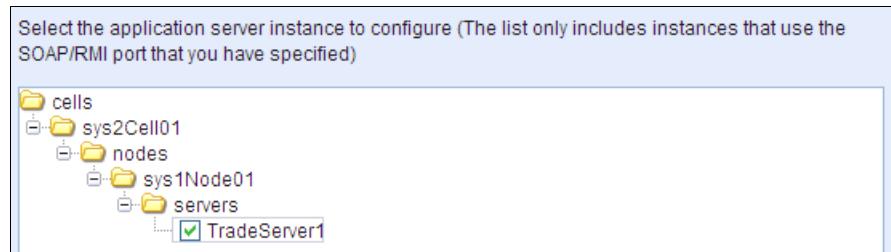


Figure 15-38 Choose servers for which to configure data collector

7. As part of the data collection configuration, the PMI data settings can be modified to include ITCAM metrics, which can be done manually at another time or the administrator can allow the configuration tool to complete the modification (Figure 15-39). Click **Next**.

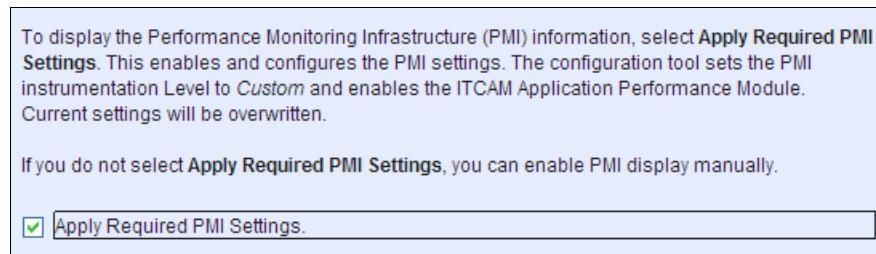


Figure 15-39 Modify PMI settings

8. On the Configure communication to ITCAM for Application Diagnostics Managing Server window, just click **Next**, as we are configuring only the data collector.
9. Finalize the configuration steps (Figure 15-40), and then click **Next**.

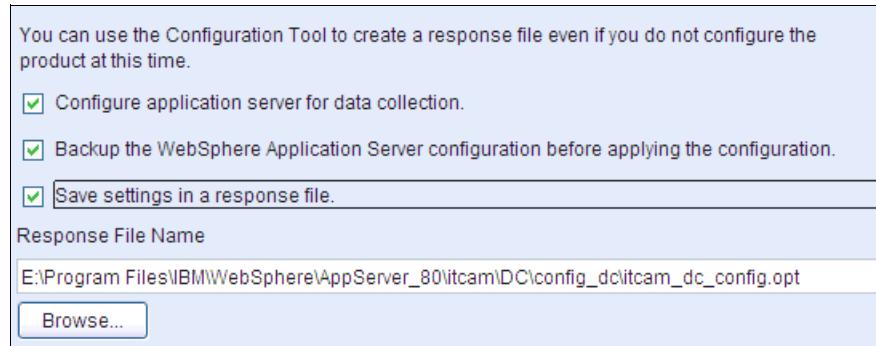


Figure 15-40 Configure the servers for data collection and create options file

10. Click **Finish** and restart the server.

11. After the data collector configuration is complete, navigate in the WebSphere administration console to the Performance Monitoring Infrastructure window for the configured server (Figure 15-41). A new ITCAM for WebSphere Application Server link is now in the Additional Properties section of the window.

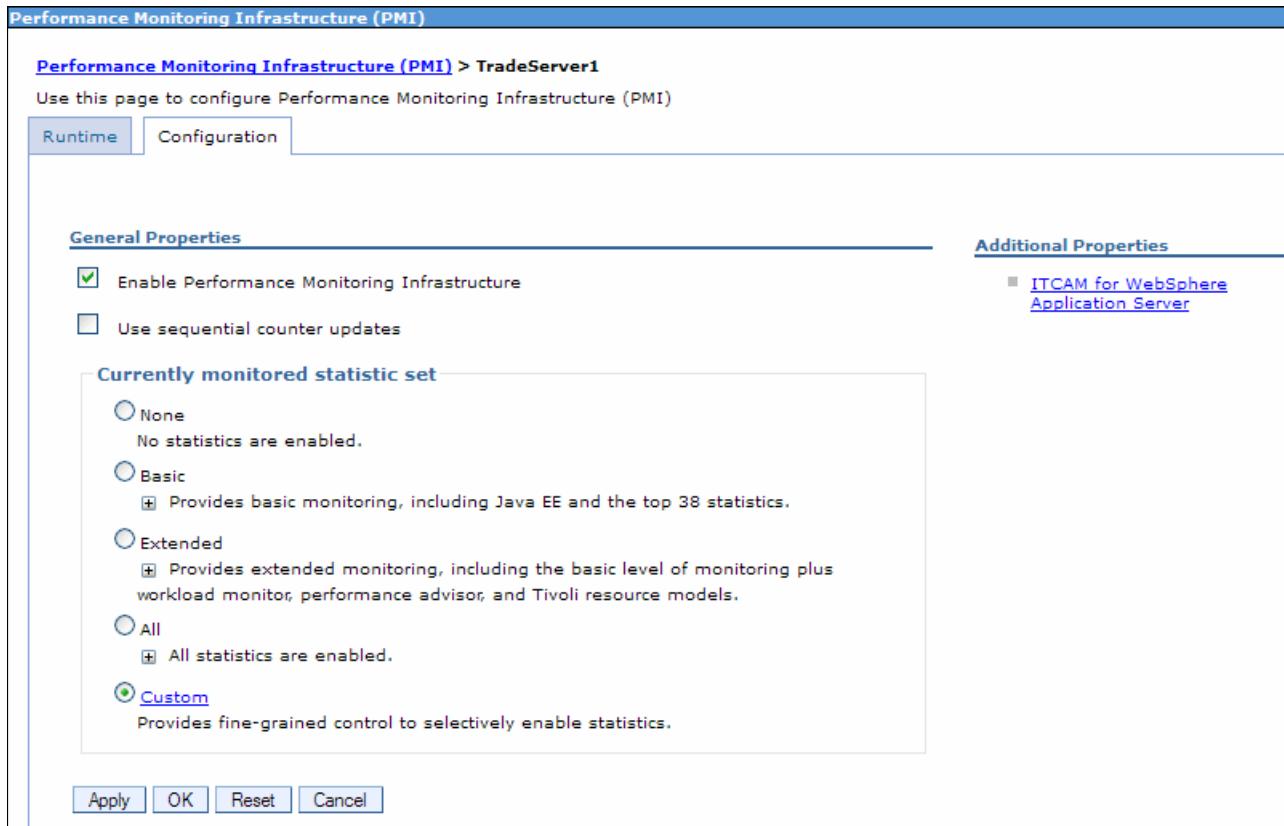


Figure 15-41 PMI configuration with ITCAM for WebSphere Application Server

12. Click the **ITCAM for WebSphere Application Server** link to open the ITCAM for WebSphere Application Server window (Figure 15-42). This window contains the setting that enables the data collector.

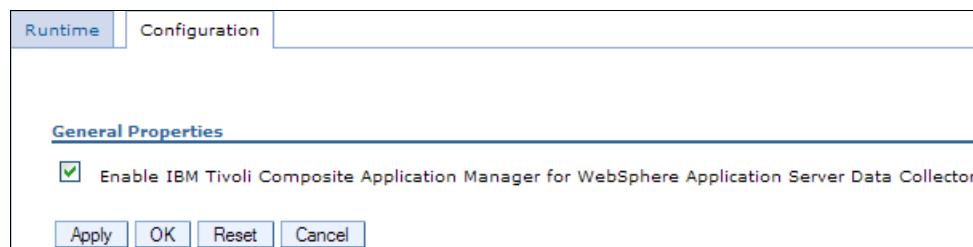


Figure 15-42 ITCAM for WebSphere Application Server window

Note: This setting will only be enabled if you choose to apply the required PMI settings, as shown in Figure 15-39 on page 580. Otherwise, you need to enable the ITCAM for WebSphere Application Server data collector and restart the server.

The PMI settings must also be configured at the custom level.

15.5.3 Viewing IBM Tivoli Composite Application Manager for WebSphere data

This section assumes that the data collector has been installed and the ITCAM for WebSphere Application Server data collector is enabled, as shown in Figure 15-42 on page 581.

1. Navigate to the Performance Monitoring Infrastructure window for the server, as shown in Figure 15-41 on page 581. Confirm that the **Custom** statistic set is selected.
2. Click the **ITCAM for WebSphere Application Server** link and navigate to open the configuration window, as shown in Figure 15-42 on page 581.
3. Click the **Runtime** tab (Figure 15-43).

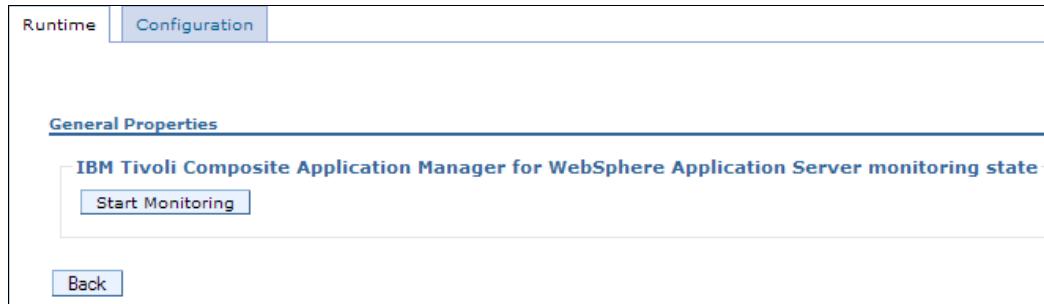


Figure 15-43 ITCAM for WebSphere Application Server Runtime tab

4. Click the **Start Monitoring** button.

Notes: You must start ITCAM for WebSphere Application Server monitoring on a server at each server restart; as this action was taken in the **Runtime** tab, it is not preserved in the configuration.

After monitoring is started, the Stop Monitoring button is displayed instead. If you decide later to stop the ITCAM for WebSphere Application Server monitoring, then return to this window and click that button.

5. Click the server link (TradeServer1) in the breadcrumb at the top of the window to return to the PMI configuration window.
6. Click the **Runtime** tab.

- Click the **Custom** link to navigate into the custom monitoring window. Click **ITCAM Application Performance** to show the ITCAM counters (Figure 15-44).

Note: ITCAM counters are only visible in the **Runtime** tab.

Select	Counter	Type	Description	Status
<input type="checkbox"/>	90%CPUUsage	CountStatistic	This metric collects the CPU usage and then calculates the 90% median CPU usage.	Disabled
<input type="checkbox"/>	90%ResponseTime	CountStatistic	This metric collects the response time and then calculates the 90% median response time.	Disabled
<input type="checkbox"/>	AverageCPUUsage	AverageStatistic	This metric collects the CPU usage and then calculates the average CPU usage in milliseconds.	Disabled
<input type="checkbox"/>	AverageResponseTime	AverageStatistic	This metric collects the response time and then calculates the average response in milliseconds.	Disabled
<input type="checkbox"/>	LastMinuteAverageCPUUsage	AverageStatistic	This metric collects the CPU usage for the last minute and then calculates the average CPU usage in milliseconds.	Disabled
<input type="checkbox"/>	LastMinuteAverageResponseTime	AverageStatistic	This metric collects the response time for the last minute and then calculates the average response in milliseconds.	Disabled
<input type="checkbox"/>	MaximumCPUUsage	CountStatistic	This metric collects the CPU usage and then calculates the maximum CPU usage in milliseconds.	Disabled
<input type="checkbox"/>	MaximumResponseTime	CountStatistic	This metric collects the response time and then calculates the maximum response in milliseconds.	Disabled
<input type="checkbox"/>	MinimumCPUUsage	CountStatistic	This metric collects the CPU usage and then calculates the minimum CPU usage in milliseconds.	Disabled
<input type="checkbox"/>	MinimumResponseTime	CountStatistic	This metric collects the response time and then calculates the minimum response in milliseconds.	Disabled
<input type="checkbox"/>	RequestCount	CountStatistic	The total number of requests.	Disabled

Figure 15-44 ITCAM Application performance counters

- The next step is to choose which ITCAM for WebSphere Application Server counters are needed. The counter window provides additional description information and the current status of the counter.

Choose the counters that are desired and click **Enable**.

Note that some counters can only be activated by the system. This situation means that they will stay in disabled mode, even if you try to enable them. This does not mean the counters are broken. In Figure 15-45, you can see that some counters remain in the Disabled state, even though all the counters were selected to be enabled.

<input type="button" value="Enable"/> <input type="button" value="Disable"/>				
<input checked="" type="checkbox"/>	Counter	Type	Description	Status
<input type="checkbox"/>	90%CPUUsage	CountStatistic	This metric collects the CPU usage and then calculates the 90% median CPU usage.	Disabled
<input type="checkbox"/>	90%ResponseTime	CountStatistic	This metric collects the response time and then calculates the 90% median response time.	Disabled
<input type="checkbox"/>	AverageCPUUsage	AverageStatistic	This metric collects the CPU usage and then calculates the average CPU usage in milliseconds.	Enabled
<input type="checkbox"/>	AverageResponseTime	AverageStatistic	This metric collects the response time and then calculates the average response in milliseconds.	Enabled
<input type="checkbox"/>	LastMinuteAverageCPUUsage	AverageStatistic	This metric collects the CPU usage for the last minute and then calculates the average CPU usage in milliseconds.	Enabled
<input type="checkbox"/>	LastMinuteAverageResponseTime	AverageStatistic	This metric collects the response time for the last minute and then calculates the average response in milliseconds.	Enabled
<input type="checkbox"/>	MaximumCPUUsage	CountStatistic	This metric collects the CPU usage and then calculates the maximum CPU usage in milliseconds.	Disabled
<input type="checkbox"/>	MaximumResponseTime	CountStatistic	This metric collects the response time and then calculates the maximum response in milliseconds.	Disabled
<input type="checkbox"/>	MinimumCPUUsage	CountStatistic	This metric collects the CPU usage and then calculates the minimum CPU usage in milliseconds.	Disabled
<input type="checkbox"/>	MinimumResponseTime	CountStatistic	This metric collects the response time and then calculates the minimum response in milliseconds.	Disabled
<input type="checkbox"/>	RequestCount	CountStatistic	The total number of requests.	Enabled

Figure 15-45 ITCAM counters enabled

9. After enabling the counters, navigate to the Current Activity Tivoli Performance Viewer window for the server that is being monitored (Figure 15-46). Open the Performance Modules tree view and then select the **ITCAM Application Performance** menu item.

Tivoli Performance Viewer > TradeServer1

Use this page to view and refresh performance data for the selected server, change user and log settings, and view summary reports.

Refresh View Module(s)

TradeServer1

- Advisor
- + Settings
- + Summary Reports
- Performance Modules
 - + DCS Statistics
 - + ExtensionRegistryStats.name
 - + SIB Service
 - + Security Authentication
 - + Security Authorization
 - + Enterprise Beans
 - + Dynamic Caching
 - + JDBC Connection Pools
 - + HAManager
 - + JCA Connection Pools
 - JVM Runtime
 - + Object Pool
 - + ORB
 - + pmiWebServiceModule
 - + Servlet Session Manager
 - + Thread Pools
 - + Transaction Manager
 - + Web Applications
 - + Workload Management
 - + ITCAM Application Performance

Servlets Summary Report

More information about this page

Start Logging

Name	Application	Total Requests	Avg Resp Time (ms)
rsp servlet	ibmasyncrsp.war	0	0
Total 1			

Figure 15-46 Select the ITCAM Application performance menu item

10. Click the **View Module(s)** button to view the data (Figure 15-47).

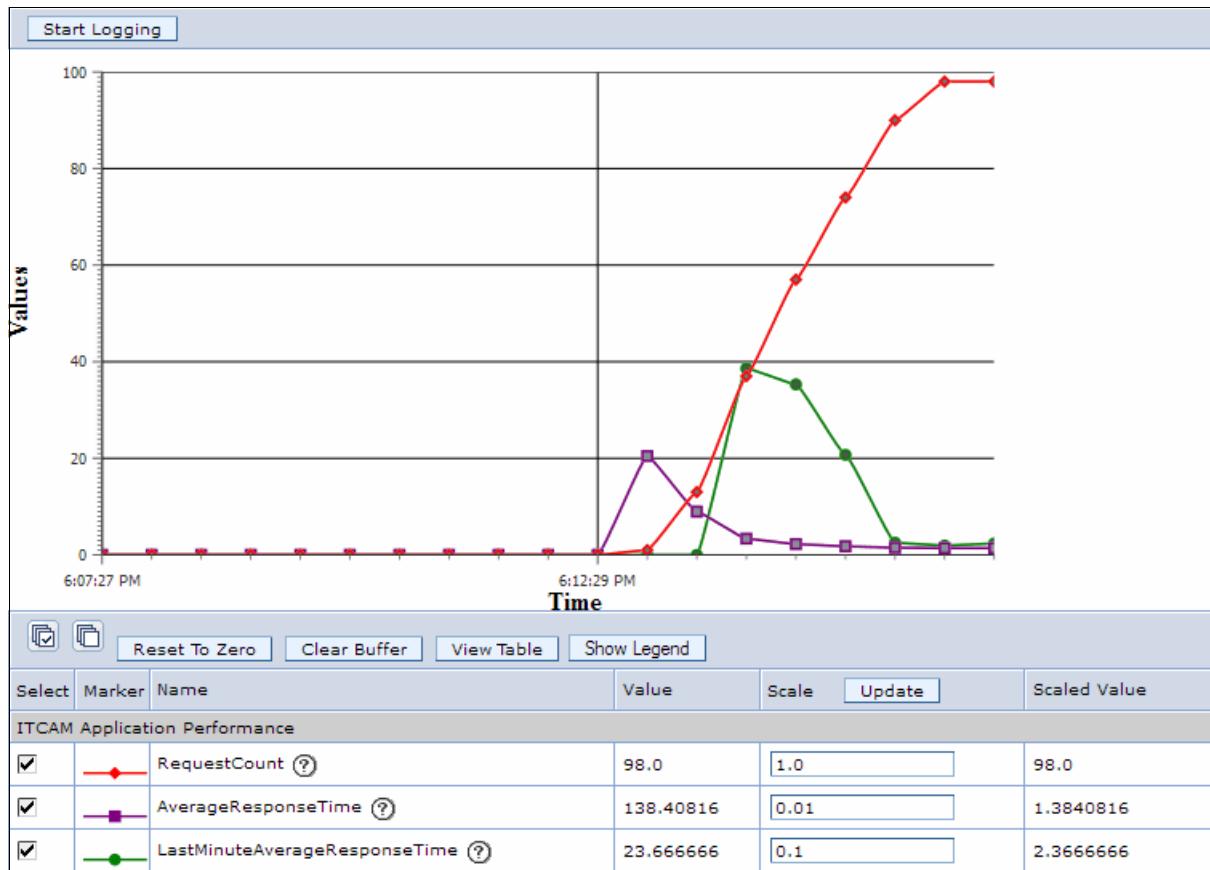


Figure 15-47 ITCAM counters graph

15.6 Additional resources for monitoring

In this section, we cover additional useful resources to be used while monitoring your WebSphere environment.

15.6.1 Verbose garbage collection

Verbose garbage collection was mentioned earlier in the chapter. Again, assuming there is appropriate disk space, a monitoring strategy should include verbose garbage collection.

To enable verbose garbage collection, click **Servers** → **Server Types** → **WebSphere application servers** → <your server> → **Server Infrastructure** → **Java and process management** → **Process definition** → **Additional Properties** → **Java Virtual Machine** and select the **Verbose garbage collection** box, as shown in Figure 15-48.

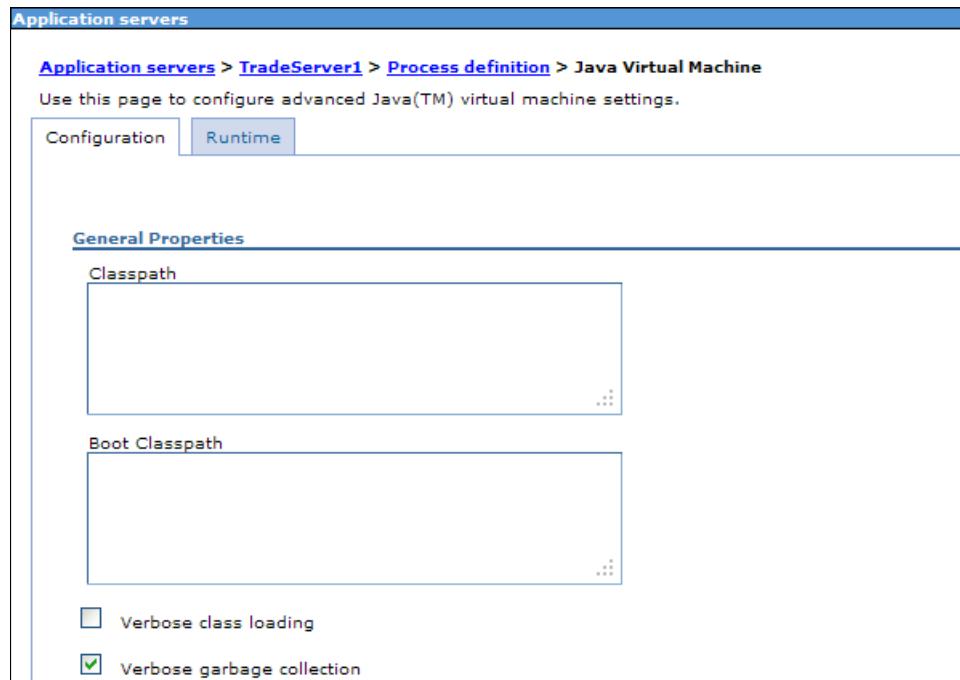


Figure 15-48 Enable verbose garbage collection

Verbose garbage collection can also be enabled on the Runtime tab, as shown in Figure 15-49.

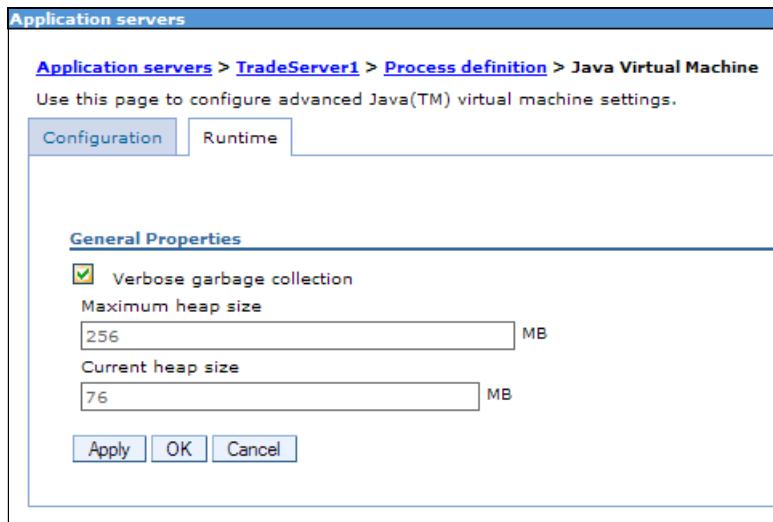


Figure 15-49 Enable verbose garbage collection at run time

When this field is enabled, a report is written to the output stream each time the garbage collector runs, the information can be found usually on native log files. In Version 7, WebSphere used the optimal throughput (optthrput) algorithm for garbage collection. Now generational concurrent garbage (gencon) collection is being used, which allows performance improvements. As optthrput uses a large contiguous heap shared by all threads, when garbage collection is invoked, all of this area is scanned. This policy is beneficial for applications that demand optimal throughput, but it has long pause times as a side effect. Generational concurrent divides the heap memory in two pieces:

- ▶ Nursery, for new objects
- ▶ Tenured, for aged objects

So, in this policy, the time spent to scan one of the areas is smaller.

Example 15-3 shows a sample garbage collection entry in the native_stderr.log file.

Example 15-3 Garbage collection entry

```
<exclusive-start id="1163" timestamp="2011-07-18T20:55:46.656"
intervalms="303968.790">
    <response-info timems="0.130" idlems="0.096" threads="2" lastid="112D9E00"
lastname="IProfiler" />
</exclusive-start>
<af-start id="1164" totalBytesRequested="16680"
timestamp="2011-07-18T20:55:46.765" intervalms="351230.504" />
<cycle-start id="1165" type="scavenge" contextid="0"
timestamp="2011-07-18T20:55:46.765" intervalms="351233.805" />
<gc-start id="1166" type="scavenge" contextid="1165"
timestamp="2011-07-18T20:55:46.765">
    <mem-info id="1167" free="18040128" total="75169792" percent="23">
        <mem type="nursery" free="0" total="14876672" percent="0" />
        <mem type="tenure" free="18040128" total="60293120" percent="29">
            <mem type="soa" free="18040128" total="60293120" percent="29" />
            <mem type="loa" free="0" total="0" percent="0" />
        </mem>
        <remembered-set count="6003" />
    </mem-info>
</gc-start>
<allocation-stats totalBytes="11057944" >
    <allocated-bytes non-tlh="2516584" tlh="8541360" />
    <largest-consumer threadName="Non-deferrable Alarm : 1" threadId="1559A700"
bytes="3554400" />
</allocation-stats>
<gc-op id="1168" type="scavenge" timems="84.993" contextid="1165"
timestamp="2011-07-18T20:55:46.843">
    <scavenger-info tenureage="5" tiltratio="82" />
    <memory-copied type="nursery" objects="44300" bytes="2937136"
bytesdiscarded="872" />
    <memory-copied type="tenure" objects="6709" bytes="2896428" bytesdiscarded="976"
/>
    <copy-failed type="nursery" objects="142" bytes="2368560" />
    <finalization candidates="409" enqueueued="316" />
    <references type="soft" candidates="1104" cleared="0" enqueueued="0"
dynamicThreshold="9" maxThreshold="32" />
    <references type="weak" candidates="225" cleared="0" enqueueued="0" />
    <references type="phantom" candidates="22" cleared="0" enqueueued="0" />
</gc-op>
```

```

<gc-end id="1169" type="scavenge" contextid="1165" durationms="88.918"
timestamp="2011-07-18T20:55:46.843">
  <mem-info id="1170" free="21001096" total="69271552" percent="30">
    <mem type="nursery" free="5905552" total="8978432" percent="65" />
    <mem type="tenure" free="15095544" total="60293120" percent="25">
      <mem type="soa" free="15095544" total="60293120" percent="25" />
      <mem type="loa" free="0" total="0" percent="0" />
    </mem>
    <pending-finalizers system="316" default="0" reference="0" classloader="0" />
    <remembered-set count="4659" />
  </mem-info>
</gc-end>
<cycle-end id="1171" type="scavenge" contextid="1165"
timestamp="2011-07-18T20:55:46.843" />
<allocation-satisfied id="1172" threadId="15030700" bytesRequested="16680" />
<af-end id="1173" timestamp="2011-07-18T20:55:46.843" />
<exclusive-end id="1174" timestamp="2011-07-18T20:55:46.843" durationms="257.158"
/>

```

There are some tools in the IBM Support Assistant that make the garbage collection analyses easier.

For more information about Java memory management, refer to the Java Information Center at the following website:

http://publib.boulder.ibm.com/infocenter/javasdk/v6r0/index.jsp?topic=%2Fcom.ibm.java.doc.diagnostics.60%2Fdiag%2Funderstanding%2Fmemory_management.html

15.6.2 Java dump and core files

Java core files are like a picture of what is occurring inside the Java virtual machine. When it is used, this kind of file is helpful to analyze scenarios where the CPU is reaching 100% percent of utilization, when threads are hanging, or where the performance is slow.

Java dump and system dumps files are a picture of the objects that were in Java virtual machine memory and are helpful in diagnosing memory related problems such as memory leaks.

Both kinds of files can be generated by the WebSphere console by completing the following steps:

1. Click **Troubleshooting** → **Java dumps and cores**.
2. Select the desired server.

3. Click one of the available buttons, as shown in Figure 15-50:

- **Heap dump**
- **Java core**
- **System dump**

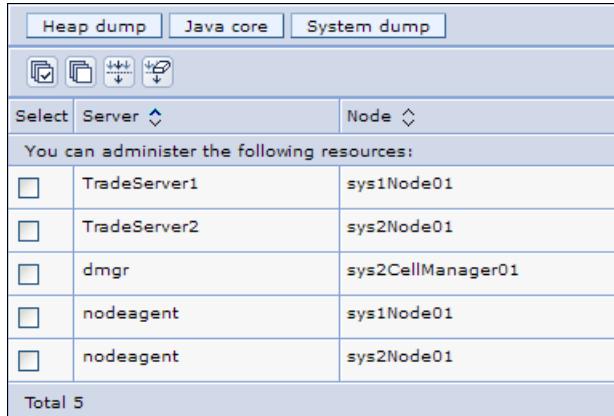


Figure 15-50 Core and dump generation

Look for the files in the profile directory.

For more information about javadumps, refer to the Java Information Center at the following website:

<http://publib.boulder.ibm.com/infocenter/javasdk/v6r0/index.jsp?topic=%2Fcom.ibm.java.doc.diagnostics.60%2Fdiag%2Ftools%2Fjavadump.html>

For more information about heapdumps, refer to the Java Information Center at the following website:

<http://publib.boulder.ibm.com/infocenter/javasdk/v6r0/index.jsp?topic=%2Fcom.ibm.java.doc.diagnostics.60%2Fdiag%2Ftools%2Fheapdump.html>

For more information about system dumps, refer to the Java Information Center at the following website:

http://publib.boulder.ibm.com/infocenter/javasdk/v6r0/index.jsp?topic=%2Fcom.ibm.java.doc.diagnostics.60%2Fdiag%2Ftools%2Fdump_viewer.html

15.6.3 Basic logging

WebSphere Application Server provides a great many logging options and the most significant environment incidents are logged automatically. The basic logging provides Java virtual machine logs, diagnostic trace, and service log files commonly named SystemOut.log, SystemErr.log, trace.log, and activity.log. To support the analysis of logging activity, the IBM Support Assistant can be used to launch the IBM Tivoli Log Analyzer. This tool supports the downloading of symptom catalogs of known issues that go into the logs. Furthermore, application developers can build symptom catalogs and write logs from their applications to add to the manageability of their applications.

Figure 15-51 shows the IBM Tivoli Log analyzer.

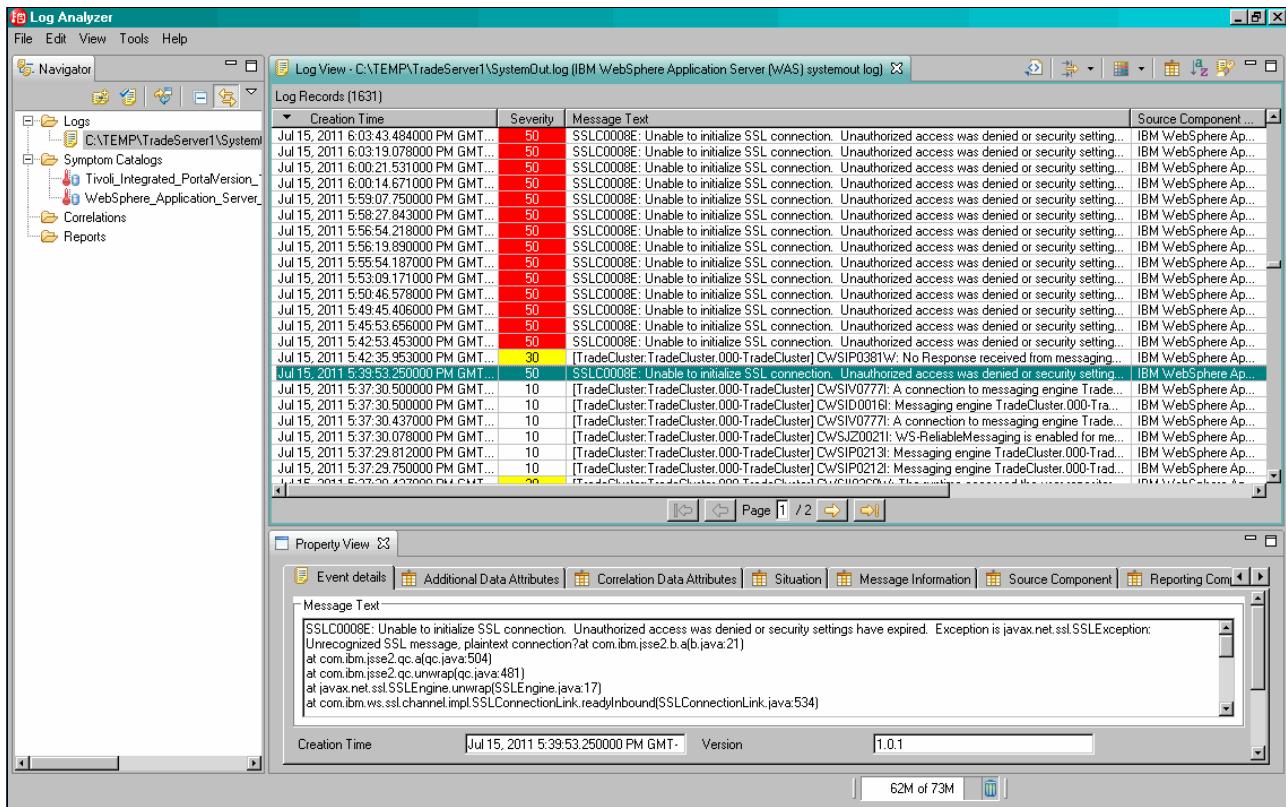


Figure 15-51 IBM Tivoli log analyzer

Notice that the log analyzer classifies messages by assigning them severity values, making it easier to find messages that need immediate attention and analysis.

15.6.4 Advanced logging

In WebSphere V8, an alternative to the basic log and trace facility called High Performance Extensible Logging (HPEL) is offered. It provides three repositories, as shown in Figure 15-52:

- ▶ Log data repository: A storage facility for log records, typically information stored in `SystemOut.log`, `SystemErr.log`, or `java.util.logging` at level detail or higher.
- ▶ Trace data repository: A storage facility for trace records, typically information written to `java.util.logging` below level detail.
- ▶ Text log: A plain text file for log and trace records, it is provided for convenience.

Note: All the data that is written to log and trace repositories are parsed and formatted to be stored in a text log file. For this reason, you should consider disabling the log file as soon as possible to enhance server performance.

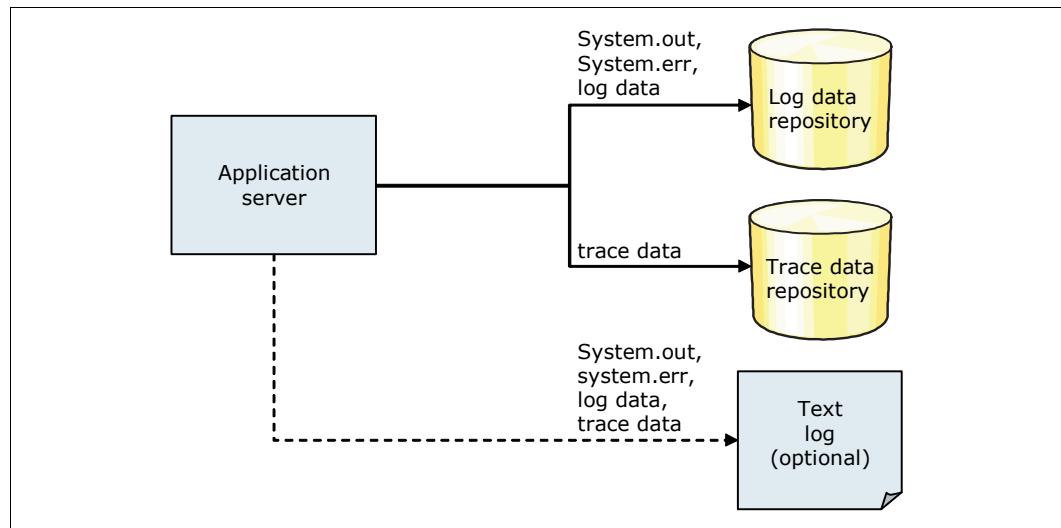


Figure 15-52 HPEL repositories

Now the data is stored in proprietary binary format instead of text as in basic logging (the only exception is for the text log repository). In this way, the following benefits are realized:

- ▶ There is no more text parsing.
- ▶ More data is available because truncation is not necessary.
- ▶ Data is not formatted unless it is needed.
- ▶ No need to clear log files before server start, for example, to diagnosis a problem.
- ▶ Trace speed is improved and more data can be available, and it has half of the impact that basic tracing has.
- ▶ Provides a common solution between z/OS and distributed platform.
- ▶ Applications running with HPEL run faster than running the same application with basic logging.

To read the log and trace records in this new format, a new command line was introduced called **LogViewer**. It reads the data from repositories, formats it, and displays it to the administrator, as shown in Figure 15-53.

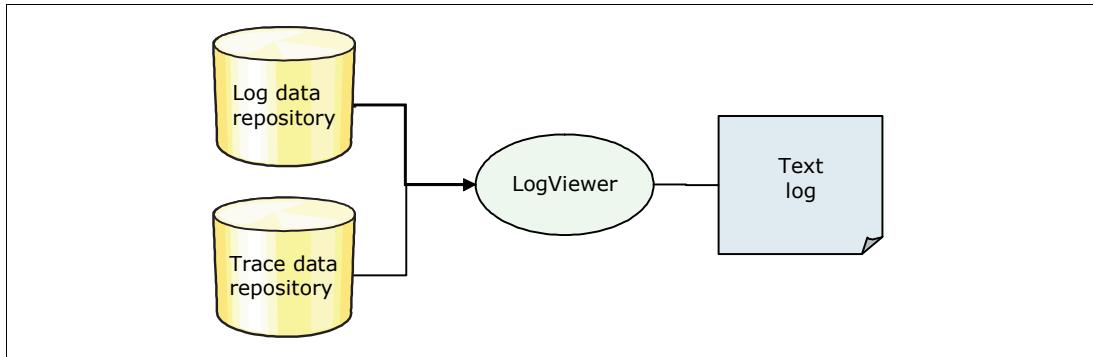


Figure 15-53 logViewer command

To activate HPEL logging and tracing, in the administrative console, click **Troubleshooting → Logs and trace → <your_server> → Switch to HPEL Mode**, as shown in Figure 15-54.

Figure 15-54 Enabling HPEL

Note: A server restart is needed after enabling HPEL logging and tracing.

Viewing data from HPEL repositories

After configuring HPEL, the data can be viewed using the **LogViewer** command from the bin subdirectory of the profile, as shown in Example 15-4.

Example 15-4 logViewer output

```

Using E:\Program Files\IBM\WebSphere\AppServer_80\profiles\Custom03\logs\TradeSe
rver1 as repository directory.
***** Start Display Current Environment *****
WebSphere Platform 8.0.0.0 [ND 8.0.0.0 n1118.03] running with process name sys2C
e1101\sys1Node01\TradeServer1 and process id 6256
Host Operating System is Windows XP, version 5.1
Java version = 1.6.0, Java Compiler = j9jit26, Java VM name = IBM J9 VM
  
```

```

was.install.root = E:\Program Files\IBM\WebSphere\AppServer_80
user.install.root = E:\Program Files\IBM\WebSphere\AppServer_80\profiles\Custom0
3
Java Home = E:\Program Files\IBM\WebSphere\AppServer_80\java\jre
ws.ext.dirs = E:\Program Files\IBM\WebSphere\AppServer_80\java\lib;E:\Program Fi
les\IBM\WebSphere\AppServer_80\profiles\Custom03\classes;E:\Program Files\IBM\We
bSphere\AppServer_80\classes;E:\Program Files\IBM\WebSphere\AppServer_80\lib;E:\P
rogram Files\IBM\WebSphere\AppServer_80\installedChannels;E:\Program Files\IBM\
WebSphere\AppServer_80\lib/ext;E:\Program Files\IBM\WebSphere\AppServer_80\web/h
elp;E:\Program Files\IBM\WebSphere\AppServer_80\deploytool\itp\plugins\com.ibm.e
tools.ejbdeploy/runtime
Classpath = E:\Program Files\IBM\WebSphere\AppServer_80\profiles\Custom03\proper
ties;E:\Program Files\IBM\WebSphere\AppServer_80\properties;E:\Program Files\IBM
\WebSphere\AppServer_80\lib\startup.jar;E:\Program Files\IBM\WebSphere\AppServer
_80\lib\bootstrap.jar;E:\Program Files\IBM\WebSphere\AppServer_80\lib\jsf-nls.ja
r;E:\Program Files\IBM\WebSphere\AppServer_80\lib\Improx.y.jar;E:\Program Files\I
BM\WebSphere\AppServer_80\lib\urlprotocols.jar;E:\Program Files\IBM\WebSphere\Ap
pServer_80\deploytool\itp\batchboot.jar;E:\Program Files\IBM\WebSphere\AppServer
_80\deploytool\itp\batch2.jar;E:\Program Files\IBM\WebSphere\AppServer_80\java\l
ib\tools.jar
Java Library path = E:\Program Files\IBM\WebSphere\AppServer_80\lib\native\win\x
86_32;;E:\Program Files\IBM\WebSphere\AppServer_80\java\jre\bin\default;E:\Progr
am Files\IBM\WebSphere\AppServer_80\java\jre\bin;;E:\Program Files\IBM\WebSpher
e\AppServer_80\lib\native\win\x86_32;E:\Program Files\IBM\WebSphere\AppServer_80
\bin;E:\Program Files\IBM\WebSphere\AppServer_80\java\bin;E:\Program Files\IBM\W
ebSphere\AppServer_80\java\jre\bin;C:\Program Files\Common Files\NetSarang;C:\Pr
ogram Files\SNI Socks\;C:\Program Files\IBM\WebSphere MQ\Java\lib;C:\WINDOWS\syst
em32;C:\WINDOWS;C:\WINDOWS\System32\Wbem;C:\Program Files\IBM\InfoPrint Select;C
:\Notes;C:\Program Files\XLView;C:\lotus\compnent;C:\Utilities;C:\Program Files\
IBM\Personal Communications\;C:\Program Files\IBM\Trace Facility\;C:\WINDOWS\Do
wnloaded Program Files;C:\Program Files\IBM\WebSphere MQ\bin;C:\Program Files\Com
mon Files\Lenovo;C:\Program Files\IBM\HostOnDemand\bin;C:\cygwin\bin;C:\Program
Files\IBM\HostOnDemand\lib\samples\DWunzip;C:\Trabalho\Download\Software;C:\Prog
ram Files\Samsung\Samsung PC Studio 3\;C:\Program Files\Intel\WiFi\bin\;C:\Progr
am Files\ThinkPad\ConnectUtilities;C:\Program Files\IBM\WebSphere\AppServer_70\j
ava\bin;C:\PROGRA~1\IBM\SQLLIB\BIN;C:\PROGRA~1\IBM\SQLLIB\FUNCTION;C:\PROGRA~1\I
BM\SQLLIB\SAMPLES\REPL;C:\PROGRA~1\IBM\LDAP\V6.2\bin;C:\PROGRA~1\IBM\LDAP\V6.2\s
bin;C:\PROGRA~1\IBM\gsk7\bin;C:\PROGRA~1\IBM\gsk7\lib;C:\Program Files\ObjREXX;C
:\Program Files\ObjREXX\OODIALOG;
Orb Version = IBM Java ORB build orb626fp1-20110419.00
***** End Display Current Environment *****
[7/19/11 15:24:52:015 BRT] 00000000 ManagerAdmin I TRAS0017I: The startup tra
ce state is *=info.
[7/19/11 15:24:52:046 BRT] 00000000 ManagerAdmin I TRAS0111I: The message IDs
that are in use are deprecated
[7/19/11 15:24:52:234 BRT] 00000000 ModelMgr I WSVR0800I: Initializing co
re configuration models
[7/19/11 15:24:54:046 BRT] 00000000 ComponentMeta I WSVR0179I: The runtime pro
visioning feature is disabled. All components will be started.
...
Lines removed
...
[7/19/11 15:31:04:031 BRT] 00000021 SystemOut O dsName = java:comp/env/jdb
c/TradeDataSource
[7/19/11 15:31:04:125 BRT] 00000021 webcontainer I com.ibm.ws.webcontainer.Virt

```

```
ualHostImpl addWebApplication SRVE0250I: Web Module DayTrader Web has been bound  
to default_host[*:9080,*:80,*:9443,*:5060,*:5061,*:443,*:9132,*:9133,*:9487,*:9  
134,*:9136,*:9137,*:9491,*:9138].  
[7/19/11 15:31:04:671 BRT] 00000021 ApplicationMg A WSVR0221I: Application sta-  
rted: DayTrader-EE6  
[7/19/11 15:31:04:671 BRT] 00000021 CompositionUn A WSVR0191I: Composition uni-  
t WebSphere:cuname=DayTrader-EE6 in BLA WebSphere:blaname=DayTrader-EE6 started.  
Operation Complete  
Processed 334 records in 66.266 seconds (5.04 records per second).
```

The above example shows all the log details available in repository because **logViewer** was invoked without any parameters. If you want to see the messages from the most recent server run, issue **logViewer -latestInstance**.

There are several options to be used with the **logViewer** command, making it a powerful tool to filter events from log and trace repositories. Here we demonstrate some possibilities that can be used:

- ▶ Showing messages starting at a specific level or higher:
`logViewer -minlevel <message_level>`
- ▶ Showing messages from log and trace for a specific thread:
`logViewer -Thread <thread_id>`
- ▶ Showing messages from log and trace in advanced format:
`logViewer -format advanced`

Example 15-5 shows the advanced format view.

Example 15-5 Advanced format view

```
...  
[7/19/11 15:31:04:031 BRT] 00000021 0 UOW= source=SystemOut class= method=  
dsName = java:comp/env/jdbc/TradeDataSource  
[7/19/11 15:31:04:125 BRT] 00000021 I UOW= source=com.ibm.ws.webcontainer  
class=com.ibm.ws.webcontainer.VirtualHostImpl method=addWebApplication org= prod=  
component= thread=[Default : 1]  
SRVE0250I: Web Module DayTrader Web has been bound to  
default_host[*:9080,*:80,*:9443,*:5060,*:5061,*:443,*:9132,*:9133,*:9487,*:9134,*:  
9136,*:9137,*:9491,*:9138].  
[7/19/11 15:31:04:671 BRT] 00000021 A UOW=  
source=com.ibm.ws.runtime.component.ApplicationMgrImpl org=IBM prod=WebSphere  
component=Application Server thread=[Default : 1]  
WSVR0221I: Application started: DayTrader-EE6  
[7/19/11 15:31:04:671 BRT] 00000021 A UOW=  
source=com.ibm.ws.runtime.component.CompositionUnitMgrImpl org=IBM prod=WebSphere  
component=Application Server thread=[Default : 1]  
WSVR0191I: Composition unit WebSphere:cuname=DayTrader-EE6 in BLA  
WebSphere:blaname=DayTrader-EE6 started.  
[7/19/11 15:36:01:781 BRT] 00000031 W UOW=  
source=com.ibm.ws.sib.utils.ras.SibMessage org=IBM prod=WebSphere  
component=Application Server thread=[System TradeCluster.000-TradeCluster : 0]  
[TradeCluster:TradeCluster.000-TradeCluster] CWSIP0381W: No Response  
received from messaging engine TradeCluster.001-TradeCluster for subscription  
request message.
```

Operation Complete

Processed 335 records in 2.219 seconds (150.969 records per second).

- ▶ Showing messages from log and trace from a specific time range:

```
logViewer -startDate <date/time/timezone> -stopDate <date/time/timezone>
```

More information about the **logViewer.sh** command is available at the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=%2Fcom.ibm.websphere.nd.doc%2Finfo%2Fae%2Frtrb_logviewer.html

Another way available to monitor logs and traces is by using the WebSphere console. Click **Troubleshooting → Logs and trace → <your_server> → View HPEL logs and trace**. Figure 15-55 shows the result.

Viewing log records from server instance July 19, 2011 15:24:52					
Number of records to show: 20					
TimeStamp	Thread ID	Logger	Level	Message	
7/19/11 15:24:52.015	00000000	loggerAdmin	INFO	TRAS0017I : The startup trace state is *=info.	
7/19/11 15:24:52.046	00000000	loggerAdmin	INFO	TRAS0111I : The message IDs that are in use are deprecated	
7/19/11 15:24:52.234	00000000	g.ModelMgr	INFO	WSVR0800I : Initializing core configuration models	
7/19/11 15:24:54.046	00000000	etaDataMgr	INFO	WSVR0179I : The runtime provisioning feature is disabled. All components will be started.	
7/19/11 15:24:54.390	00000000	iderTracker	INFO	com.ibm.ffdc.osgi.ProviderTracker AddingService FFDC1007I : FFDC Provider Installed: com.ibm.ffdc.util.providerFfdcOnDirProvider@2038255	
7/19/11 15:24:54.453	00000000	iderTracker	INFO	com.ibm.ffdc.osgi.ProviderTracker AddingService FFDC1007I : FFDC Provider Installed: com.ibm.ws.ffdc.impl.FfdcProvider	
7/19/11 15:24:55.078	00000000	inInitializer	AUDI	ADMN0015I : The administration service is initialized.	
7/19/11 15:24:56.625	00000000	ServiceImpl	INFO	PLGC0057I : The plug-in configuration service started successfully.	
7/19/11 15:24:56.750	00000000	ponentImpl	INFO	CWPKI0001I : SSL service is initializing the configuration	
7/19/11 15:24:56.765	00000000	VSKeyStore	WARN	CWPKI0041W : One or more key stores are using the default password.	
7/19/11 15:24:56.812	00000000	figManager	INFO	CWPKI0027I : Disabling default hostname verification for HTTPS URL connections.	
7/19/11 15:24:56.859	00000000	sticModule	INFO	CWPKI0014I : The SSL component's FFDC Diagnostic Module com.ibm.ws.ssl.core.SSLDiagnosticModule registered successfully.	
7/19/11 15:24:56.859	00000000	ponentImpl	INFO	CWPKI0002I : SSL service initialization completed successfully	
7/19/11 15:24:56.875	00000000	ConfigHome	INFO	com.ibm.wsspi.rasdiag.DiagnosticConfigHome setStateCollectionSpec RASD0012I : Updating State Collection Spec from L *,*,*=0	
7/19/11 15:24:56.890	00000000	nt.PMIImpl	AUDI	CWPNI1001I : PMI is enabled	
7/19/11 15:24:58.296	00000000	:Component	INFO	CWLR56000I : GAP (Grid Application Placement) Component has initialized successfully on process ManagedProcess.	
7/19/11 15:24:58.421	00000000	SibMessage	INFO	[::] CWSIU0000I : Release: WAS80.SIB Level: d1115.28	

Figure 15-55 HPEL log and trace viewed in WebSphere console

If you expand **Content and Filtering Details** at the top of the page, you can see all of the filter options that can be used when viewing logs and traces, as shown in Figure 15-56.

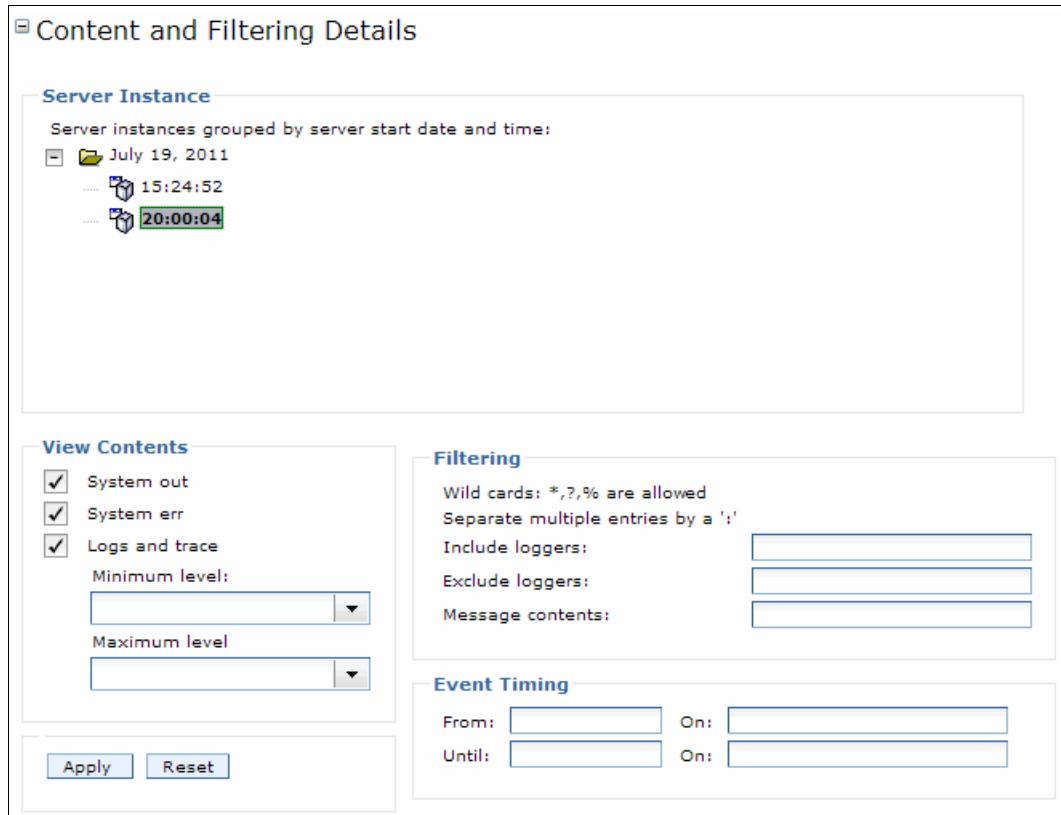


Figure 15-56 HPEL console filtering options

Some more filtering options are available through the buttons at the top of the console messages, as shown in Figure 15-55 on page 596.

More information about HPEL can be found in the Information Center at the following website:
http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=%2Fcom.ibm.websphere.nd.doc%2Finfo%2Fae%2Fae%2Fctrb_HPELOverview.html

15.6.5 Operating system monitoring

Most operating systems have tools for the monitoring of memory, CPU utilization, and disk I/O. They should be monitored and controlled.

15.6.6 Summary of monitoring tips

The following summary lists a simple set of best practices about how to establish a useful monitoring environment:

- ▶ Take time to understand the applications that are being deployed into the environment. Use this knowledge to plan for the kinds of metrics that will be beneficial in understanding application performance.

- ▶ Activate monitoring at the planned level in all testing environments, especially when benchmarking. Although monitoring is vital to your ability to understand an environment, it has a processing impact, and uses CPU, memory, and other resources. Monitoring can impact capacity planning.
- ▶ Use monitoring to understand normal operations and gain an appreciation of what is normal for your systems.
- ▶ Check for differences in your systems after all release changes, especially if full performance testing and benchmarks have not been re-established.
- ▶ Use the basic metrics of PMI as a good starting set of metrics, and then customize them to your needs.

Monitoring WebSphere Application Server alone is not enough. Application server monitoring needs to be part of an overall monitoring strategy.



Part 4

Managing z/OS systems



Performance tuning

Performance tuning is a complex task that spans multiple components and areas with the goal of improving system performance. It is heavily dependent on the hardware, software, and external factors that are employed in your enterprise. Performance numbers advertised in this chapter are based on measurements and projections using standard IBM benchmarks in a controlled environment. Therefore, the performance gains for your system can differ. Using a standard performance methodology and extensive profiling is advisable to understand an application's bottlenecks in production load scenarios.

WebSphere Application Server V8 delivers a highly scalable and highly available platform for applications. It provides multiple tuning options to optimize runtime performance.

This chapter focuses on the z/OS system and suggests methods by which you can improve performance through a combination of product features and application development considerations. It includes the following topics:

- ▶ Introduction to WebSphere Application Server for z/OS V8 performance
- ▶ External factors and z/OS specifics
- ▶ Performance tuning templates
- ▶ 64-bit considerations
- ▶ JVM tuning
- ▶ Connection pool tuning
- ▶ Runtime provisioning
- ▶ Pass by reference
- ▶ Logging and tracing
- ▶ Tuning workload management on z/OS systems
- ▶ Fast response cache accelerator and caching
- ▶ Using WebSphere for z/OS Optimized Local Adapters
- ▶ IBM HTTP Server Status monitoring page
- ▶ Tools

16.1 Introduction to WebSphere Application Server for z/OS V8 performance

WebSphere Application Server for z/OS V8 has seen version to version improvements of around 20 percent compared to WebSphere Application server V7. Some of these improvements are described in this chapter.

End-to-end performance enhancements include:

- ▶ Up to 15% for DayTrader 2.0 running WebSphere V8 in comparison to the V7
- ▶ An additional 43% for DayTrader 2.0 running on a z196 machine in comparison to a IBM System z10™ machine
- ▶ Up to 24% for SOA Benchmark running WebSphere V8 in comparison to V7
- ▶ An additional 40% for SOA Benchmark running on an IBM System z196 machine in comparison to a z10 machine

These numbers are informational only.

WebSphere Application Server for z/OS V8 also includes Java Virtual Machine J9 2.6 and the following sets of native DLL libraries inside the delivered Java run time:

- ▶ 31-bit version
- ▶ 64-bit version
- ▶ 64-bit version optimized for the z196 architecture

The Java specification is still V6, because there is no change in the application layer interfaces, but it is enhanced to V6.0.1. WebSphere Application Server for z/OS V8 also includes a new garbage collection mechanism and an optimized just-in-time (JIT) compiler.

WebSphere Application Server for z/OS V8 is hardware platform aware and introduces z196 hardware usage into the J9 JVM to use new instructions and paradigms, such as out-of-order-execution pipeline and InfiniBand infrastructure for I/O bus or Parallel Sysplex coupling.

This release focuses on the following areas:

- ▶ Optimized runtime performance for programming models
- ▶ Memory optimization and improvement of server startup time
- ▶ Improved installation time, application deployment, and configuration functions
- ▶ Improved runtime performance as a base for stack products
- ▶ Small, simple, and fast development environment

For more detailed information about performance improvements with WebSphere Application Server for z/OS V8, go to the following website:

[http://www-03.ibm.com/support/techdocs/atsmastr.nsf/5cb5ed706d254a8186256c71006d2e0a/981aad32cbab471886257604001e1ec8/\\$FILE/WP101532%20-%20Why%20WebSphere%20Application%20Server%20for%20zOS%20-%20Notes.pdf](http://www-03.ibm.com/support/techdocs/atsmastr.nsf/5cb5ed706d254a8186256c71006d2e0a/981aad32cbab471886257604001e1ec8/$FILE/WP101532%20-%20Why%20WebSphere%20Application%20Server%20for%20zOS%20-%20Notes.pdf)

16.2 External factors and z/OS specifics

External factors, such as collocation and hardware configuration, can play an important role in performance tuning scenarios. In this section, we discuss the more common external factors that can affect performance and provide suggestions for ways to obtain the best performance possible.

16.2.1 Getting the most benefit from collocation

Collocation of the data layer and application layer can be beneficial during application processing on the same system or in the same sysplex. Collocation provides the following benefits:

- ▶ Takes advantage of cross-memory data transfer, reduces overall request latency, improves overall throughput, and reduces overall CPU utilization through the elimination of network traffic handling
- ▶ Allows for the assertion of security identity, maintains the same thread of execution, and manages within a single Workload Manager (WLM) environment

16.2.2 Addressing hardware configuration

Hardware is also influential to overall system performance. Sources of performance boosts include CPU, disks with parallel access volumes support, network cards and switch speed, processor cache, system memory, and machine model. In this section, we list important configuration settings to consider.

Additional resource: For information about hardware configuration settings, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.zseries.doc/info/zseries/ae/tprf_tunehdwc.html

16.2.3 z/OS tuning tips

In this section, we provide information about z/OS tuning tips.

ZFS

Using cache-enabled and sysplex-aware z/OS Distributed File Service zSeries file system (zFS) is preferred over hierarchical file system (HFS). HFS generally provides better response times for all types of operations.

You can verify the cache and average response time per operation type using the following command:

F zfsproc,QUERY,ALL

Additional resource: For more information about ZFS tuning, refer to *z/OS Distributed File Service zSeries File System Implementation z/OS V1R11*, SG24-6580.

Shared library and UNIX System Services

Use the SHRLIBRGNSIZE OMVS parameter carefully with 31-bit servers, because it limits the usable region size system-wide. Review OMVS options to understand the limits that your environment imposes and current usage that it represents. WebSphere Application Server for z/OS provides a way to disable the shared libraries by implementing the standard WebSphere environment variable _BPXK_DISABLE_SHLIB. The application server prints out the SHRLIBRGNSIZE value as part of the message BB000341I during startup.

Example 16-1 BB000341I message printout with shared library size of 16 MB

BB000341I VARIOUS RESOURCE MONITORING DATA:
(64):(16777216)::():():():():():()

Try to eliminate any unnecessary STEPLIBs, and review your PATH statement to prioritize frequently referenced programs. Consider caching your high activity read-only files.

Additional tips: For more UNIX System Services tips, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.zseries.doc/info/zseries/ae/rprf_tunezuss.html

Region size

Verify that there are no limiting factors to region size, such as IEFUSI exit or insufficient MEMLIMIT on your system when preparing to use large heaps. Also verify that sufficient memory is available so that you do not incur a paging penalty.

SMF

Consider turning off collection of SMF type 92 file system related records, which can represent a significant impact for any UNIX-based process operating with a multitude of files. WebSphere Application Server for z/OS creates SMF record type 120. Consider enabling server interval SMF records and container interval SMF records rather than server activity records and container activity records. Also, using only SMF type 120 subtype 9 records that provide a unified request-based view can prove beneficial because this configuration represents less than a 1% impact. Verify that SMF data sets are allocated optimally.

RACF

Use RACLIST for the appropriate Resource Access Control Facility (RACF) classes and use crypto cards where advantageous. Do not activate classes you do not use. Use the virtual lookaside facility (VLF) to cache user identifiers (UIDs) and group identifiers (GIDs) in the COFVLFx member.

RACF tuning: For more specific information about RACF tuning, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.zseries.doc/info/zseries/ae/rprf_tunezsec.html

TCP/IP

Consider setting the NODELAYACKS parameter either on WebSphere Application Server ports or globally on the TCPCONFIG statement. This option specifies that an acknowledgment is returned immediately when data is received that has the PUSH bit set in the TCP header. This setting can improve throughput by as much as 50%. Increase the TCPSENDBFRSIZE and TCPRCVBFRSIZE parameters to at least 64 KB. Optimize your DNS configuration so that lookups for frequently-used servers and clients are being cached.

TCP/IP tuning tips: For more TCP/IP tuning tips, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.zseries.doc/info/zseries/ae/tprf_tunetcip.html

GRS

WebSphere Application Server uses global resource serialization (GRS) for global transaction processing involving multiple servers. In a sysplex environment, set up the coupling facility structure for GRS processing.

For more information, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.zseries.doc/info/zseries/ae/rprf_tunezgrs.html

RRS

Resource Recovery Services (RRS) are crucial to sync point processing. Use coupling facility (CF) log streams or high performing DASD for DASD-only log streams that are allocated with large control interval (CI) sizes for RRS log streams.

Sample RRS log stream definition jobs are provided in the `/WAS_product_image_path/util/zos/JCL/` directory. To off load the entire directory to a partitioned data set, use the `zCopyFilesToPds.sh` script provided in the `/WAS_product_image_path/bin/` directory, as shown in Example 16-2.

Example 16-2 Using the zCopyFilesToPds.sh script to off load directory content

```
/usr/lpp/zWebSphere/V8R0/bin/zCopyFilesToPds.sh -sourceDirectory  
/WAS_product_image_path/util/zos/JCL/ -targetPDS 'YOUR.DATA.SET' -debug
```

For more information about RRS specific tuning, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.zseries.doc/info/zseries/ae/rprf_tunezrrs.html

Language Environment

For best performance, ensure that the CEELPA Language Environment® load library is in your link pack area (LPA) concatenation by specifying it in your active LPALSTxx member.

Do not enable the Language Environment HEAPCHK option unless absolutely necessary, as the performance cost associated with heap diagnostic testing is high.

Language Environment tuning tips: For Language Environment for z/OS specific tuning tips, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.zseries.doc/info/zseries/ae/rprf_tunezle.html

For more information about tuning your Language Environment heap, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Finfo%2Fzseries%2Fae%2Ftprf_tunezleheap.html

16.3 Performance tuning templates

WebSphere Application Server V7 introduced three tuning templates as suggested starting points to improve application server performance. WebSphere Application Server for z/OS V8 contains the latest version of these tuning templates:

- ▶ Peak applies tuning parameters that are well suited for performance test and proof of concept (POC) environments, where peak runtime performance is desired.
- ▶ Development tunes the application server for a development environment where frequent application updates are performed and system resources are typically constrained.
- ▶ Default (Standard) returns the server configuration to the standard defaults.

These templates provide a multitude of the latest preferred practices and tuning recommendations. Review and test the templates for application impact before using them on any option environments.

Attention: Some parameters that are tuned by these scripts (most notably the ORB Pass-By-Reference setting) can impact the functionality of applications.

You can apply these templates by selecting the **Server runtime performance tuning setting** option when you create a WebSphere Application Server V8 profile (using the profile management tool for z/OS), as shown in Figure 16-1.

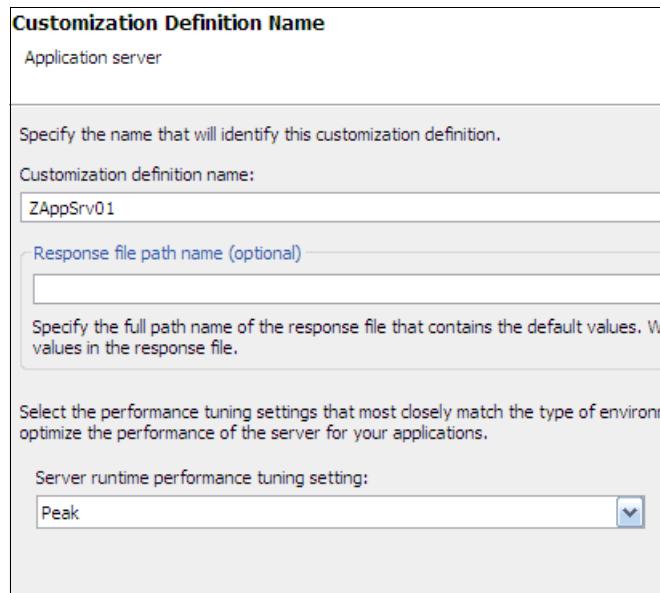


Figure 16-1 Applying profile template during WebSphere Application Server creation in zPMT

You can also apply a template manually by running the `ApplyPerfTuning` Jython script, as shown in Example 16-3.

Example 16-3 Manually invoking the `ApplyPerfTuning` Jython script

```
`${WAS_INSTALL_ROOT}/bin/wsadmin.sh -lang jython -f  
/WAS_product_image_path/scriptLibraries/perfTuning/V70/ApplyPerfTuning.py  
-nodeName nodename -serverName servername -templateFile template.name`
```

Applying your own templates: If you develop your own performance templates based on the provided templates, be aware that customized templates might not be portable with other levels or versions of WebSphere Application Server. The underlying structure of the configuration XML files is prone to change.

For more information about tuning profiles and the values that they introduce, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=%2Fcom.ibm.websphere.nd.multiplatform.doc%2Finfo%2Fae%2Ftprf_tuneappserv_script.html

16.4 64-bit considerations

This section provides a comprehensive overview of 64-bit mode and its impact on the application server's performance. The 64-bit mode was available in WebSphere Application Server V6. Since WebSphere Application Server V7, the installation defaulted to 64-bit mode, because 31-bit mode was deprecated. Consider reconfiguring application servers that were migrated from previous releases to 64-bit mode.

16.4.1 Enablement of 64-bit mode

Ensure that z/OS page set allocations are sufficient and that JCL and Automation procedures do not specify the AMODE=31 parameter on the **start** command before you switch to 64-bit mode. If the detected mode differs from the AMODE setting, an error message of BB000336E is produced, and the server fails to start.

Refer to the following website for further information:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=%2Fcom.ibm.websphere.nd.multiplatform.doc%2Finfo%2Fae%2Fcrun_z64bit.html

To change to 64-bit mode using the administrative console, complete the following steps:

1. Click **Servers** → **Server types** → **WebSphere application servers** → *servername*.

2. On the Configuration tab, select **Run in 64 bit JVM mode**, as shown in Figure 16-2.

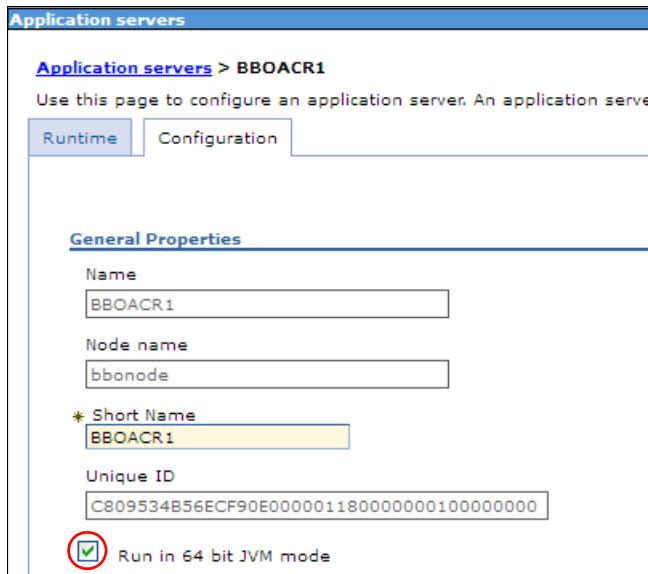


Figure 16-2 Switch to 64-bit mode

3. Click **Apply**.
4. For a Network Deployment configuration, click **Review**, then click **Synchronize changes with nodes**, and click **Save**. Alternatively, click **System administration** → **Nodes**, and click **Synchronize** with the appropriate node or nodes selected or select the **Synchronize changes with Nodes** option in **System administration** → **Console Preferences**.
5. Click **Restart** for the target server in the **Servers** → **Server types** → **WebSphere Application servers** view.

Tip: The admin console is using the following command to facilitate the bit mode change to 64-bit addressing inside the XML configuration files:

```
AdminTask.setJVMMode('[-nodeName nodename -serverName servername -mode bitmode]').
```

For application development purposes, JVM provides the `com.ibm.vm.bitmode` programmatic API, to determine the bit mode setting in which an application server is running.

16.4.2 Effects of switching to 64-bit mode

Running WebSphere Application Server in 64-bit mode allows for heap relief. It grants WebSphere Application Server the ability to run heaps larger than 1 GB and provides access for up to 16 EB of virtual memory. Alternatively, the cost of using 64-bit object references can enlarge your heap by as much as 40%. The inherently bigger objects also affect data locality, and thus contribute to higher translation look-aside buffer (TLB) and data cache miss rates. These higher rates slow dynamic address translation (DAT) and affect application performance.

The compressed references and large page support features can provide relief for reduced throughput and memory footprint growth incurred when migrating from 31-bit JVM to 64-bit JVM. With these two JVM properties turned on, 64-bit environments can match and out perform previous 31-bit environments. These features are not turned on by default because they have software and hardware requirements.

Compressed references

Compressed references is a method for managing object pointers with the JVM. Some workloads in 64-bit environments have shown objects increased by up to 45% because the object header and object references doubles in width. Compressed references mode is usable for JVM heaps of up to 30 GB on z/OS. It reduces the size of the 64-bit object pointer to 4 bytes (word).

In compressed references mode, object header referenced data (such as class-related data and thread data) are allocated below the 2 GB bar. This allocation enables all the references to be 32-bit and no padding to occur in the object header, as illustrated in Figure 16-3.

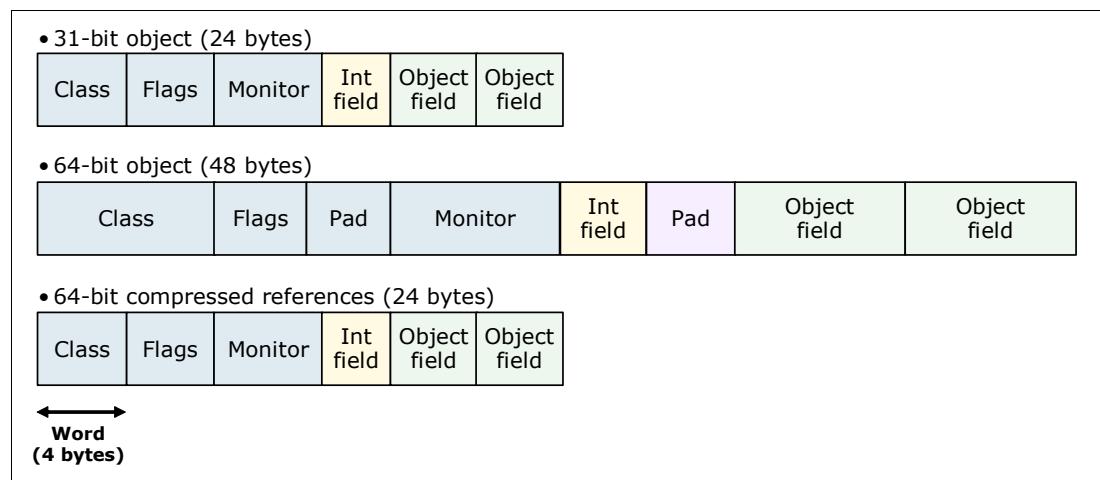


Figure 16-3 Header and object references in 31-bit and 64-bit mode

For object data references, we use a paradigm where 64-bit heap allocations are aligned on double-word boundaries, leaving the lowest 3 bits at zero. This allows for a right-shift compression for object references. Adequate compression is chosen based on the top of Java heap specified by the `-Xmx` maximum Java heap option value, as listed in Table 16-1.

Table 16-1 Compressed references algorithm choice based on maximum heap size

Max heap size specified by the <code>-Xmx</code> option	Top of the heap located	Shift amount used for object reference compression
2 GB or less	below 2^{32}	0
6 GB or less	below 2^{33}	1
14 GB or less	below 2^{34}	2
30 GB or less	below 2^{35}	3
Greater than 30 GB	above 2^{35}	Only supported without compressed references

Using a shift-by-1 compression scheme: In a shift-by-1 compression scheme (2 GB to 6 GB Java heap), the IBM JVM on System z can, in many cases, remove the cost of the shift operation using the IBM System z architecture base index register memory references. Explicit shifting is still required for array accesses and the garbage collection run time. As such, the performance of the shift-by-1 compression scheme is still worse than that of the shift-by-0 compression scheme. However, shift-by-1 does typically perform better than shift-by-2 or shift-by-3 compression schemes.

For further information about compressed references mode, go to the following website:

<http://public.dhe.ibm.com/partnerworld/pub/whitepaper/1d71a.pdf>

To change to compressed references mode for the JVM, complete the following steps:

1. Click **Environment** → **WebSphere variables**.

You can also enable this variable on a process basis by clicking **Servers** → **Server Types** → **WebSphere application servers** → **server_name** → **Java and process management** → **ProcessDefinition** → **servant** → **Environment entries**.

2. Select a scope based on your desired affected environment.
3. Click **New**, and specify IBM_JAVA_OPTIONS in the name field.
4. Add or append the **-Xcompressedrefs** value, as shown in Figure 16-4.
5. Click **Apply**.

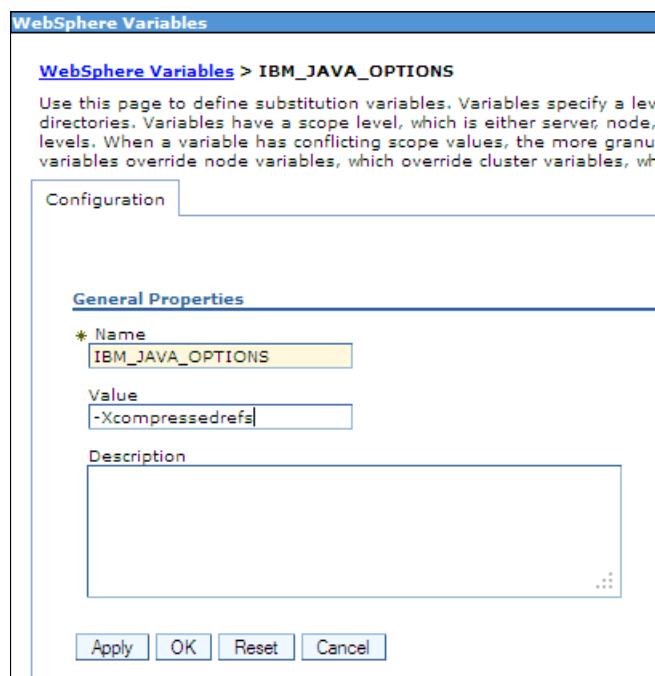


Figure 16-4 Enabling compressed references

6. For a Network Deployment configuration, click **Review**, then click **Synchronize changes with nodes**, and click **Save**. Alternatively, click **System administration** → **Nodes**, and click **Synchronize** with the appropriate node or nodes selected or select the **Synchronize changes with Nodes** option in **System administration** → **Console Preferences**.
7. Click **Restart** for the affected server in the **Servers** → **Server types** → **WebSphere Application servers** view.

Note: Be aware that specifying **-Xcompressedrefs** as a generic JVM argument (on IBM System z9® or earlier) or for heaps over 30 GB produces an error and JVM fails to start.

Use the **F servername, JAVACORE** command or click **Troubleshooting → Java dumps and cores** to produce the dump file with list of JVM arguments or VerboseGC output, as shown in Example 16-4, to confirm the usage of the feature.

Example 16-4 VerboseGC information for compressed references

```
<initialized id="1" timestamp="2011-07-09T12:56:32.600">
  <attribute name="gcPolicy" value="-Xgcpolicy:gencon" />
  <attribute name="maxHeapSize" value="0xc0000000" />
  <attribute name="initialHeapSize" value="0xc0000000" />
  <attribute name="compressedRefs" value="true" />
  <attribute name="compressedRefsDisplacement" value="0x0" />
  <attribute name="compressedRefsShift" value="0x1" />
<vmargs>
  <vmarg name="-Xcompressedrefs" />
```

Note: When running compressed references, a different JVM initializes; therefore, when analyzing problems, you need to enable compressed references with the dump extractor to analyze dumps that are produced by the JVM.

For more information about enabling the feature, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=/com.ibm.websphere.zseries.doc/info/zseries/ae/tprf_tunejvm_v61.html

Requirements: The compressed references feature has the following requirements:

- ▶ IBM z/OS V1R8 or later
- ▶ IBM Developer Kit for Java 6, 64-bit edition, SR4 or later (APAR PK82091)
- ▶ IBM z/OS support APAR OA26294

Large page support

With the evolution of 64-bit environments, applications produce a higher virtual footprint and use freely the pointers to 2^{64} virtual storage map. The mapping of real or absolute real storage ranges to virtual one is done by using hardware dynamic address translation (DAT) structures.

To avoid DAT hardware access to main storage tables for every request, the translation look-aside buffer (TLB) was introduced. TLBs are fast array memory within each processing unit. In cases where TLB hits, translation processing is much faster. If a TLB miss occurs, calculating the virtual-to-real mapping of a page can take several dozen cycles. Even with 512 TLB entries per processing unit in z10, EC model it is not enough to generate high TLB hit ratio in today's virtual storage intensive applications.

Therefore, z10 mainframes now support large pages with enhanced-DAT architecture. One large page can hold 256 times more virtual memory than a 4 KB page, guaranteeing fewer TLB misses and TLB entries to represent the data footprint of the application.

Large pages are allocated above the 2 GB virtual bar. They are 1 MB in size fixed (non-swappable) pages and backed up by real storage. As such, they are not available to 31-bit applications, and application executable code cannot reside in them. The amount of storage for large pages is defined by the LFAREA IEASYSxx system parameter and can be up to 80% of real memory. You must perform an IPL to introduce this parameter.

If the system is constrained for 4 KB pages and the large pages are still available, it will convert the available large pages into 4 KB pages. The system can also swap 4 KB pages if more large pages are needed. When allocated, however, large pages cannot be converted into 4 KB pages because they are fixed.

Java applications that allocate lots of Java objects and cause frequent garbage collection will typically benefit from large pages. The data access pattern can also affect the benefits of large pages.

For more information, refer to the following website:

<http://public.dhe.ibm.com/partnerworld/pub/whitepaper/1d71a.pdf>

To enable this feature, complete the following steps:

1. Click **Environment** → **WebSphere variables**.

You can also enable this variable on a by process basis by clicking **Servers** → **Server Types** → **WebSphere application servers** → **server_name** → **Java and process management** → **ProcessDefinition** → **servant** → **Environment entries** or by specifying the **-Xlp** option on the generic JVM argument for the process.

2. Select a scope based on your desired affected environment.
3. Click **New** and specify IBM_JAVA_OPTIONS in the name field.
4. Add or append the **-Xlp** option on the Value field, as shown in Figure 16-5.
5. Click **Apply**.

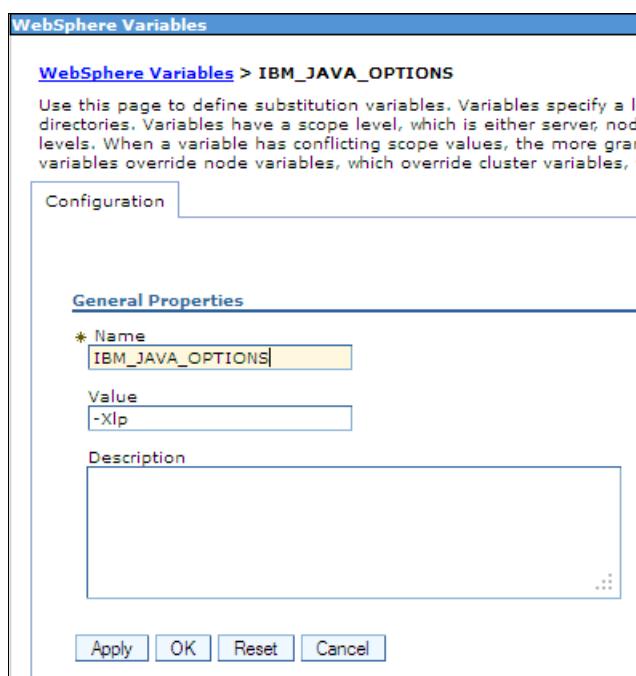


Figure 16-5 Enabling large pages support

6. For a Network Deployment configuration, click **Review**, click **Synchronize changes with nodes**, and click **Save**. Alternatively, click **System administration** → **Nodes**, and click **Synchronize** with the appropriate node or nodes selected or select the **Synchronize changes with Nodes** option in **System administration** → **Console Preferences**.
7. Click **Restart** for the target server in the **Servers** → **Server types** → **WebSphere Application servers** view.

Note: If large page support is not enabled or if there are not enough large pages to satisfy the allocation request during JVM initialization, it falls back to the default usage of 4 KB pages.

Use **F servername,JAVACORE** command or click **Troubleshooting** → **Java dumps and cores** to produce the dump file with a list of JVM arguments. Alternatively, use VerboseGC output, as shown in Example 16-5, to confirm the usage of the feature.

Example 16-5 VerboseGC information for large pages

```
<initialized id="1" timestamp="2011-07-09T12:56:32.600">
  <attribute name="gcPolicy" value="-Xgcpolicy:gencon" />
  <attribute name="pageSize" value="0x1000" />
  <attribute name="requestedPageSize" value="0x100000" />
  <attribute name="gcthreads" value="2" />
<vmargs>
  <vmarg name="-Xlp" />
```

Requirements: This feature requires the following components:

- ▶ IBM z/OS V1R9 or later
- ▶ IBM Developer Kit for Java6, 64-bit edition, SR2 or later (APAR PK82091)
- ▶ IBM System z hardware using the IBM System z10® processors or later
- ▶ IBM z/OS support APAR OA20902 (only needed for IBM z/OS V1R9)
- ▶ IBM z/OS support APAR OA25485

For more information about enabling the feature, refer to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=/com.ibm.websphere.zseries.doc/info/zseries/ae/tprf_tunejvm_v61.html

16.5 JVM tuning

This section provides basic tuning tips for JVM performance.

16.5.1 Default garbage collection

WebSphere Application Server for z/OS V8 changes the default policy from using the **optthruput** option to using the **gencon** option, which was introduced with IBM Java 5.

With the **optthruput** option, all objects were allocated from one large contiguous heap that was shared by all threads. When a garbage collection was invoked, the entire heap was scanned, and unreferenced objects were marked and swapped. This option was best for applications that required optimal throughput but introduced longer garbage collection pause times.

The **gencon** option is a Generational Concurrent Garbage Collector and reduces these longer pause times by dividing the heap into two sections, the *nursery* and *tenured* areas. New objects are always allocated in the nursery part of the heap. When objects age (have been scanned multiple times by garbage collection), they are moved to the tenured area. The JVM constantly monitors the size of both the nursery and tenured areas and adjusts the size of each based on garbage collection frequency and pause times.

This philosophy is based on the observation that most objects are short term. Thus, by allocating them in separate heaps, scanning becomes more efficient because none of the longer-lived objects are scanned. This type of garbage collection policy is ideal for transactional workloads or heavily cached workloads because it reduces the impact of garbage collection by referencing these long-lived objects infrequently.

For more information about the Generational Concurrent Garbage Collector policy, go to the following website:

http://publib.boulder.ibm.com/infocenter/javasdk/v6r0/topic/com.ibm.java.doc.diagnostics.60/diag/understanding/mm_gc_generational.html

16.5.2 General JVM suggestions

This section provides general JVM suggestions.

JVM heap

Verify your current and actively used heap size using the following command. Example 16-6 shows the output of this command.

F servername,DISPLAY,JVMHEAP

Example 16-6 Output of the DISPLAY,JVMHEAP command

```
BB000201I JVM HEAP INFORMATION FOR SERVER BB0C003/BBOACR1/STC05742
BB000202I (STC05742) HEAP(NURSERY), COUNT(00000026), FREE 000
STORAGE(0000000044457E0), TOTAL STORAGE(000000005410000)
BB000202I (STC05742) HEAP(MATURE), COUNT(00000000), FREE 001
STORAGE(0000000090987D8), TOTAL STORAGE(00000000C000000)
BB000204I JVM HEAP INFORMATION FOR SERVER BB0C003/BBOACR1/STC05742
COMPLETE
```

Do not increase the heap size if paging is occurring for your application server. You can use administrative console or generic JVM arguments to set the minimum and maximum heap sizes:

- ▶ The following JVM argument sets the initial Java heap size:
-Xms<size>
- ▶ The following JVM argument sets the maximum Java heap size:
-Xmx<size>

To set the minimum and maximum heap sizes through Administrative console, complete the following steps:

1. Click **Servers** → **Server Types**, and click **WebSphere application servers**, as shown in Figure 16-6.



Figure 16-6 WebSphere application servers in Servers expandable

2. In the Application servers view, select your desired application server, as shown in Figure 16-7.

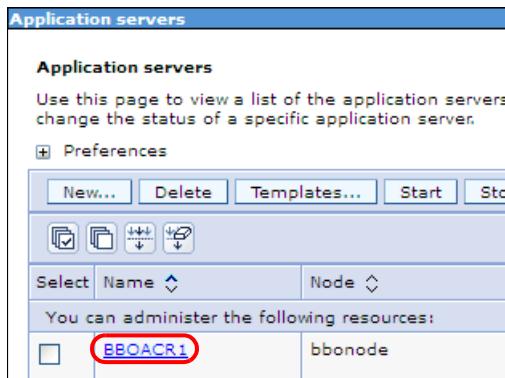


Figure 16-7 WebSphere Application Server view server selection

3. Navigate to Server Infrastructure, and expand the Java and Process Management option. Select **Process definition**, as shown in Figure 16-8.



Figure 16-8 Specific application servers configuration window

4. Select the desired process type. In this demonstration, we chose the servant process, as shown in Figure 16-9.

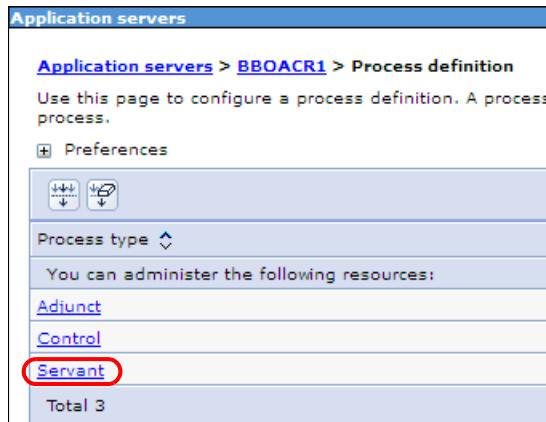


Figure 16-9 Servant process inside the Process definition view

5. Click **Java Virtual Machine** on the Additional Properties window, as shown in Figure 16-10.



Figure 16-10 Java Virtual Machine option on the Servant process view

- Specify an initial heap size and a maximum heap size for the servant process JVM, as shown in Figure 16-11.

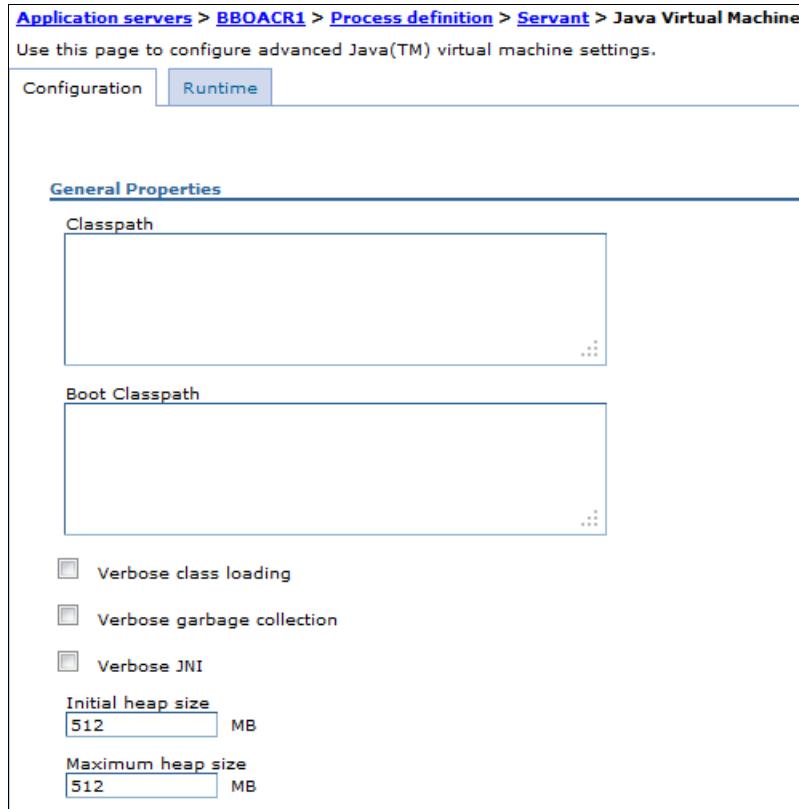


Figure 16-11 Java Virtual Machine view of the servant process

Specifying the same initial and maximum heap size can avoid compaction and expansion phase of the garbage collection and, in turn, benefit performance. However, using the same size can also have a negative impact on the heap fragmentation and can increase garbage collection cycles compared to using a lower initial heap size. Test your application storage behavior and analyze garbage collection traces to find the optimum settings for your application server environment. Garbage collection should run no more than 5% of the total time.

Preferred practice: For a `gencon` garbage collection policy with JVM heap over 1 GB, consider changing the default values on the `IBM_JAVA_OPTIONS` environment variable as follows:

- ▶ `-XX:NewSize=640m`
- ▶ `-XX:MaxNewSize=640m`
- ▶ `-XX:SurvivorRatio=16`

The balanced policy is designed for large heap sizes and can be useful when you use the `gencon` policy with a heap size greater than 4 GB or if you use very large arrays.

For more information about balanced garbage collection policy, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.java.doc.60_26/J9/GC/mm_gc_balanced_when.html

Application behavior

Verify the behavior of your application in relation to the Java memory usage to ensure that your application meets the following suggestions:

- ▶ Adheres to preferred programming practices
- ▶ Does not overutilize objects
- ▶ Does not create memory leaks

JVM arguments

During Java configuration, verify that no unnecessary classes are present in the class path. At the same time, make sure that no class is missing because the class search can be I/O intensive. The classes that are referenced most frequently should be located near the front of the path.

You can use several general JVM properties to speed up the server start time at a cost of lower runtime performance or disabling functions. You can use this approach for development environments where start speed is preferred over runtime optimization. However, carefully review the following options and their consequences:

- ▶ **-Xquickstart**
JVM uses a lower optimization level for class method compiles.
- ▶ **-Xverify:none**
JVM skips the class verification stage during class loading. Corrupted or invalid class data is not detected.
- ▶ **-Xgcthreads<number of processors>**
Garbage collection uses several threads during collection.
- ▶ **-Xnocompactgc**
Disables the heap compaction garbage collection operation.

For more information about JVM arguments, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=/com.ibm.websphere.express.doc/info/exp/ae/urun_rconfproc_jvm.html

Dump processing

To limit dump processing, export these environment variables on the WebSphere Application Server profile:

```
JAVA_DUMP_OPTS=ONANYSIGNAL(JAVADUMP[3],SYSDUMP[1]),ONINTERRUPT(NONE)  
JAVA_DUMP_TDUMP_PATTERN=DUMP.D%y%m%d.T%H%M%S.%job
```

Additional tips: For more information about JVM specific tuning, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.zseries.doc/info/zseries/ae/tprf_tunejvm_v61.html

16.6 Connection pool tuning

For more information about connection pool tuning, refer to 13.2.2, “Data source tuning” on page 499.

16.7 Runtime provisioning

Runtime provisioning is a feature introduced in WebSphere Application Server V7. It focuses on improving application server startup time by providing intelligent analysis of application set and server configuration to determine the needed subset of components to be loaded. This feature does not load an entire runtime library and thus decreases the memory footprint and shortens the start time.

There is no need for administrators and application developers to modify any processes to take advantage of the runtime provisioning. To turn this feature on, complete the following steps:

1. Click **Servers** → **Server Types** → **WebSphere application servers**, as shown in Figure 16-12.



Figure 16-12 WebSphere application servers window

2. In the **WebSphere application servers** view, select your desired application server, as shown in Figure 16-13.

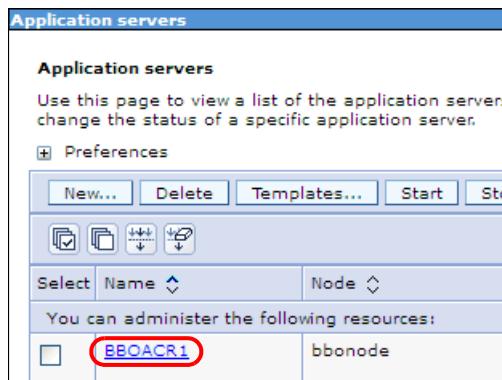


Figure 16-13 WebSphere application server view server selection

3. Select **Start component as needed** to enable runtime provisioning in this server, as shown in Figure 16-14.

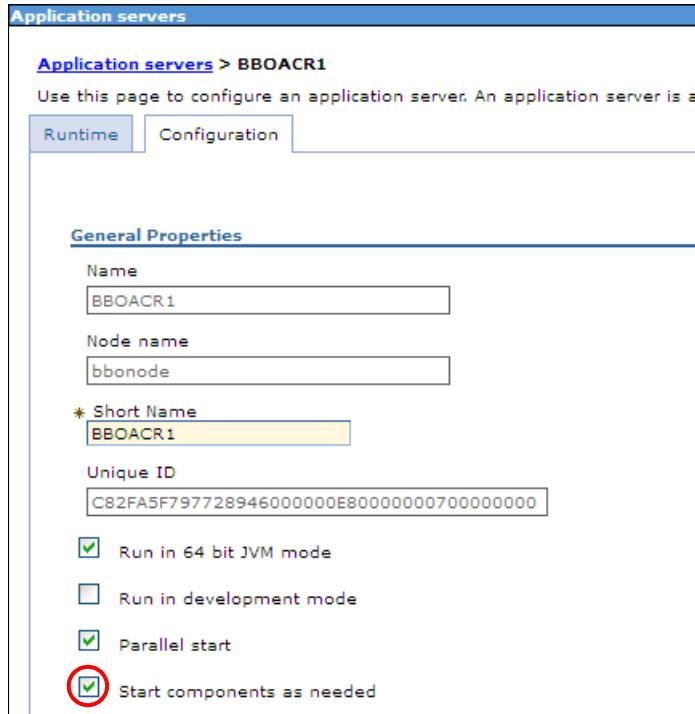


Figure 16-14 Application servers configuration window

WebSphere Application Server V8 provides a way to disable WebSphere MQ functionality inside the application server.

For more information about this feature, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Finfo%2Fzseries%2Fae%2Ftmm_wmq_disable.html

16.8 Pass by reference

Refer to 13.4.2, “The pass by reference parameter” on page 512 for more information about this ORB service option.

To learn how to enable Pass message payload by reference for JMS, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Finfo%2Fzseries%2Fae%2Fcjn_passbyref_steps.html

To learn how to use PassByReference optimization in SCA applications, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.zseries.doc/info/zseries/ae/tsca_passby_ref.html

16.9 Logging and tracing

This section focuses on the logging and tracing options that are specific to the z/OS system and the new standardized model of logging that is introduced in WebSphere Application Server V8.

16.9.1 HPEL overview

High Performance Extensible Logging (HPEL) is a complete solution for standardized logging and tracing in WebSphere Application Server V8. It is an alternative to the existing log and trace facilities that are offered on the z/OS platform, which use LogStreams, Component Trace, Job Entry Subsystem, Hierarchical File System, or other facilities. It is a new strategic logging solution delivered across platforms. It does not include native traces.

HPEL uses mechanisms with data repositories where logging and tracing data is stored all in one place in binary format. To improve performance, data is not shared across instances and is not converted by LogViewer to plain text unless needed. HPEL provides an optional text log function for debugging convenience, where log or trace data can be written to text log file in plain text format immediately. You can use the administrative console or the LogViewer commands to view the HPEL collected logs and traces.

HPEL outperforms existing log and trace facilities and can have performance benefits for log intensive applications; however, HPEL is not enabled by default. An HPEL API is provided in the `com.ibm.websphere.logging.hpel` package to simplify development of tools for log and trace processing.

For more information, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.zseries.doc/info/zseries/ae/ctrb_HPELOverview.html

16.9.2 Enabling HPEL mode

To enable HPEL logging, refer to 18.4.5, “Advanced logging” on page 723. You should disable text logging after enabling HPEL to improve performance.

16.9.3 z/OS logging and tracing tips

For WebSphere Application Server for z/OS in basic logging mode, the `SystemOut.log`, `trace.log`, and `STDOUT` streams are redirected to the `SYSPRINT` ddname by default. The `System.err`, and `STDERR` streams are redirected to the `SYSOUT` ddname. WebSphere Application Server for z/OS cataloged procedures associate these ddnames with `print (SYSOUT=*)` data sets, causing message logs to go into WebSphere Application Server job output.

In basic mode, ensure that your logging and tracing level is not unnecessarily high by completing the following steps:

1. Navigate to **Troubleshooting** → **Logs and trace**, as shown in Figure 16-15.



Figure 16-15 WebSphere Application Server Logs and trace view

2. Click the target application server name, as shown in Figure 16-16.

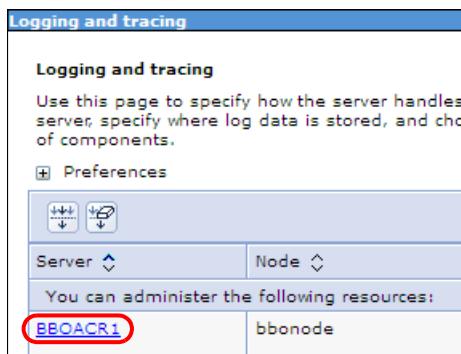


Figure 16-16 Application server choice on Logging and tracing view

3. Click **Change log detail levels**, as shown in Figure 16-17.

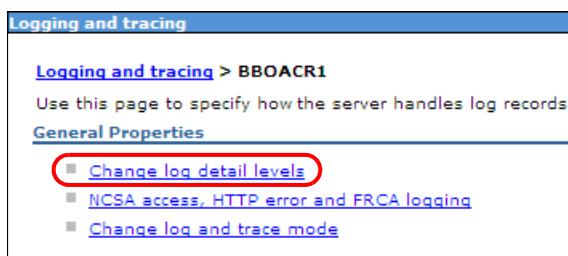


Figure 16-17 Change log detail levels on Application server Logging and tracing view

- Verify your logging and trace detail level. The default logging level is *=info, as shown in Figure 16-18.

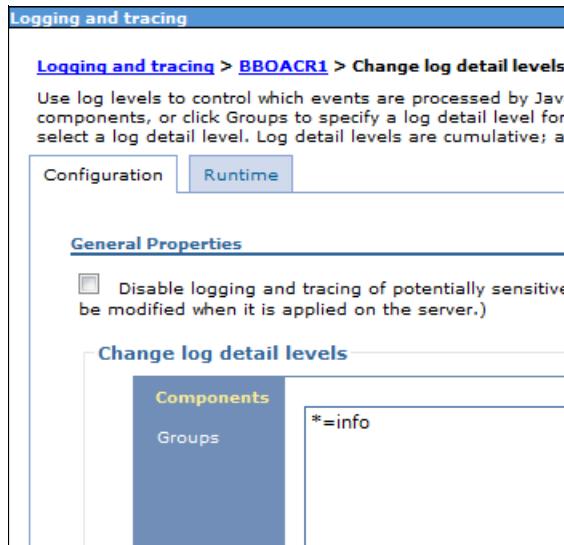


Figure 16-18 Configuration view of the change log detail level view

Preferred practice: Start the trace string from the most broad component groups, and then select more specific traces. The advantage to this approach is that the trace settings for classes or packages that are contained in a larger group are specified correctly by including them later in the trace string.

The logging string is processed from left to right. During the save processing optimization, part of the logging string might be modified or removed if the levels they configure are overridden by another part of the logging string.

For more information about trace controls, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.zseries.doc/info/zseries/ae/rtrb_tracecontrols.html

In addition to the administrative console, you can also use WebSphere Application Server V8 modify commands to change the tracing levels dynamically:

- ▶ **F *servername*,TRACEALL={0,1,2,3}**
Establishes a general trace level for the server.
- ▶ **F *servername*,TRACEINIT**
Resets the trace settings to the initial trace settings.
- ▶ **F *servername*,TRACENONE**
Turns off all trace settings.
- ▶ **F *servername*,TRACETOSYSPRINT={YES|NO}**
Selects whether to send the trace to SYSPRINT.

- **F *servername*,TRACEBASIC=x** and **F *servername*,TRACEDETAIL=x**

Sets the trace level, where *x* is a value from Table 16-2 that represents a component.

Note: Subcomponents, specified by numbers, receive detailed traces. Other parts of the product receive tracing as specified on the TRACEALL parameter.

Table 16-2 TRACEBASIC and TRACEDETAIL components values

Value	Product component
0	RAS
1	Common utilities
3	COMM
4	ORB
6	OTS
7	Shasta
9	z/OS Wrappers
A	Daemon
E	Security
F	Externalization
J	JRAs
L	Java EE

On z/OS, you can use CTRACE as your tracing option. You can use the CTRACE facilities in WebSphere Application Server for z/OS to manage the collection and storage of trace data. CTRACE data is written to address space buffers in private (pageable) storage, which can be formatted using an interactive problem control system (IPCS) if a dump file of the address space is created.

CTRACE data can also be written to trace data sets on disk or tape using an external writer procedure. This procedure uses the BBOCTIxx parmlib member and BBOWTR procedure with samples provided in the */WAS_product_image_path/util/zos/JCL/* directory. Because CTRACE uses system resources efficiently, you can collect valuable trace data with minimal impact on performance. Ensure that CTRACE for all components is configured to MIN or OFF when not used.

For information about how to set up CTRACE support, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.zseries.doc/info/zseries/ae/ttrb_setupCTRACE.html

You can display the CTRACE status using the following command. Example 16-7 shows the output of the command.

D TRACE,COMP=*cell_short_name*

*Example 16-7 D TRACE,COMP=*cell_short_name* command output*

```
-D TRACE,COMP=BBOCCELL
IEE843I 00.50.47 TRACE DISPLAY 792
          SYSTEM STATUS INFORMATION
```

```

ST=(ON,0001M,00002M) AS=ON BR=OFF EX=ON MO=OFF MT=(ON,024K)
COMPONENT      MODE BUFFER HEAD SUBS
-----
BBOCELL        ON          HEAD    11
ASIDS          *NOT SUPPORTED*
JOBNAMES       *NOT SUPPORTED*
OPTIONS        MINIMUM
WRITER         *NONE*

```

As in previous releases, you can set up an error log using the coupling facility or using a DASD-only log stream for a single WebSphere Application Server or shared error log for several servers. If you decide to use the feature, use a coupling facility log stream that is shareable across the sysplex.

You can view the error log records using the log browse utility (BBORBLOG) located in the `/WAS_product_image_path/util/zos/EXEC/` directory.

For more information about using BBORBLOG EXEC, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Finfo%2Fzseries%2Fae%2Frprf_tuneztrace.html

If you are using Transaction XA partner logs or SIP recovery log streams, use coupling facility log streams. For more information, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.installation.zseries.doc/info/zseries/ae/cins_logstrm.html

For information about JDBC tracing, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.zseries.doc/info/zseries/ae/rtrb_jdbccomp.html

For internal tracing tips, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Finfo%2Fzseries%2Fae%2Frprf_tuneztrace.html

16.10 Tuning workload management on z/OS systems

This section discusses the tailoring needs and benefits of workload management on z/OS systems in conjunction with WebSphere Application Server.

16.10.1 The concept of workload management on z/OS systems

Workload management consists of categorizing, prioritizing, routing, and reporting on requests. On z/OS systems, the Workload Manager (WLM), which is an operating system component, allows the system programmer to configure the rules that are used to differentiate and organize units of work. When classified, WLM can determine the service level agreement (SLA) for this type of work. WLM then assigns system resources to units of work from that workload, making a best attempt to allow all work to meet the specified goals. Less important work is sacrificed to meet the goals of more important work if necessary.

WebSphere Application Server for z/OS uses this facility for workload classification by default. It uses a controller-servant region mechanism and can run several instances of servant regions for a single WebSphere Application Server, called a *dynamic servant region expansion*. WLM manages these servant instances in a dynamic application environment. The instances are started as dictated by workload within the guidelines of the WLM MIN/MAX SERVANT parameters. WLM then routes work to the appropriate servant based on the classification rules. It also uses sophisticated algorithms to choose the best candidate if multiple servants are bound for the same service class.

All application work that runs inside WebSphere Application Server runs under WLM-managed enclaves. Proper WLM goals can significantly affect application throughput. To allow servants to start in parallel, use the `wlm_servant_start_parallel` custom property.

For more complex scenarios and to understand the overall workload goals in your organization, consult with your WLM systems programmer.

For more information about WebSphere Application Server and the z/OS WLM, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.nd.multiplatform.doc/info/ae/ae/crun_wlmzos.html

16.10.2 Classification rules

The work priority of location service daemons and controllers should be higher than the priority for servants. Work inside controllers and daemons is categorized based on started task control (STC) classification rules, where a servant's classification is more complex.

The WebSphere Application Server's servant life cycle with WLM includes the following phases:

- ▶ The start phase before the servant is bound to a service class queue
- ▶ The initialized phase after the servant is bound to a service class queue

During the first phase, the servant is guided by the STC classification rules, which are most likely of the velocity type. During the second phase, application work runs in enclaves that are guided by CBtypeclassification rules. Specify achievable response time goals with a percentile here. You can also classify work using the collection name of the cluster. WLM performs better with less service classes.

Velocity goals for application work are not meaningful and should be avoided.

If running in multi-instanced servant environment, use WLM classification and define unique service classes for different priority work that is running in the same server. Also be sure to allow for at least one servant per unique service class.

Using non-enclave threads: z/OS V1.12 introduced a parameter to bind non-enclave threads to the goals of the STC service class for the life of the servant. This IEAOPTxx parmlib member parameter is called MANAGENONENCLAVEWORK. It applies to supporting tasks (such as garbage collection threads) inside the servant. It can be set dynamically and WLM policy is refreshed as part of the SET OPT=xx command.

For more information about controlling WLM dynamic application environment, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=%2Fcom.ibm.websphere.nd.doc%2Finfo%2Fae%2Fcdat_resourcead.htm

For WebSphere Application Server work request management, go to the following website:
http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=/com.ibm.websphere.nd.multiplatform.doc/info/ae/ae/crun_wlmwork.html

For more information about WLM tuning tips for z/OS, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Finfo%2Fzseries%2Fae%2Frprf_tunezwlm.html

You should also add a classification rule for OMVS type of work for BPXBATCH to prevent elongation of the start process and execution of the applyPTF.sh script. For more information, go to the following website:

<http://www-03.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/TD102730>

16.10.3 Classification XML

To help WLM identify the work that is running inside the servant and to provide finer classification rules for applications, WebSphere Application Server uses workload classification XML. It is introduced by specifying a path to the XML for the `wlm_classification_file` WebSphere variable, as shown in Figure 16-19.

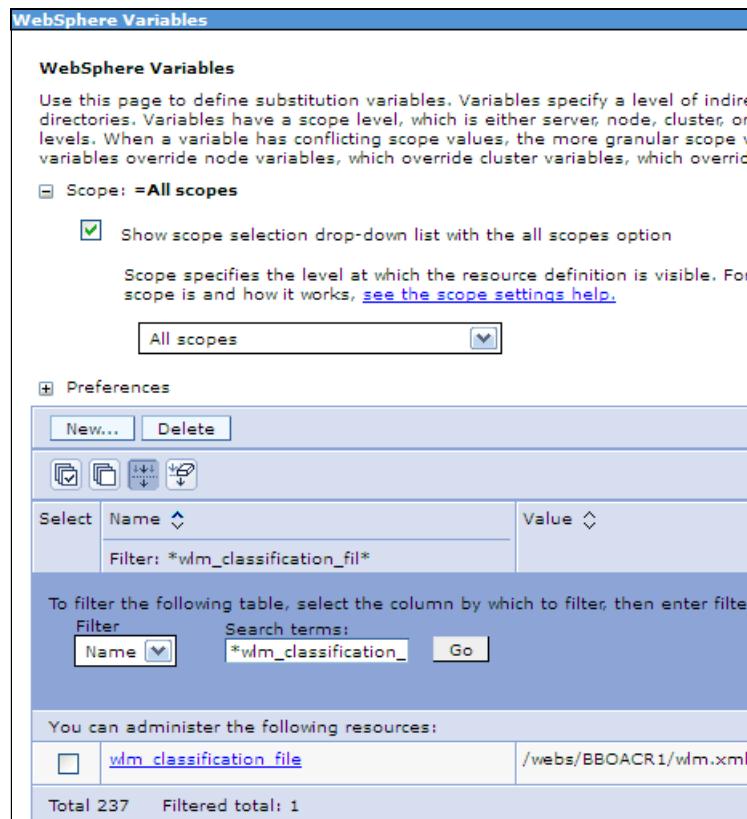


Figure 16-19 The `wlm_classification_file` environment variable

Example 16-8 shows a simple `classify.xml` file for use with sample applications.

Example 16-8 Simple `classify.xml` example

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE Classification SYSTEM "Classification.dtd" >
```

```

<Classification schema_version="1.0">

    <InboundClassification type="http"
        schema_version="1.0"
        default_transaction_class="WSHTTP">
        <http_classification_info transaction_class="HTTP"
            host="wtsc60.itso.ibm.com"
            description="ITSO host">
        <http_classification_info transaction_class="WS_XML"
            uri="/xmlsamples/*"
            description = "XML" />
        <http_classification_info transaction_class="WS_PLANT"
            uri="/PlantsByWebSphere/*"
            description = "PLANTS" />
        </http_classification_info>
    </InboundClassification>
</Classification>

```

For information about the full syntax of classification XML, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=/com.ibm.websphere.zseries.doc/info/zseries/ae/rrun_wlm_tclass_dtd.html

WebSphere Application Server for z/OS V8 provides increased reliability, availability, and serviceability (RAS) granularity with request level classification. It allows you to specify RAS attribute values for HTTP, IIOP, optimized local adapter, and certain MDB requests.

For more information about RAS granularity, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Finfo%2Fzseries%2Fae%2Frrun_ras_granularity.html

16.10.4 Commands and tools

To analyze and monitor your WLM environment, you can use the WLM Work Queue Viewer (WQUEUE) tool. This ISPF-based tool can help you in displaying the application environments that are used on your system, as shown in the Example 16-9. For more information, refer to the link in 16.14, “Tools” on page 644.

Example 16-9 WQUEUE main panel

TIME: 12.Jul.2011 10:22:04 REL: HBB7770 SYS: SC60 PLEX: PLEX60 ENV: TSO

Select: ALL

Application Environment Monitor

AppEnv	Type	SubName	WMAS	Del	Dyn	NQ	QLen	Str	Hav	Unb	Trm	Min	Max	ICnt
BB0C003	CB	BBOACR1	004A	Off	On	1	0	0	1	0	0	6	18	0
BB0C004	CB	BBOACR2	0049	Off	On	1	0	0	1	0	0	6	6	0
BB0DMGR	CB	BB0DMGR	0045	Off	On	1	0	0	1	0	0	6	6	0

You can use the SMF 120.9 browser tool to build a sample classification XML for your existing environment from the provided SMF 120.9 records, as shown in Example 16-10. For more information, refer to the link in 16.14, “Tools” on page 644.

Example 16-10 Invoke command for SMF Browser classify function

```
java com.ibm.ws390.sm.smfview.SMF "INFILE(YOUR.SMF.DATA)"  
"PLUGIN(com.ibm.ws390.sm.smfview.ClassificationXMLFilter,/path/to/put/classify.xml  
)"
```

The following modify command is available to verify classification and processing cost. Example 16-11 shows the output of the command.

F *servername*,DISPLAY,WORK,CLINFO

Example 16-11 Output of the DISPLAY,WORK,CLINFO

```
BB000281I CLASSIFICATION COUNTERS FOR HTTP WORK  
BB000282I CHECKED 147, MATCHED 147, USED 0, COST 2, DESC: HTTP root  
BB000282I CHECKED 147, MATCHED 147, USED 5, COST 4, DESC: ITSO host  
BB000282I CHECKED 147, MATCHED 28, USED 28, COST 3, DESC: XML  
BB000282I CHECKED 119, MATCHED 114, USED 114, COST 4, DESC: PLANTS  
BB000283I FOR HTTP WORK: TOTAL CLASSIFIED 147, WEIGHTED TOTAL COST 560  
BB000188I END OF OUTPUT FOR COMMAND DISPLAY,WORK,CLINFO
```

The following modify command is available to reclassify work dynamically. Example 16-12 shows the output of the command.

F *servername*,RECLASSIFY,FILE='/path/to/new_classify.xml'

Example 16-12 Output of the RECLASSIFY,FILE

```
BBOJ0129I: The /webs/BBOACR1/wlm2.xml workload classification file was  
loaded at 2011/07/10 23:41:19.655 (EDT)  
BB000211I MODIFY COMMAND RECLASSIFY,FILE='/webs/BBOACR1/wlm2.xml'  
COMPLETED SUCCESSFULLY
```

Classification file note: If the new workload classification file cannot be loaded, then the application server discards the reloaded classification settings. The application server continues to run with the classification settings in effect before the modify command was issued.

Use the IBM z/OS Resource Measurement Facility™ (RMF™) Postprocessor workload activity reportor WebSphere Application server Runtime Performance Advisors to periodically review WebSphere Application Server performance indicators.

For more information about the RMF workload activity report, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=/com.ibm.websphere.zseries.doc/info/zseries/ae/tprf_capwar.html

16.11 Fast response cache accelerator and caching

This section discusses fast response cache accelerator (FRCA) support and caching enhancements in WebSphere Application Server V8.

16.11.1 FRCA overview

FRCA is a facility of the z/OS Communication Server TCP/IP component that IBM HTTP Server for z/OS has used for years. It provides a high speed caching mechanism where cached responses are served with high performance, using a minimum amount of CPU cycles. It provides caching capabilities for both static and dynamic content, such as pictures, HTML files, servlets, and JavaServer Pages (JSP) files.

Support note: Currently, FRCA cache is supported only for non-SSL connections.

For static content requests served from FRCA cache, the performance is comparable to the same scheme on a web server. Caching static content with FRCA is 14 times faster for 1 KB files and 4 - 11 times faster for 5 KB to 100 KB files than when using DynaCache. When used in dynamic caching, FRCA is using 2.7 times less CPU cycles than DynaCache. For the dynamic content requests, FRCA is taken as an extension to the DynaCache capability of each application server. There FRCA is defined as an “External Cache group” that is accessed by an adapter bean and that is owned by WebSphere Application Controller. Since z/OS 1.11, FRCA also allows web traffic to be carried on an IPv6 network.

Attention: FRCA functionality needs z/OS V1.9 or higher to be used. For FRCA to work properly, the fix for APAR PK72551 (UK42691) must be applied to the Communications Server TCP/IP on z/OS Version 1.9. If this fix is not applied, the server will issue the BB000347E or BB000348E error message. TCP/IP uses CSM storage to maintain this cache. So, verify and plan for your CSM usage with your system programmer.

16.11.2 Enabling FRCA in WebSphere Application Server

The FRCA function is enabled transparently by a signal from WebSphere Application Server for z/OS. As such, TCP/IP configuration statements are not required to enable this support.

To enable FRCA support within WebSphere Application Server, complete the following steps:

1. Click **Servers** → **Server Types** → **WebSphere application servers** → **server_name** → **Container Services**.

- Click **Dynamic cache service**, as shown in Figure 16-20.

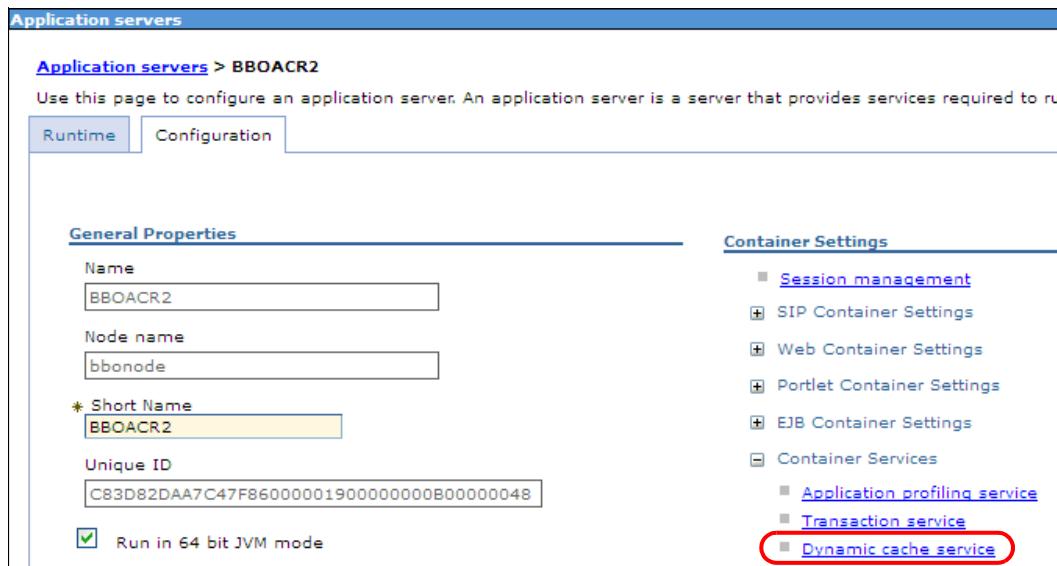


Figure 16-20 Application servers configuration window

- Verify that the **Enable service at server startup** General Properties option is selected. Also, select **External cache groups** from the additional properties, as shown in Figure 16-21.

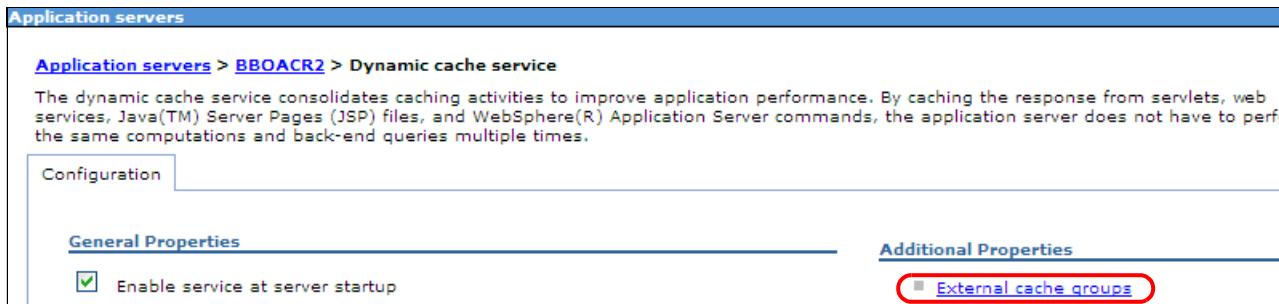


Figure 16-21 Dynamic cache service window

- Click **New** to create a new external cache group, as shown in Figure 16-22.



Figure 16-22 External cache group view

5. Specify a name for the external cache group in the Name field, as shown in Figure 16-23, and then click **Apply**.

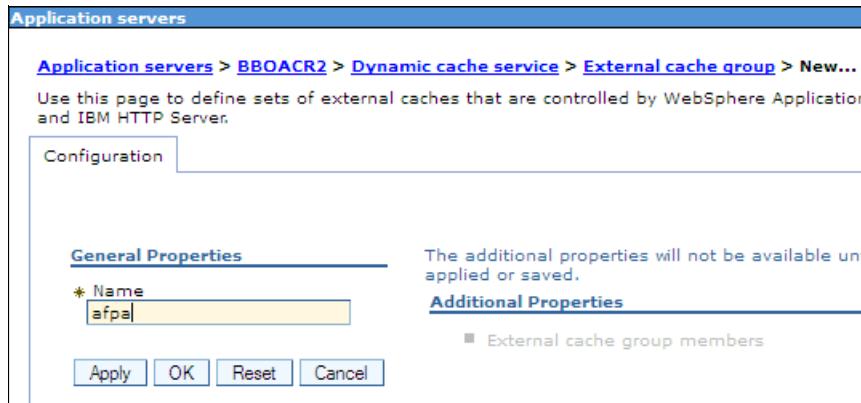


Figure 16-23 New External cache group window

6. Select your newly created group, as shown in Figure 16-24.

Application servers		
Application servers > BBOACR2 > Dynamic cache service > External cache group		
Use this page to define sets of external caches that are controlled by WebSphere Application and IBM HTTP Server.		
[+] Preferences		
<input type="button" value="New..."/>	<input type="button" value="Delete"/>	
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Select	Name	Type
You can administer the following resources:		
<input type="checkbox"/>	EsiInvalidator	Not shared
<input type="checkbox"/>	afpa	Not shared
Total	2	

Figure 16-24 External cache group view

7. Select **External cache group members**, as shown in Figure 16-25.

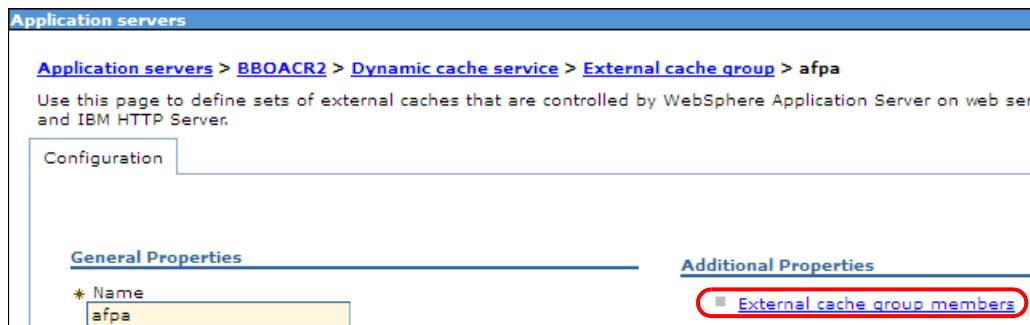


Figure 16-25 External cache group window

8. Click **New** to create the external cache group member, as shown in Figure 16-26.

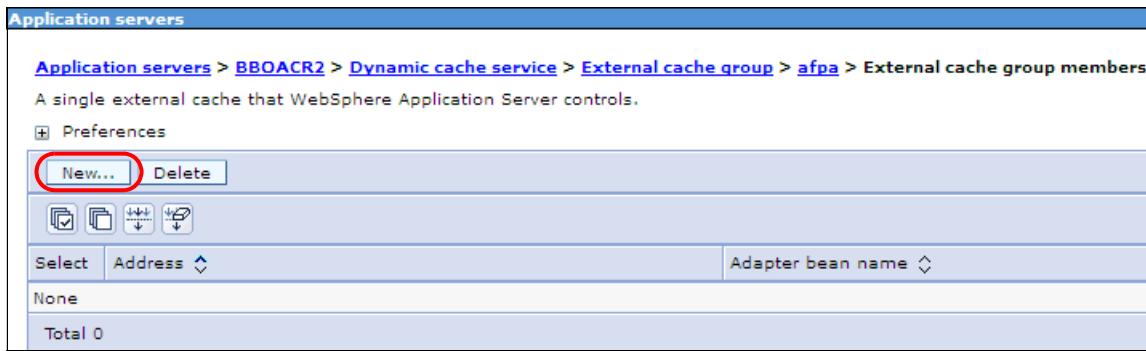


Figure 16-26 New External cache group member window

9. Select **Advanced Fast Path Architecture** with an adapter bean name of `com.ibm.ws.cache.servlet.Afpa` and indicate a port for AFPA to listen on internally (0 for net-new usage). Select **Enable Fast response cache accelerator**, as shown in Figure 16-27. Apply your configuration by clicking **Apply**.

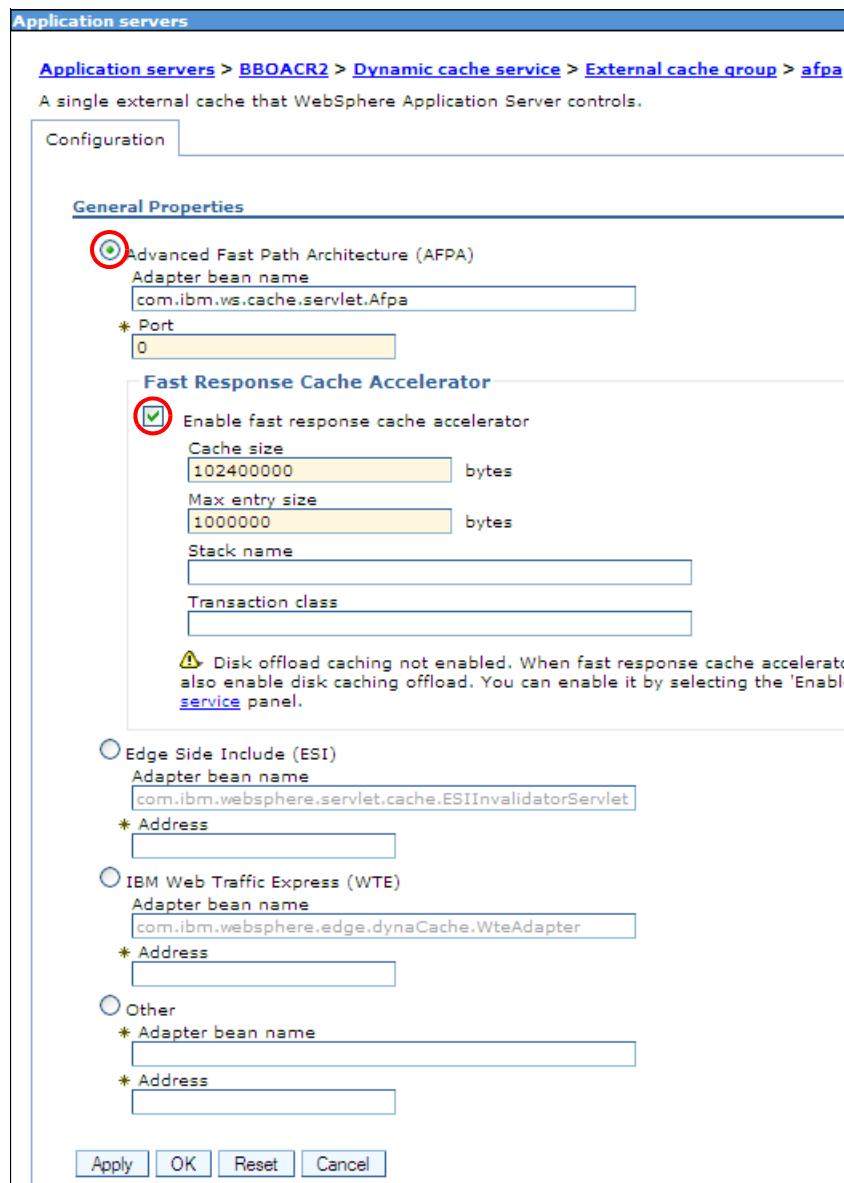


Figure 16-27 General properties page of the external cache group member

10. For a Network Deployment configuration, click **Review**, then click **Synchronize changes with nodes**, and click **Save**. Alternatively, click **System administration → Nodes**, and click **Synchronize** with appropriate node or nodes selected or select the **Synchronize changes with Nodes** option in **System administration → Console Preferences**.
11. We suggest that you also enable disk caching offload by clicking **Servers → Server Types → WebSphere application servers → server_name → Container Services → Dyna cache service**.
12. Restart your application server by using the **Restart** button for the target server in the **Servers → Server types → WebSphere Application servers** view.

For more information, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.zseries.doc/info/zseries/ae/tdyn_httpserverz.html

Disabling FRCA caching: By default, the FRCA cache is active on all channel chains that contain a web container channel and that do not contain an SSL channel. You can disable FRCA for specific channel chains and listener ports using the configuration tab for transport channels. Click **Servers** → **Server Types** → **WebSphere application servers** → **server_name** → **Web Container** → **Settings** → **Web container transport chains** → **transport_chain**, and select the **Disable FRCA caching** option.

To turn on memory-to-memory replication for dynamic cache service, refer to 17.3.5, “Cache replication” on page 694. Also consider turning on portlet and servlet caching.

To use FRCA caching, add a cache policy in the `cachespec.xml` file for your FRCA targeted object together with property name that specifies the FRCA external cache group name. Include the cache specification file with the deployment module.

Verification: Review the servant’s log file to verify that the `cachespec.xml` file was loaded successfully. A successful load is indicated by a DYN0047I message. In case of syntax or other errors, the cache file is not used.

In our scenario, we used the snoop servlet from the `DefaultApplication.ear` file, which is located in the `/WAS_product_image_path/installableApps` directory. You can invoke this file at `http://host:port/snoop` to demonstrate the FRCA enablement.

We updated a portion of the default `cachespec.sample.xml` file, which is located in the `/WAS_product_image_path/properties/` directory, to cache the snoop servlet into the FRCA cache, as shown in Example 16-13.

Example 16-13 FRCA enablement inside the portion of `cachespec.sample.xml`

```
<?xml version="1.0" ?>
<!DOCTYPE cache SYSTEM "cachespec.dtd">
<cache>
    <cache-entry>
        <class>servlet</class>
        <name>/snoop</name>
        <cache-id>
            <component id="*" type="parameter">
                <required>false</required>
            </component>
            <component id="" type="pathinfo">
                <required>false</required>
            </component>
            <component id="host" type="header">
                <required>false</required>
            </component>
            <timeout>180</timeout>
        </cache-id>
    <property name="ExternalCache">afpa</property>
    </cache-entry>
</cache>
```

For more information and the full syntax for the cache specification XML file, refer to 16.11.3, “Cache specification XML” on page 637.

You can use the following commands to verify that FRCA is working inside WebSphere Application Server:

- ▶ **F servername,DISPLAY,FRCA**
- ▶ **F servername,DISPLAY,FRCA,CONTENT**
- ▶ **F servername,DISPLAY,FRCA,STATS**

Example 16-14, Example 16-15, and Example 16-16 show the output from these commands.

Example 16-14 Output of the DISPLAY,FRCA command

```
BB000351I TIME OF LAST FRCA DISPLAY 2011/07/13 17:28:18.768656
BB000352I FRCA CURRENT CACHED OBJECTS 1
BB000353I FRCA TOTAL CACHED OBJECTS 1 (DELTA 0)
BB000354I FRCA CURRENT CACHED SIZE 14K
BB000355I FRCA OBJECTS PUSHED 0 (DELTA 0)
BB000188I END OF OUTPUT FOR COMMAND DISPLAY,FRCA
```

Example 16-15 Output of the DISPLAY,FRCA,CONTENT command

```
BB000352I FRCA CURRENT CACHED OBJECTS 1
BB000358I LIST OF CACHED OBJECTS WRITTEN TO JOBLOG
BB000357I SIZE: 13351 KEY: wtsc80.itso.ibm.com:32674/snoop
BB000188I END OF OUTPUT FOR COMMAND DISPLAY,FRCA,CONTENT
```

Example 16-16 Output of the DISPLAY,FRCA,STATS command

```
BB000351I TIME OF LAST FRCA DISPLAY 2011/07/13 17:25:41.210033
BB000352I FRCA CURRENT CACHED OBJECTS 1
BB000353I FRCA TOTAL CACHED OBJECTS 1 (DELTA 0)
BB000354I FRCA CURRENT CACHED SIZE 14K
BB000355I FRCA OBJECTS PUSHED 0 (DELTA 0)
BB000356I FRCA OBJECTS OK - 16K 1
BB000356I FRCA OBJECTS 16K - 32K 0
BB000356I FRCA OBJECTS 32K - 64K 0
BB000356I FRCA OBJECTS 64K - 256K 0
BB000356I FRCA OBJECTS 256K - 1M 0
BB000356I FRCA OBJECTS >1M 0
BB000188I END OF OUTPUT FOR COMMAND DISPLAY,FRCA,STATS
```

Use the following command to verify TCP/IP FRCA cache, as shown in Example 16-17:

```
D TCPIP,,NET,CACHEINFO
```

You can also use the equivalent **netstat -C** command.

Example 16-17 Output of the D TCPIP,,N,CACH command

```
D TCPIP,TCPIP,N,CACH
EZZ2500I NETSTAT CS V1R12 TCPIP 221
CLIENT: BBOACR1           LISTENING SOCKET: 0.0.0.0..32674
  CACHETYPE:      SHARED      ASID:          00BB
  MAXCACHESIZE:   0000025000  CURRCACHESIZE:  0000000004
  MAXNUMOBJECTS:  0999999999  CURRNUMOBJECTS: 0000000001
  NUMCONNS:       0000000008  CONNSPROCESSED: 0000000001
```

```

CONNSDEFERRED: 0000000007 CONNSTIMEDOUT: 0000000000
REQUESTSPROCESSED: 0000000001 INCOMPLETEREQUESTS: 0000000000
NUMCACHEHITS: 0000000001 NUMCACHEMISSES: 0000000067
NUMUNPRODCACHEHITS: 0000000000

CLIENT: BBOACR1 LISTENING SOCKET: 0.0.0.0..32672
CACHETYPE: SHARED ASID: 00BB
MAXCACHESIZE: 0000025000 CURRCACHESIZE: 0000000004
MAXNUMOBJECTS: 0999999999 CURRNUMOBJECTS: 0000000001
NUMCONNS: 0000000000 CONNSPROCESSED: 0000000000
CONNSDEFERRED: 0000000000 CONNSTIMEDOUT: 0000000000
REQUESTSPROCESSED: 0000000000 INCOMPLETEREQUESTS: 0000000000
NUMCACHEHITS: 0000000000 NUMCACHEMISSES: 0000000000
NUMUNPRODCACHEHITS: 0000000000

2 OF 2 RECORDS DISPLAYED
END OF THE REPORT

```

Attention: If you need to cache objects larger than 10 MB, set the protocol_http_large_data_response_buffer custom property by clicking **Environment** → **WebSphere variables** for the desired server. The value for this property should be higher than the maximum size of the object that should be cached.

For information about how to optionally turn on FRCA logging, go to the following website:
http://publib.boulder.ibm.com/infocenter/wasinfo/fep/index.jsp?topic=/com.ibm.websphere.nd.multiplatform.doc/info/ae/ae/utrb_httplogs.html

For information about monitoring the cache contents and overall settings, refer to 16.11.6, “Using IBM Extended Dynamic Cache Monitor to supervise caching” on page 638.

16.11.3 Cache specification XML

With caching enabled, rules are needed to define cacheable objects and the policies that guide them. You can define rules using the elements inside the cache specification XML file, as opposed to using an API-based type of caching. You can define multiple caching instances to cache the same servlet or object inside the single server.

The cache specification file supports the following content types:

- ▶ Static
- ▶ Servlet
- ▶ Portlet
- ▶ Webservice
- ▶ JAXRPCClient, for a web service client
- ▶ Command, for a WebSphere Application Server command programming model

You can save a global cachespec.xml file in the application server properties directory, but the best practice is to place the cache configuration file with the deployment module inside the application’s web module WEB-INF or enterprise bean META-INF directory.

When the server starts, the cache parses the cachespec.xml file and extracts a set of configuration parameters from each cache-entry element. Every time a new servlet or other cacheable object initializes, the cache attempts to match each of the cache-entry elements to find the configuration information for that object. Grouping, sharing, and invalidation mechanisms are available for distributing the cache and keeping it current. Disk off load is available for overflow and persistence.

For more information and the full syntax of the cache specification XML file, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=/com.ibm.websphere.base.doc/info/aes/ae/rdyn_cachespec.html

16.11.4 FRCA and RACF integration

FRCA services can be restricted. If access is restricted (the SERVAUTH class is active and the FRCA resource are defined) in your environment, then WebSphere Application Server must be granted access.

If the access is restricted, then the message shown in Example 16-18 is issued.

Example 16-18 FRCA access denied message

```
BB00nnnE FRCA INITIALIZATION FAILED. SERVER NOT AUTHORIZED TO USE FRCA SERVICES.  
IOCTL RV=%d, RC=%d, RSN=%08X
```

To use FRCA, the following RACF definitions are required:

1. Define a new SERVAUTH profile with the corresponding system name and TCP/IP procedure name:

```
RDEFINE SERVAUTH EZB.FRCAACCESS.<system_name>.<TCPIP_procname> UACC(NONE)
```

2. Give the application server control region READ access to this SERVAUTH profile:

```
PERMIT EZB.FRCAACCESS.<system_name>.<TCPIP_procname> CLASS (SERVAUTH) ID  
(UID_CR) ACCESS (READ)
```

3. Refresh the SERVAUTH class to activate the changes:

```
SETRPOTS RACLST (SERVAUTH) REFRESH
```

16.11.5 Caching enhancements in WebSphere Application Server V8

WebSphere Application Server V8 introduces integration of dynamic cache with the Java Persistence API (JPA) second level (L2). Dynamic cache service plugs in as a cache provider for JPA L2 cache and shares entity states across various persistence contexts, transactions, and users for both DataCache and QueryCache. As such, it can use all the monitoring, synchronization, and replication capabilities provided with the dynamic cache. For information about enabling JPA L2 cache for your applications, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=%2Fcom.ibm.websphere.nd.doc%2Finfo%2Fae%2Frdyn_openjpa.html

Dynamic cache provides servlet caching support for the Servlet 3.0 specification. For more information, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=%2Fcom.ibm.websphere.express.doc%2Finfo%2Fexp%2Fae%2Fcdyn_servlet3caching.html

16.11.6 Using IBM Extended Dynamic Cache Monitor to supervise caching

IBM Extended Dynamic Cache Monitor is a tool delivered by IBM for dynamic object cache monitoring. It provides a real-time view of the dynamic cache state and is the only way to manipulate the cache resident data. It allows for simple cache statistics, cache entries, and cache policy information for servlet cache instances.

Extended Dynamic Cache Monitor brings the following enhancements to the monitoring capabilities:

- ▶ Display the contents of object cache instances.
- ▶ Display the Dynamic Cache Mbean statistics for cache instances across all members of a cluster.
- ▶ Look at the push-pull table associated with a cache instance.
- ▶ Search memory contents, disk contents, and the push-pull table for cache IDs using a regular expression.
- ▶ Compare cache instances.

For more information about installing the dynamic cache monitor, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/topic/com.ibm.websphere.express.doc/info/exp/ae/tdyn_servletmonitor.html

For more information about installing Extended Dynamic Cache Monitor, go to the following website:

http://www.ibm.com/developerworks/websphere/downloads/cache_monitor.html

For more information about tuning dynamic cache, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/index.jsp?topic=/com.ibm.websphere.express.doc/info/exp/ae/tdyn_cache_tune.html

16.12 Using WebSphere for z/OS Optimized Local Adapters

This section describes how to optimize local adapters. This feature is a high performance connectivity feature that is specific to a z/OS system.

16.12.1 Introduction to Optimized Local Adapters

WebSphere for z/OS Optimized Local Adapters (WOLA) is a function introduced in WebSphere Application Server V7.0.0.4 with Information Management Support (IMS) support included in V7.0.0.12. It evolved from a local communications mechanism of the daemon server that was present for local IIOP calls in WebSphere Application Server for z/OS. Local communications routines are now externalized and programmatic APIs are available, together with a standard JCA adapter.

WOLA is an inbound and outbound method of local cross-memory communication between WebSphere Application Server for z/OS and external address spaces, as shown in Figure 16-28.

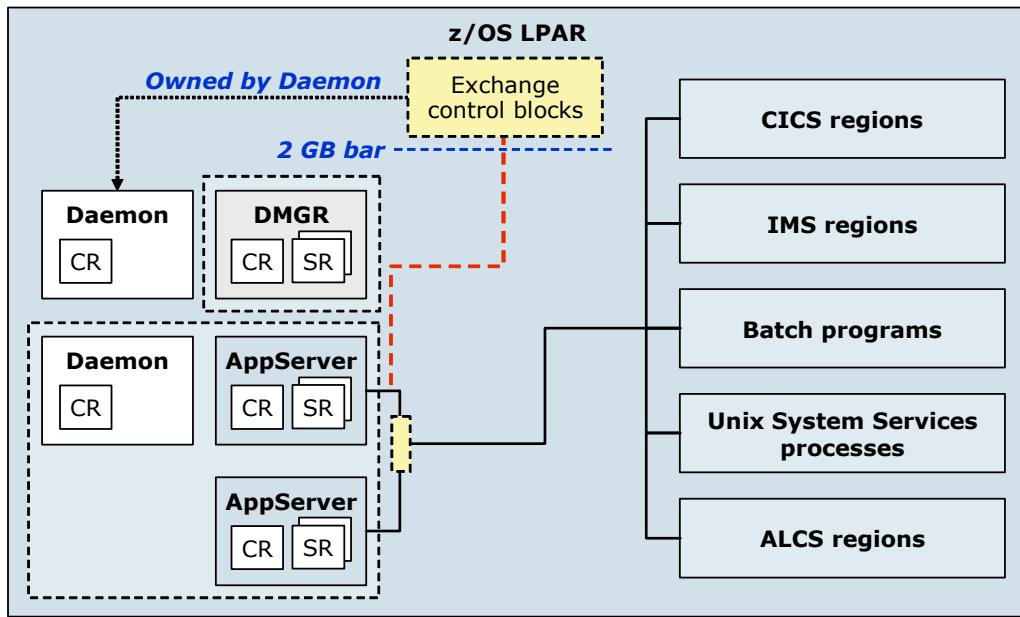


Figure 16-28 WOLA overview

As shown in the figure, WOLA communicates with the following external address spaces:

- ▶ CICS regions
- ▶ IMS regions
- ▶ Batch programs
- ▶ UNIX System Services processes
- ▶ Airlines line control (ALCS) programs

The following supported languages can be used to invoke the callable services:

- ▶ Cobol
- ▶ C/C++
- ▶ PL/I
- ▶ High level assembler

WOLA represents a way to link WebSphere Application Server for z/OS and external address spaces in a secure, optimized, high-speed, cross-memory, and bidirectional manner. WOLA is high throughput solution, with a low impact per transaction or exchange.

However, because WOLA is a low-level, cross-memory exchange mechanism between the daemon and specific WOLA-enabled control region, it is limited to a single z/OS operating system only. The daemon owned shared space exchange control blocks reside above the 2 GB bar. External address space always registers with the local daemon through the **BB0A1REG** call and provides target environment values together with registration name to initiate communication.

The WOLA function is provided as a complement to currently offered solutions as a fast and efficient way to invoke business logic and services inside WebSphere Application Server for z/OS.

For more information about the WOLA feature, go to the following website:
<http://www-03.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP101490>

16.12.2 Enabling WebSphere for z/OS Optimized Local Adapters

To enable WOLA support in WebSphere Application Server V8, complete the following steps:

1. Use the **copyZOS.sh** script to off load WOLA modules from UNIX System Services, as shown in Example 16-19. It creates automatically a 35-track load library with the name specified on the script arguments.

Example 16-19 Using copyZOS to create symbolic links and to copy WOLA modules

```
/WAS_product_image_path/bin/copyZOS.sh OLAMODS YOUR.DESIRED.LOADLIB
```

2. Use the **olaRar.py** script to install the WOLA resource adapter and J2C connection factory, as shown in Example 16-20. The cell scoped `WAS_DAEMON_ONLY_enable_adapter` variable is set during the process, and the configuration is validated and saved.

Example 16-20 Install WOLA adapter using the olaRAR.py script

```
 ${WAS_INSTALL_ROOT}/bin/wsadmin.sh -lang jython -f  
 /WAS_product_image_path/util/zos/OLASamples/olaRar.py bbocell bbonode
```

3. Click **Environment** → **WebSphere variables**. Select **All scopes**, and verify the existence of the `WAS_DAEMON_ONLY_enable_adapter` variable that covers your desired scope, as shown in Figure 16-29.

The screenshot shows the 'WebSphere Variables' interface. At the top, there is a note about defining substitution variables. Below it, a section titled 'Scope: All scopes' has a checked checkbox for 'Show scope selection drop-down list with the all scopes option'. A dropdown menu labeled 'All scopes' is shown. Below this, there is a 'Preferences' section with 'New...' and 'Delete' buttons, and a toolbar with icons for creating, deleting, and filtering resources. A table lists variables, with a filter applied for 'Name' containing '*enable_adapter*'. One row is visible in the table, showing the variable 'WAS_DAEMON_ONLY_enable_adapter' with value 'true' and scope 'Cell=bbocell'. At the bottom, there is a summary: 'Total 237 Filtered total: 1'.

Figure 16-29 New WAS_DAEMON_ONLY_enable_adapter variable in cell scope

4. Verify the existence of the new resource adapter on the desired scope, as shown in Figure 16-30.

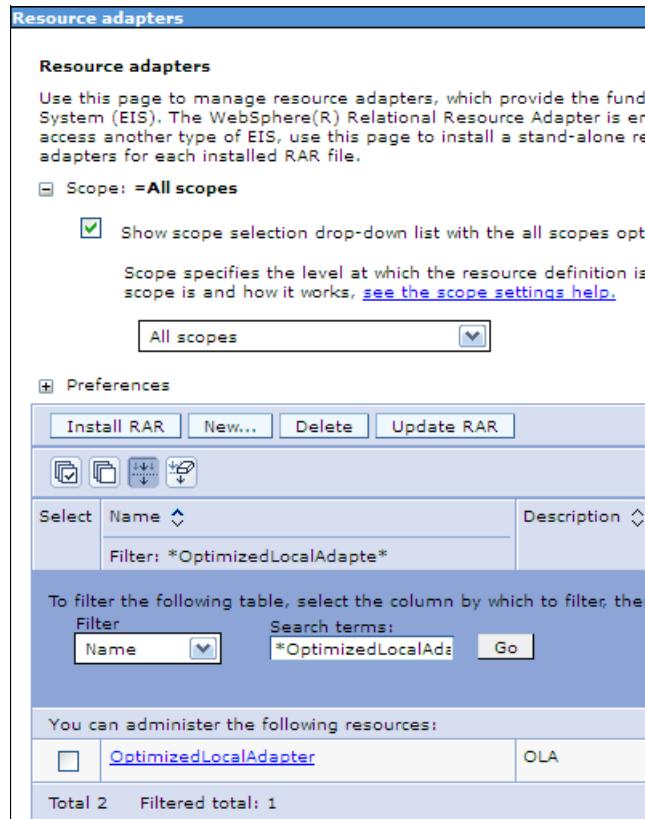


Figure 16-30 New OLA resource adapter

- Verify the existence of the new J2C connection factory on the desired scope, as shown in Figure 16-31.

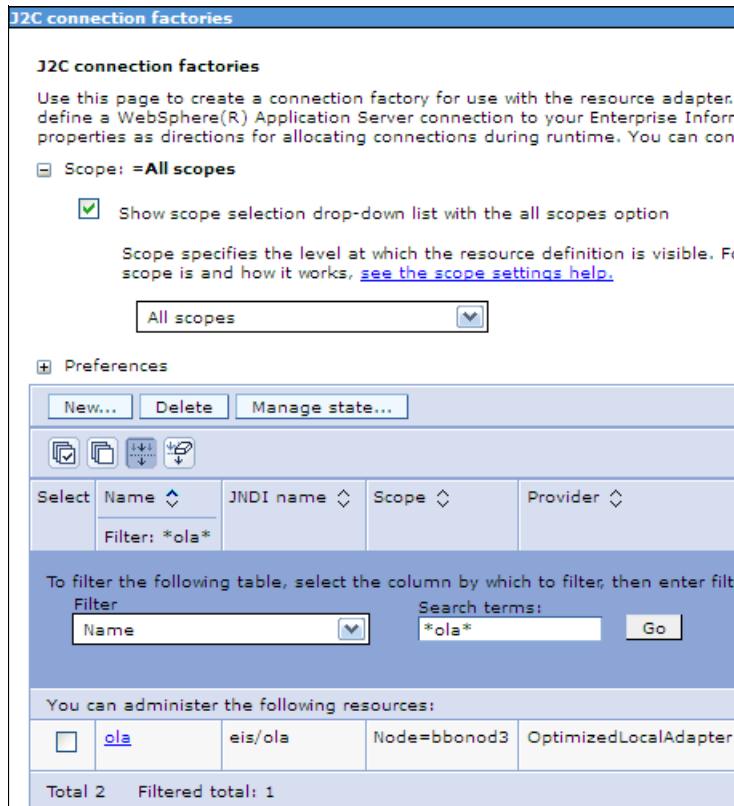


Figure 16-31 New OLA J2C connection factory

- Restart the WebSphere Application Server daemon. (This action also stops all the connected cell servers on the same z/OS LPAR.)
- Verify the presence of the BB0M0001I message enable_adapter:1 during the daemon start.

For more information about enabling WOLA support in WebSphere Application Server, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=%2Fcom.ibm.websphere.nd.multiplatform.doc%2Finfo%2Fae%2Fae%2Ftdat_enableconnector.html

For WOLA samples, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=/com.ibm.websphere.zseries.doc/info/zseries/ae/cdat_olasamples.html

For information about enabling WOLA support in CICS, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.nd.multiplatform.doc/info/ae/ae/tdat_enableconnectorcics.html

For information about enabling WOLA support in IMS, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.zseries.doc/info/zseries/ae/tdat_enableconnectorims.html

For more info on using WOLA with ALCS, go to the following website:

<http://www-01.ibm.com/software/tpf/alcs/pubs/wassampl.pdf>

For an overview of WOLA APIs and error and reason codes, go to the following website:
http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=/com.ibm.websphere.nd.multiplatform.doc/info/ae/ae/cdat_olaapis.html

In WebSphere Application Server for z.OS V8, WOLA also participates in the JCA failover scenarios. It can fail over to another external address space (for example, CICS region) on the same LPAR.

For more information, refer to 17.3, “Failover and fallback” on page 676.

16.13 IBM HTTP Server Status monitoring page

Refer to 13.5.5, “IBM HTTP server status monitoring page” on page 516 for more information about the IBM HTTP Server status monitoring page.

HTTP Server plug-in properties might affect workload distribution and performance.

For more information, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=/com.ibm.websphere.zseries.doc/info/zseries/ae/uwsv_plugin_props.html

16.14 Tools

There is wide variety of WebSphere Application Server V8 tools available that can help with application server performance monitoring or application profiling. You can use these tools during problem determination and as a means of measuring the performance of your application and the components that it encompasses. When using these tools in production environments, consider the possible performance costs that are associated with collecting data.

Refer to 18.4.6, “z/OS monitoring” on page 729 for more information about performance monitoring tools.

Also, consult the following resources:

- ▶ To understand how to monitor Performance Monitoring Infrastructure (PMI) data with Tivoli Performance Viewer, go to the following website:
http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.nd.doc/info/ae/ae/tprf_tpmonitor.html
- ▶ For RMF usage, refer to the *z/OS V1R12.0 Resource Measurement Facility User's Guide*, SC33-7990
- ▶ For information about IBM Tivoli Composite Application Manager for Application Diagnostics V7.1, go to the following website:
<http://www-01.ibm.com/software/tivoli/products/composite-application-mgr-diagnostics/>
- ▶ To download IBM Support Assistant, go to the following website:
<http://www-01.ibm.com/software/support/isa/>

You can find a complete list of IBM Support Assistant add-ons at the following website:

http://www-01.ibm.com/support/docview.wss?uid=swg27013116&cmp=101AM&ct=101AMES3&cr=01_Net&cm=S&csr=Websphere&co=On&cot=A&cd=2011-04-10&cpg=CIOP&cn=Impact2011_ISA

- ▶ To download WLMQUE with Rexx Alternate Library, go to the following website:
<http://www-03.ibm.com/systems/z/os/zos/features/wlm/tools/wlmque.html>
- ▶ To download SMF120.9 browser with instructions, go to the following website:
<https://www14.software.ibm.com/webapp/iwm/web/preLogin.do?source=zosos390>
- ▶ To download the Extended Dynamic Cache Monitor, go to the following website:
<http://www.ibm.com/developerworks/apps/download/index.jsp?contentid=210854&filename=cachemonitor.zip&method=http&locale=>
- ▶ Visit the IBM tools and toys for z/OS page at the following website:
<http://www-03.ibm.com/systems/z/os/zos/features/unix/tools/>



Clustering and high availability

High availability is a system design approach that tries to eliminate single points of failure and ensure system availability by employing redundancy.

The high availability framework that is delivered with WebSphere Application Server for z/OS is complemented by a degree of availability that is provided by hardware components, such as IBM z/Architecture® and the z/OS operating system. Parallel Sysplex as a zSeries clustering technology in this topology provides efficient redundancy with a view of the systems as a single logical computing environment. A coupling facility in a sysplex is used for data sharing and reliable messaging between the members of the complex. Several logical partitions or independent machines can be used to prevent unplanned software or hardware outages. The operating system takes advantage of the self-healing attributes of the hardware and extends them by adding functions such as recovery services for all operating system code, address space isolation, and storage key protection. Functions such as Workload Manager (WLM), Resource Recovery Services (RRS), and automatic restart manager (ARM) assure the availability of applications. A sysplex distributor with defined dynamic virtual IP address (DVIPA) handles IP address availability with the possibility of a host role assignment and takes advantage of workload balancing.

This chapter focuses on the z/OS system and introduces high availability concepts and practices for WebSphere Application Server for z/OS features.

It includes the following topics:

- ▶ Clustering on z/OS systems
- ▶ High availability
- ▶ Failover and failback

17.1 Clustering on z/OS systems

Clustering technology is an integral part of WebSphere Application Server for z/OS V8. In this section, we provide information about the concept of clustering and how to create a cluster on z/OS systems.

17.1.1 Clustering for scalability and failover

Clustering as a high availability approach is used extensively in WebSphere Application Server to provide for scalability and failover protection at the same time. A *cluster* consists of multiple copies of the same component with the expectation that at least one of the copies will be available to service a request. In general, the cluster works as a unit where there is collaboration among the individual copies to ensure that the request can be directed toward a copy that is capable of servicing the request.

A WebSphere Application Server cluster is composed of individual cluster members, with each member containing the same set of applications. In front of a WebSphere Application Server cluster is a *workload distributor*, which routes the workload to individual members. z/OS Workload Manager (WLM) assigns system resources to units of work from the workload, making a best attempt to allow all work to meet the specified goals.

Clusters can be vertical within an LPAR (that is, two or more cluster members residing in the same z/OS system), or they can be placed horizontally across LPARs on different machines to obtain the highest availability in the event that an LPAR or machine that contains a member has an outage. In a vertical cluster, the servers compete with each other for resources.

For more information about clustering in WebSphere Application Server for z/OS V8, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.zseries.doc/info/zseries/ae/crun_srvgrp.html

17.1.2 Creating a cluster on a z/OS system

To create a WebSphere Application Server for z/OS cluster, complete the following steps:

1. Click **Servers** → **Clusters** → **WebSphere application server clusters**.
2. Click **New** in the WebSphere Application Server clusters view, as shown in Figure 17-1.

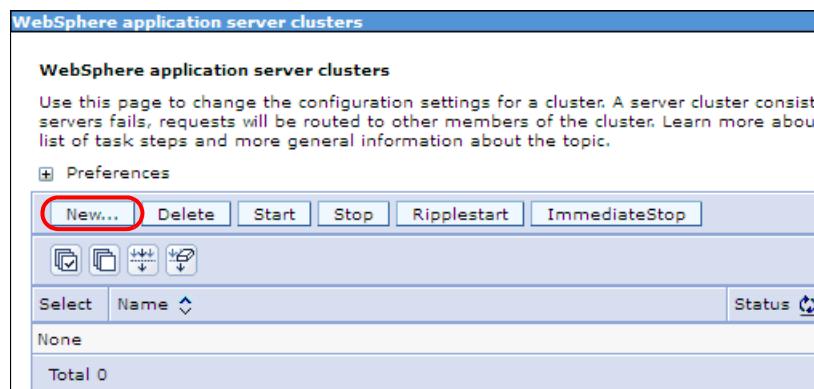


Figure 17-1 WebSphere application server clusters view

3. Enter basic cluster information, as shown in Figure 17-2. If you are converting an existing application server into the cluster, leave the “Short name” field clear to default to the generic short name of that application server. Click **Next**.

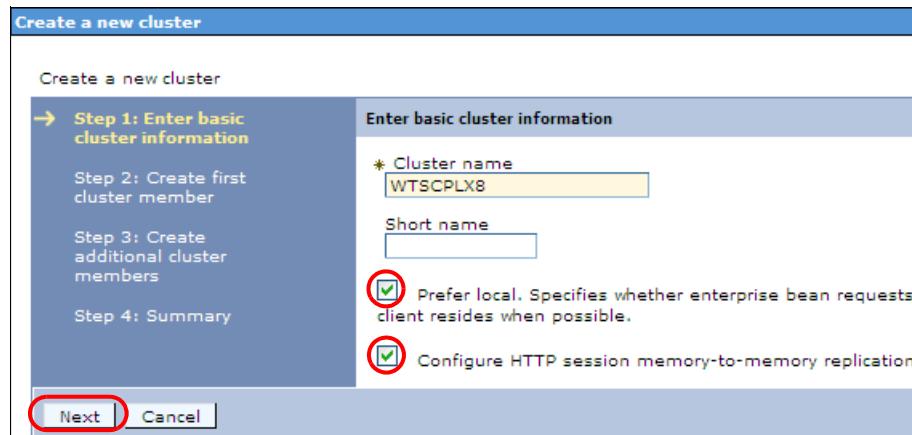


Figure 17-2 Enter basic cluster information

4. Create the first cluster member. In the “Select basis for first cluster member” section, select **Create the member by converting an existing application server**, as shown in Figure 17-3. Click **Next**.

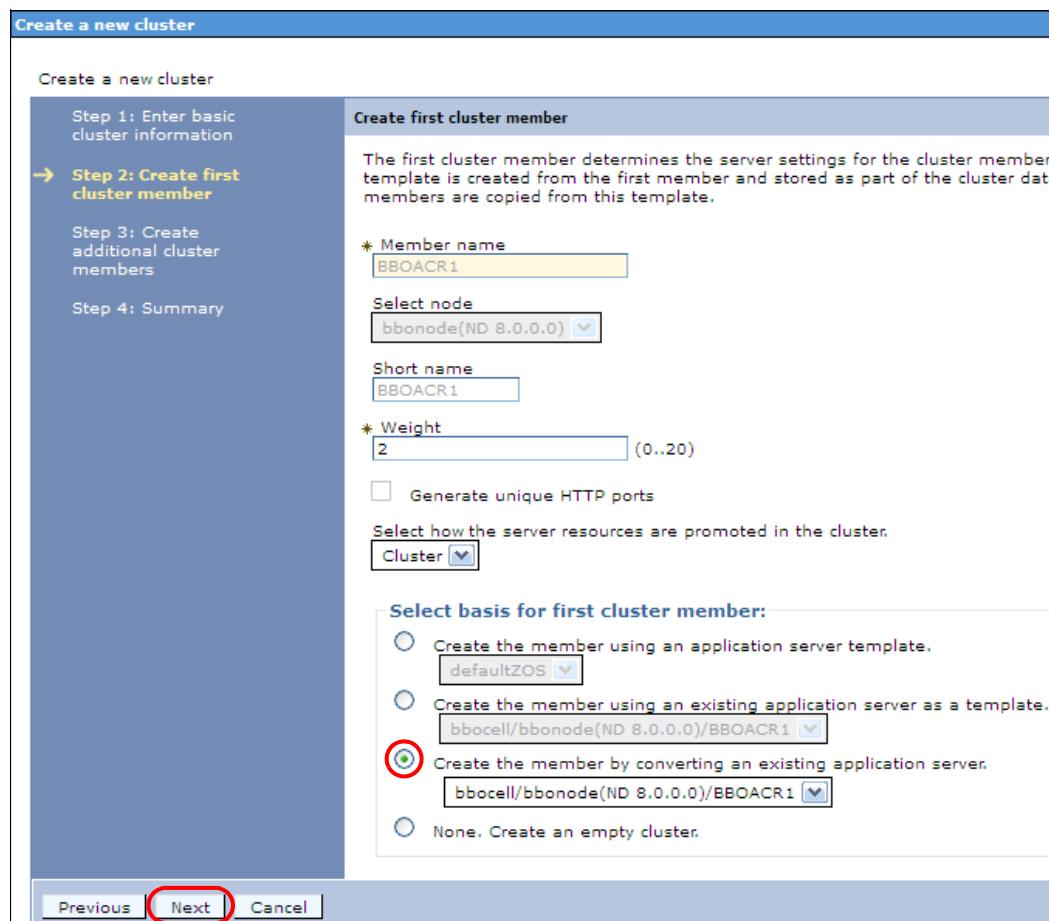


Figure 17-3 Create first cluster member

5. Add additional cluster members by specifying the member name and short name. Click **Add Member**, as shown in Figure 17-4. Click **Next**.

Create a new cluster

Create a new cluster

Step 1: Enter basic cluster information

Step 2: Create first cluster member

→ Step 3: Create additional cluster members

Step 4: Summary

Create additional cluster members

Enter information about this new cluster member, and click Add Member to add this member to the member list. A server configuration template is created from the first member, and of the cluster data. Additional cluster members are copied from this template.

* Member name
BBOACR4

Select node
bbonod2(ND 8.0.0.0) ▾

Short name
BBOACR4

* Weight
2 (0..20)

Generate unique HTTP ports

Add Member

Use the Edit function to modify the properties of a cluster member in this list. Use the Delete function to remove a cluster member from this list. You are not allowed to edit or remove the member.

Edit Delete

Select	Member name	Nodes	Version
<input type="checkbox"/>	BBOACR1	bbonode	ND 8.0.0.0
<input type="checkbox"/>	BBOACR2	bbonod2	ND 8.0.0.0
<input type="checkbox"/>	BBOACR3	bbonode	ND 8.0.0.0
Total 3			

Figure 17-4 Create additional cluster members

6. Review your setup in the summary and click **Finish**, as shown Figure 17-5.

Create a new cluster

Create a new cluster																																																															
Step 1: Enter basic cluster information Step 2: Create first cluster member Step 3: Create additional cluster members → Step 4: Summary	Summary Summary of actions: <table border="1"> <thead> <tr> <th>Options</th> <th>Values</th> </tr> </thead> <tbody> <tr> <td>Cluster Name</td> <td>WTSCPLX8</td> </tr> <tr> <td>Core Group</td> <td>DefaultCoreGroup</td> </tr> <tr> <td>Node group</td> <td>DefaultNodeGroup</td> </tr> <tr> <td>Prefer local</td> <td>true</td> </tr> <tr> <td>Configure HTTP session memory-to-memory replication</td> <td>true</td> </tr> <tr> <td>Server name</td> <td>BBOACR1</td> </tr> <tr> <td>Node</td> <td>bbonode(ND 8.0.0.0)</td> </tr> <tr> <td>Weight</td> <td>2</td> </tr> <tr> <td>Clone Template</td> <td>bbocell/bbonode(ND 8.0.0.0)/BBOACR1</td> </tr> <tr> <td>Clone Basis</td> <td>Create the member by converting an existing application server</td> </tr> <tr> <td>Select how the server resources are promoted in the cluster.</td> <td>cluster</td> </tr> <tr> <td>Generate unique HTTP ports</td> <td>false</td> </tr> <tr> <td>Server name</td> <td>BBOACR2</td> </tr> <tr> <td>Node</td> <td>bbonod2(ND 8.0.0.0)</td> </tr> <tr> <td>Short name</td> <td>BBOACR2</td> </tr> <tr> <td>Weight</td> <td>2</td> </tr> <tr> <td>Clone Template</td> <td>Version 8 member template</td> </tr> <tr> <td>Generate unique HTTP ports</td> <td>true</td> </tr> <tr> <td>Server name</td> <td>BBOACR3</td> </tr> <tr> <td>Node</td> <td>bbonode(ND 8.0.0.0)</td> </tr> <tr> <td>Short name</td> <td>BBOACR3</td> </tr> <tr> <td>Weight</td> <td>2</td> </tr> <tr> <td>Clone Template</td> <td>Version 8 member template</td> </tr> <tr> <td>Generate unique HTTP ports</td> <td>true</td> </tr> <tr> <td>Server name</td> <td>BBOACR4</td> </tr> <tr> <td>Node</td> <td>bbonod2(ND 8.0.0.0)</td> </tr> <tr> <td>Short name</td> <td>BBOACR4</td> </tr> <tr> <td>Weight</td> <td>2</td> </tr> <tr> <td>Clone Template</td> <td>Version 8 member template</td> </tr> <tr> <td>Generate unique HTTP ports</td> <td>true</td> </tr> </tbody> </table>	Options	Values	Cluster Name	WTSCPLX8	Core Group	DefaultCoreGroup	Node group	DefaultNodeGroup	Prefer local	true	Configure HTTP session memory-to-memory replication	true	Server name	BBOACR1	Node	bbonode(ND 8.0.0.0)	Weight	2	Clone Template	bbocell/bbonode(ND 8.0.0.0)/BBOACR1	Clone Basis	Create the member by converting an existing application server	Select how the server resources are promoted in the cluster.	cluster	Generate unique HTTP ports	false	Server name	BBOACR2	Node	bbonod2(ND 8.0.0.0)	Short name	BBOACR2	Weight	2	Clone Template	Version 8 member template	Generate unique HTTP ports	true	Server name	BBOACR3	Node	bbonode(ND 8.0.0.0)	Short name	BBOACR3	Weight	2	Clone Template	Version 8 member template	Generate unique HTTP ports	true	Server name	BBOACR4	Node	bbonod2(ND 8.0.0.0)	Short name	BBOACR4	Weight	2	Clone Template	Version 8 member template	Generate unique HTTP ports	true
Options	Values																																																														
Cluster Name	WTSCPLX8																																																														
Core Group	DefaultCoreGroup																																																														
Node group	DefaultNodeGroup																																																														
Prefer local	true																																																														
Configure HTTP session memory-to-memory replication	true																																																														
Server name	BBOACR1																																																														
Node	bbonode(ND 8.0.0.0)																																																														
Weight	2																																																														
Clone Template	bbocell/bbonode(ND 8.0.0.0)/BBOACR1																																																														
Clone Basis	Create the member by converting an existing application server																																																														
Select how the server resources are promoted in the cluster.	cluster																																																														
Generate unique HTTP ports	false																																																														
Server name	BBOACR2																																																														
Node	bbonod2(ND 8.0.0.0)																																																														
Short name	BBOACR2																																																														
Weight	2																																																														
Clone Template	Version 8 member template																																																														
Generate unique HTTP ports	true																																																														
Server name	BBOACR3																																																														
Node	bbonode(ND 8.0.0.0)																																																														
Short name	BBOACR3																																																														
Weight	2																																																														
Clone Template	Version 8 member template																																																														
Generate unique HTTP ports	true																																																														
Server name	BBOACR4																																																														
Node	bbonod2(ND 8.0.0.0)																																																														
Short name	BBOACR4																																																														
Weight	2																																																														
Clone Template	Version 8 member template																																																														
Generate unique HTTP ports	true																																																														
<input type="button" value="Previous"/> <input type="button" value="Finish"/> <input type="button" value="Cancel"/>																																																															

Figure 17-5 Summary

7. Click **Review**, click **Synchronize changes with nodes**, and click **Save**. Alternatively, click **System administration** → **Nodes** and click **Synchronize** with the appropriate node or nodes selected or enable the **Synchronize changes with Nodes** option in **System administration** → **Console Preferences**.
8. Restart the converted application server by clicking **Servers** → **Server Types** → **WebSphere application servers**.

Replication note: When you select HTTP replication, a replication domain of the same name as the cluster name is created automatically with a Single replica option.

For more information about creating a cluster, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.zseries.doc/info/zseries/ae/trun_wlm_cluster_v61.html

17.2 High availability

The high availability framework that is provided with the product eliminates single points of failure and provides peer-to-peer failover for applications and processes that are running within the product environment.

In this section, we provide an overview of the high availability features in WebSphere Application Server for z/OS.

17.2.1 High availability manager

High availability manager provides services for product components so that they can make themselves highly available. By default, high availability manager instance runs on every application server, proxy server, node agent, and deployment manager in a cell.

A cell can be divided into multiple high availability domains known as *core groups*. Each high availability manager instance establishes network connectivity with all other high availability manager instances in the same core group, using a specialized, dedicated, and configurable transport channel. The transport channel provides mechanisms that allow the high availability manager instance to detect when other members of the core group start, stop, or fail. Automatic restart manager (ARM), Tivoli System Automation, or other automation software can be configured to restart failed WebSphere Application Server controllers.

Within a core group, high availability manager instances are elected to coordinate high availability activities. An instance that is elected is known as a *core group coordinator*. The coordinator is highly available, such that if a process that is serving as a coordinator stops or fails, another instance is elected to assume the coordinator role, without loss of continuity.

The high availability manager periodically runs a number of background tasks, such as checking the health of highly available singleton services that it is managing. Most of these background tasks consume trivial amounts of CPU.

The exceptions are the regularly scheduled Discovery and Failure Detection Protocols:

- ▶ The Discovery Protocol discovers when other core group processes start and opens network connections to these other members.
- ▶ The View Synchrony Protocol establishes reliable messaging with other core group members after the connections are opened.
- ▶ The Failure Detection Protocol detects when other core group members stop or become unreachable because of a network partition.

Distribution and Consistency Services

The Distribution and Consistency Services (DCS) transport chain provides the underlying group services framework for the high availability manager, such that each application server process knows the health and status of JVMs and singleton services. DCS provides a view of synchronous services to the high availability manager. DCS itself uses reliable multicast messaging (RMM) as its publish / subscribe message framework.

RMM is an ultra high speed publish / subscribe system that WebSphere uses internally for its core group communication fabric as well as for DRS traffic. WebSphere Application Server for z/OS running in sysplex environment can use cross-system coupling facility services as an alternate protocol provider to enhance DCS performance. It reduces the consumption of system resources for DCS traffic and especially view changes. For more information, refer to “DCS alternate protocol providers” on page 670.

High availability manager provides the following services:

- ▶ Memory-to-memory replication
- ▶ Singleton failover
- ▶ Workload management routing
- ▶ On-demand configuration routing

We describe these services in the sections that follow.

Memory-to-memory replication

The data replication service (DRS) that is provided with the WebSphere Application Server is used to replicate HTTP session data, stateful EJB sessions, and dynamic cache information among cluster members. When DRS is configured for memory-to-memory replication, the transport channels that are defined for the high availability managers are used to pass this data among the cluster members.

Singleton failover

Singleton failover is a cluster-based service. Singleton services that use this framework include the transaction managers for cluster members and the default messaging provider, also known as the service integration bus.

Workload management routing

In the following section, we refer to workload management (WLM) as the internal WebSphere component and Workload Manager as the z/OS system component.

Workload management propagates the following classes or types of routing information:

- ▶ Routing information for the default messaging engine, which is also referred to as the service integration bus
- ▶ Routing HTTP requests through the IBM WebSphere Application Server proxy server
- ▶ Routing Web Services Addressing requests through the IBM WebSphere Application Server proxy server
- ▶ Routing Session Initiation Protocol (SIP) requests.

WLM uses the high availability manager to both propagate the routing information and to make it highly available. Although WLM routing information typically applies to clustered resources, it can also apply to non-clustered resources, such as stand-alone messaging engines.

The Workload Manager for z/OS system component is working together with the sysplex distributor to provide intelligent routing and workload balancing. When dynamic virtual IP address (DVIPA) is used as the daemon IP name for the cell, Workload Manager capabilities are used for workload balancing and failover of IIOP requests between the LPARs. Location service daemons provide the CORBA location service in support of Remote Method Invocation and Internet Inter-ORB Protocol (RMI/IIOP).

In a cell, one location service daemon definition exists for each sysplex node group. A location service daemon process runs on each system that has a node in a sysplex node group in that cell. When a client makes a remote call to an enterprise bean, a location service daemon determines which server or servers are eligible to process the request, and routes the request to the selected server. This mechanism is highly efficient.

For more information about Workload Manager, refer to 16.10.1, “The concept of workload management on z/OS systems” on page 625.

On-demand configuration routing

In a WebSphere Application Server Network Deployment system with high availability manager enabled, the on-demand configuration routing is used for IBM WebSphere Application Server proxy server routing.

State data exchange

The high availability manager provides a specialized messaging mechanism that enables processes to exchange information about their current state. Each process sends or posts information related to its current state, and can register to be notified when the state of the other processes changes. This mechanism is commonly referred to as the bulletin board. The workload management (WLM) component uses this mechanism to build and maintain routing table information. Routing tables built and maintained using this mechanism are highly available.

For more information about high availability configuration on z/OS, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.zseries.doc/info/zseries/ae/crun_ha_config.html

For information about how to disable high availability manager on a process, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.nd.doc/info/ae/ae/trun_ha_ham_enable.html

Disabling the high availability manager: When you disable the high availability manager on a core group member, make sure that you disable it on all the other core group members as well. You can define a special core group for the high availability manager disabled processes.

Do not disable the high availability manager on administrative processes, such as node agents and the deployment manager, unless the high availability manager is disabled on all processes they manage.

Also, leave high availability manager enabled on any application server that produces or consumes either IIOP or messaging engine routing information.

For more information about IIOP support with high availability infrastructure disabled, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.zseries.doc/info/zseries/ae/trun_wlm_cluster_routetable.html

17.2.2 Core groups

A *core group* is a high availability domain that consists of a set of processes in the same cell that can directly establish high availability relationships. Highly available components can fail over only to another process in the same core group and replication can occur only between members of the same core group.

A cell must contain at least one core group, although multiple core groups are supported. By default, a cell has a single core group, called *DefaultCoreGroup*. A single core group is usually sufficient. However, some topologies or special circumstances require multiple core groups. Bridges can be established between core groups.

By default, every deployment manager, node agent, application server, and proxy server is a member of a core group and has high availability manager service enabled. When a process is created, it is added automatically to a core group. The core group membership is stored in a WebSphere Application Server configuration document. You can move processes from one core group to another. When a core group member starts, the core group transport and the associated default Discovery Protocol, default Failure Detection Protocol, and View Synchrony Protocol also start.

Figure 17-6 shows high availability manager protocol traffic within a core group.

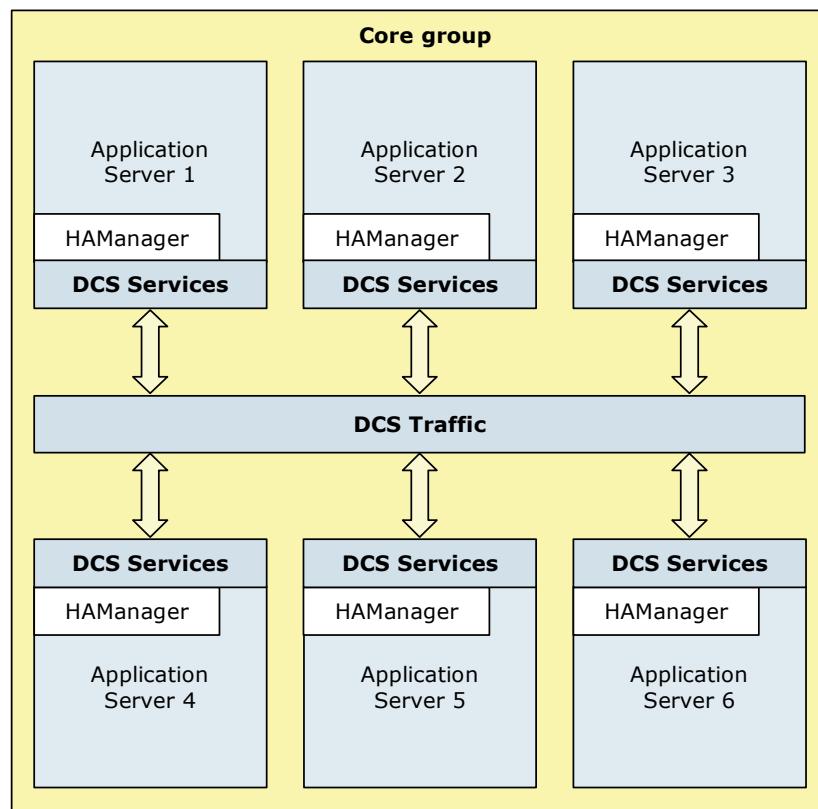


Figure 17-6 Core group DCS traffic

Each core group contains a core group coordinator to manage its high availability relationships and a set of high availability policies that are used to manage the highly available components within that core group.

The following aspects are managed by the core group coordinator:

- ▶ Maintaining all group information, including the group name, group members, and the policy of the group
- ▶ Keeping track of the states of group members as they start, stop, or fail, and communicating that information to every member
- ▶ Assigning singleton services to group members and handling failover of services based on core group policies

When a JVM process with the active coordinator is no longer active (because it is stopped or crashes), the high availability manager elects the first inactive server in the preferred coordinator servers list. If there are no servers available, the high availability manager elects the lexically lowest named inactive server.

The newly elected coordinator initiates a state rebuild, sending a message to all JVMs in the core group to report their states. This operation is the most processor-intensive operation of a coordinator.

Creating a new core group

To create a new core group, complete the following steps:

1. Click **Servers** → **Core Groups** → **Core group settings**.
2. Click **New** to create new core group, as shown in Figure 17-7.



Figure 17-7 Create a new core group

- Specify the name, the number of coordinators, and the transport memory size in the Configuration tab, as shown in Figure 17-8. Click **Apply**.

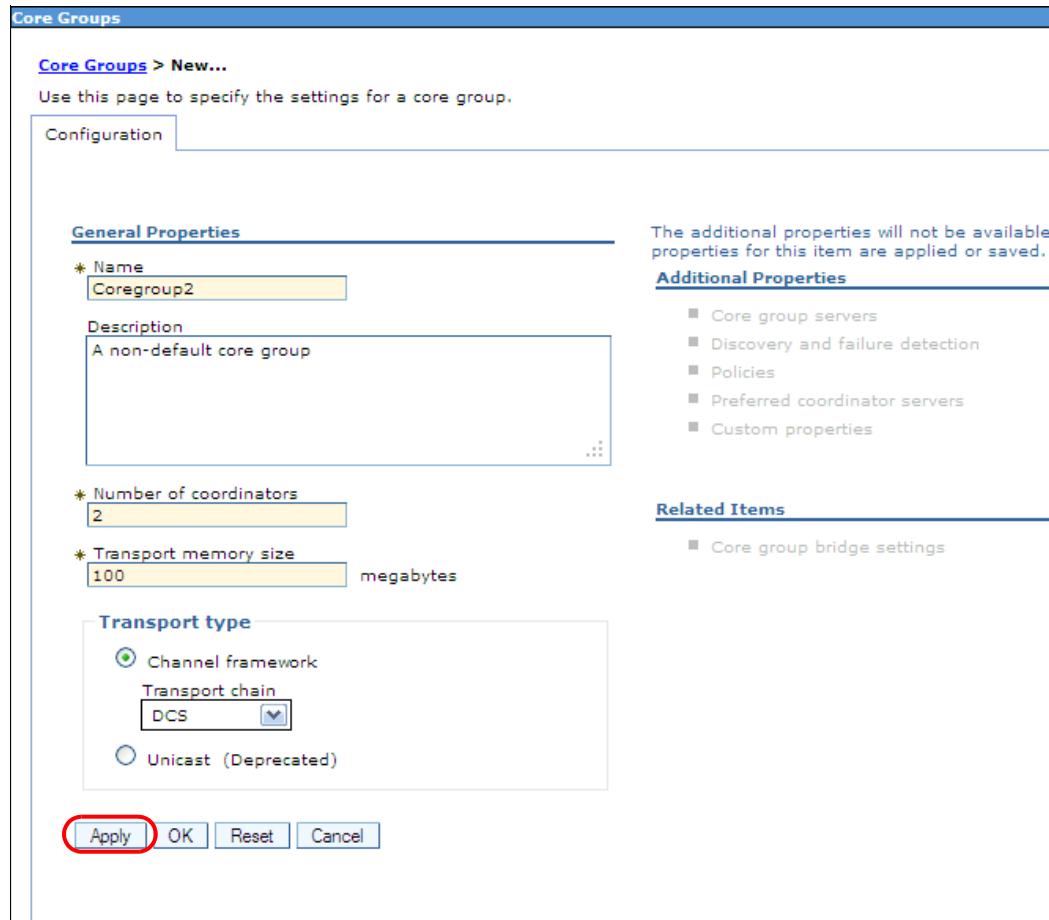


Figure 17-8 New core group window in core group view

- Next, you can add Additional Properties.
- Click **Review**, click **Synchronize changes with nodes**, and click **Save**. Alternatively, click **System administration** → **Nodes** and click **Synchronize** with the appropriate node or nodes selected or enable the **Synchronize changes with Nodes** option in **System administration** → **Console Preferences**.

Inherited values: The number of coordinators and the transport memory size values for the new core group are inherited from the DefaultCoreGroup.

The core group should not have more than 50 members. If your cell has more than 50 processes defined, consider creating a second core group and moving the overflow members into it.

Moving core group members between core groups

You can move processes from one core group to another, as long as the following core group requirements are not violated:

- A non-empty core group retains at least one node agent or deployment manager as a member of that group. (The high availability manager configuration change listeners are available only on the node agent or deployment manager servers.)

- ▶ All members of a cluster must be members of the same core group. If one or more of the servers that you are moving belongs to a cluster, you must move all of the members of that cluster. A core group can span multiple product clusters.

To move core group members to another core group, complete the following steps:

1. Click **Servers** → **Core Groups** → **Core group settings**, as shown in Figure 17-11 on page 659.
2. Select your desired core group, as shown in Figure 17-9.

The screenshot shows the 'Core Groups' management interface. At the top, there's a header bar with tabs for 'Core Groups' and 'Deployment Manager'. Below the header, a descriptive text explains what a core group is. There are buttons for 'New...', 'Delete', and other actions. A table lists the core groups, with one entry highlighted: 'DefaultCoreGroup'. A tooltip for this entry states: 'Default Core Group. The default core group cannot be deleted.' The table also includes columns for 'Select', 'Name', and 'Description'. At the bottom, it says 'Total 1'.

Figure 17-9 Core Groups view

3. Click **Core group servers** under Additional Properties, as shown in Figure 17-10.

The screenshot shows the 'Core Groups > DefaultCoreGroup' settings page. It has tabs for 'Runtime' and 'Configuration'. Below these are sections for 'General Properties' (with a 'Name' field containing 'DefaultCoreGroup') and 'Additional Properties'. A link labeled 'Core group servers' is circled in red.

Figure 17-10 Specifying additional properties

4. Select the members that you want to move to another core group and click **Move**, as shown in Figure 17-11.

Core Groups

[Core Groups](#) > [DefaultCoreGroup](#) > [Core group servers](#)

Use this page to view and manage the servers that belong to a core group. A core group server can be an application server, a manager, or a node agent that is a member of a high availability core group.

Preferences

Move...

Select	Name	Node	Version	Type
<input checked="" type="checkbox"/>	BBOABC1	bbonode	ND 8.0.0.0	Application Server
<input checked="" type="checkbox"/>	BBOABC2	bbonod2	ND 8.0.0.0	Application Server

Figure 17-11 Select members

5. From a drop-down menu, select a core group name to which to move members and click **Apply**, as shown in Figure 17-12.

Core Groups

[Core Groups](#) > [DefaultCoreGroup](#) > [Core group servers](#) > [Move](#)

Use this page to move core group servers from one core group to another.

All members of a cluster must belong to the same core group. If you move one or more servers, all cluster members are preselected for you because all of them must be moved.

You must stop a core group server before you move it.

Configuration

General Properties

* Move selected servers
bbonode(ND)/BBOABC1 bbonod2(ND)/BBOABC2

* From core group
DefaultCoreGroup

* To core group
Coregroup2

Apply OK Reset Cancel

Figure 17-12 Move core group servers

6. Click **Review**, click **Synchronize changes with nodes**, and click **Save**. Alternatively, click **System administration** → **Nodes** and click **Synchronize** with the appropriate node or nodes selected or enable the **Synchronize changes with Nodes** option in **System administration** → **Console Preferences**.

Important: You cannot move a core group member to another core group while it is running.

Each process can be a member of only one core group, and all members of a given cluster must belong to the same core group.

You need to follow a special procedure when moving node agents or deployment manager processes. For more information, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.zseries.doc/info/zseries/ae/trun_ha_addcgmems.html

Setting core group custom properties

High availability protocols include the following major categories:

- ▶ A collection of lower level protocols, which are also referred to as the *lower-level wire format protocols*. The setting for the IBM_CS_WIRE_FORMAT_VERSION core group custom property determines the protocol version that is used for this group of protocols.
- ▶ A collection of higher level protocols, which are also referred to as the *high availability manager protocols*. The setting for the IBM_CS_HAM_PROTOCOL_VERSION core group custom property determines the protocol version that is used for this group of protocols.

To set or change core group custom properties, complete the following steps:

1. Click **Servers** → **Core Groups** → **Core group settings**.
2. Select your desired core group, as shown in Figure 17-13.

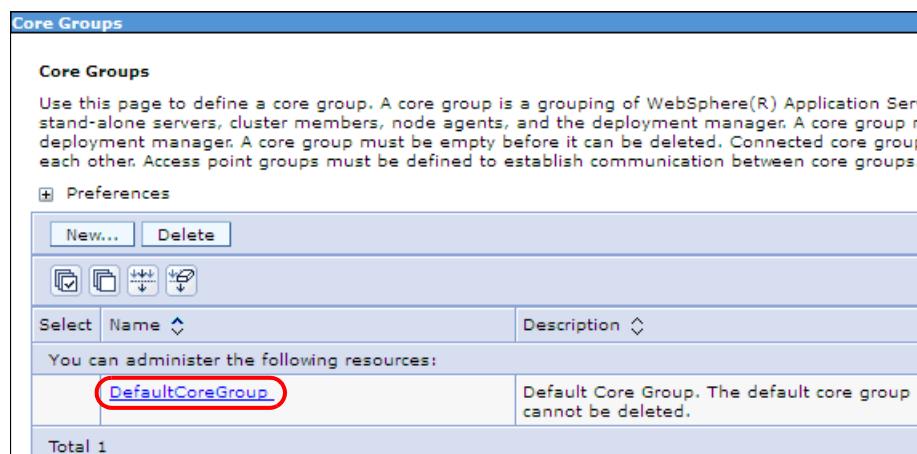


Figure 17-13 Select the core group

3. Select **Custom properties** under Additional Properties, as shown in Figure 17-14.

The screenshot shows the 'Core Groups' interface with the 'DefaultCoreGroup' selected. The 'Runtime' tab is active. In the 'General Properties' section, there is a field for 'Name' containing 'DefaultCoreGroup' and a 'Description' field containing 'Default Core Group. The default core group cannot be deleted.'. On the right, under 'Additional Properties', there is a list of options: 'Core group servers', 'Discovery and failure detection', 'Policies', 'Preferred coordinator servers', and 'Custom properties'. The 'Custom properties' option is highlighted with a red oval.

Figure 17-14 Specifying additional properties

4. Click **New** to define the set of properties for IBM_CS_HAM_PROTOCOL_VERSION and IBM_CS_WIRE_FORMAT_VERSION, as shown in Figure 17-15.

The screenshot shows the 'Core Groups' interface with the 'DefaultCoreGroup' selected and the 'Custom properties' section open. The 'New...' button is visible. A table lists two properties: 'IBM_CS_HAM_PROTOCOL_VERSION' with value '6.0.2.31' and 'IBM_CS_WIRE_FORMAT_VERSION' with value '6.1.0'. The total count is shown as 'Total 2'.

Select	Name	Value
<input type="checkbox"/>	IBM_CS_HAM_PROTOCOL_VERSION	6.0.2.31
<input type="checkbox"/>	IBM_CS_WIRE_FORMAT_VERSION	6.1.0

Figure 17-15 Custom properties

5. Click **Apply**.
6. Click **Review**, click **Synchronize changes with nodes**, and click **Save**. Alternatively, click **System administration** → **Nodes** and click **Synchronize** with the appropriate node or nodes selected or enable the **Synchronize changes with Nodes** option in **System administration** → **Console Preferences**.

Hint: By default, the protocol is set to oldest supported version. Investigate the latest version for the group of protocols that your installation supports and add the protocol custom properties, as shown in Figure 17-15, for protocol improvements.

For a list of core group custom properties along with supported protocol versions, visit:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Finfo%2Fzseries%2Fae%2Furun_ha_cg_custprop.html

The DCSV0005I and HMGR0226I messages indicate the currently used version of the protocols, as shown in Example 17-1 and Example 17-2.

Example 17-1 Message HMGR0226I output

ExtendedMessage: BB000222I: HMGR0226I: The core stack configuration parameter IBM_CS_WIRE_FORMAT_VERSION has been set to 6.0.2.31.

Example 17-2 Message DCSV0005I output

ExtendedMessage: BB000222I: DCSV0005I: DCS Stack DefaultCoreGroup at Member bbocell1\bbodmgr\dmgr: Started. Stack version information: DCSBV_WAS6_1_20060409. Stack protocol information: 61002.

Support note: If you are running on Version 7.0.0.1 or later, set the IBM_CS_HAM_PROTOCOL_VERSION core group custom property to 6.0.2.31 for all core groups to avoid a possible high availability state outage during core group bridge failover. When this custom property is set to 6.0.2.31, the remaining bridges recover the high availability state of the failed bridge without the data being unavailable in the local core group.

Bridging core groups

If members of different core groups need to share WLM routing information, use the *core group bridge service* to connect these core groups. The core group bridge service uses access point groups to connect the core groups. A core group access point defines a set of bridge interfaces that resolve to IP addresses and ports. The core group bridge service uses this set of bridge interfaces to enable members of one core group to communicate with members of another core group.

To create a core group bridge service for two core groups within the same cell with mesh topology, complete the following steps:

1. Click **Servers** → **Core Groups** → **Core group bridge settings**.
2. Select **Access point groups** under Additional Properties, as shown in Figure 17-16.

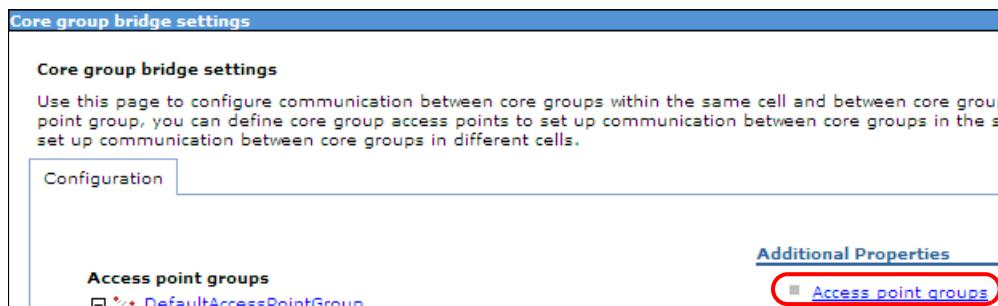


Figure 17-16 Access point groups

3. For our topology, we select **DefaultAccessPointGroup**, as shown in Figure 17-17. To use chain topology, you need to create multiple access point groups.

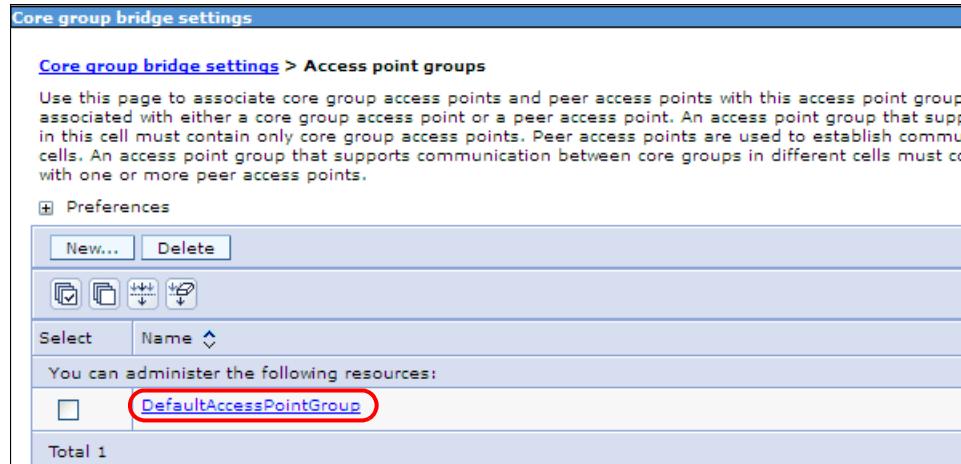


Figure 17-17 Access point groups window

4. Select **Core group access points**, as shown in Figure 17-18.

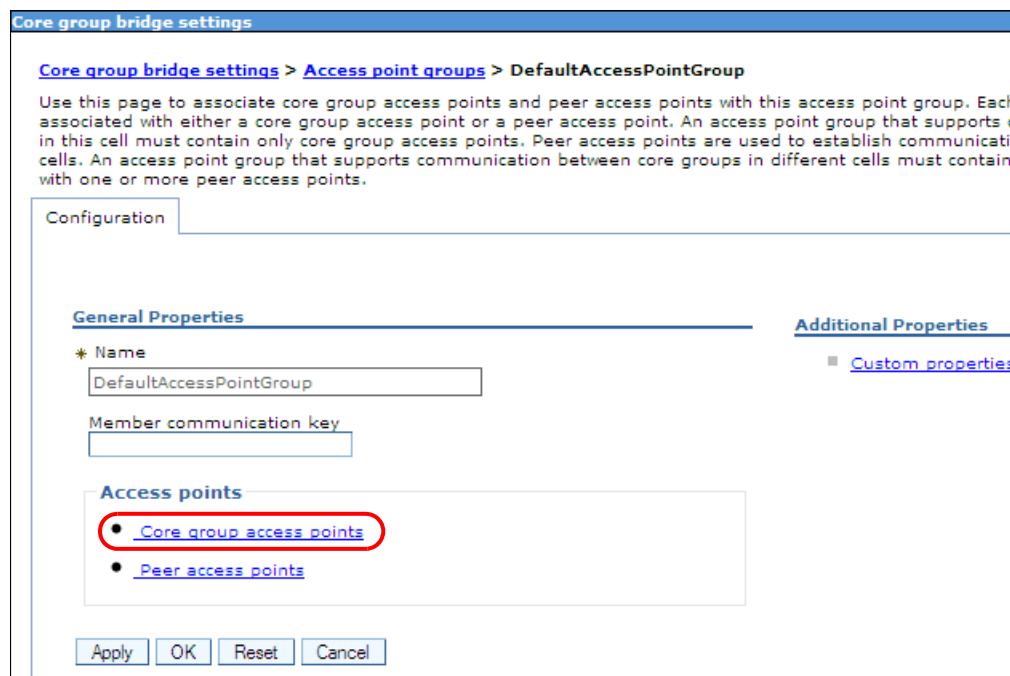


Figure 17-18 DefaultAccessPointGroup configuration window

5. Select **CGAP_1\DefaultCoreGroup** and click **Show Detail**, as shown in Figure 17-19.

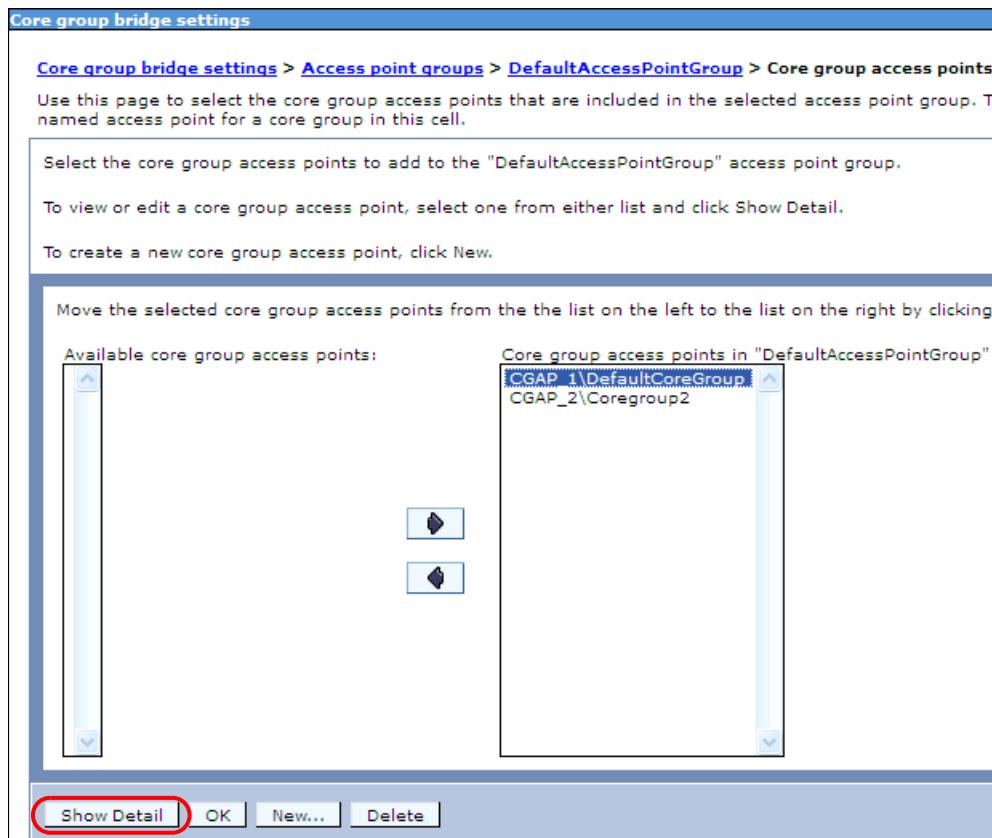


Figure 17-19 Core group access points window

6. Select **Bridge interfaces** from Additional Properties, as shown in Figure 17-20.

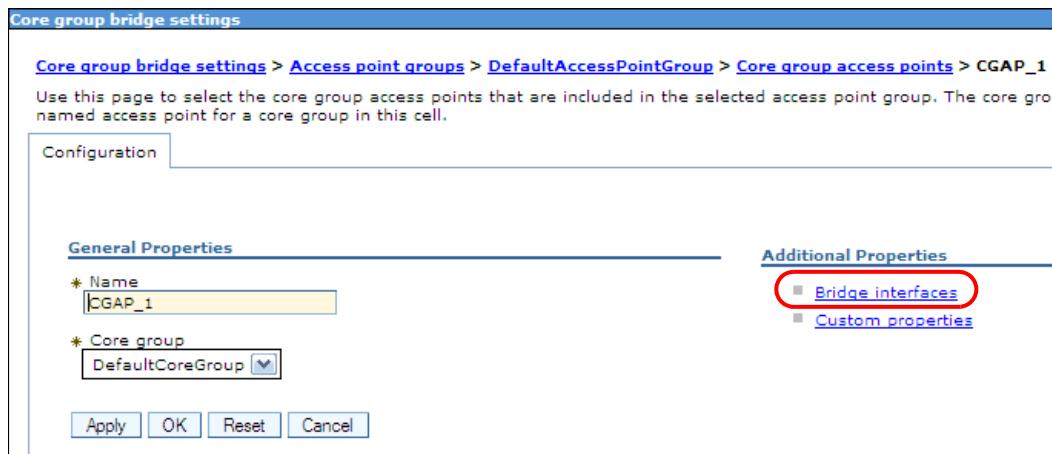


Figure 17-20 CGAP_1 Core group access point window

- Click **New** to create a new Bridge interface for the CGAP_1 Core group access point, as shown in Figure 17-21.

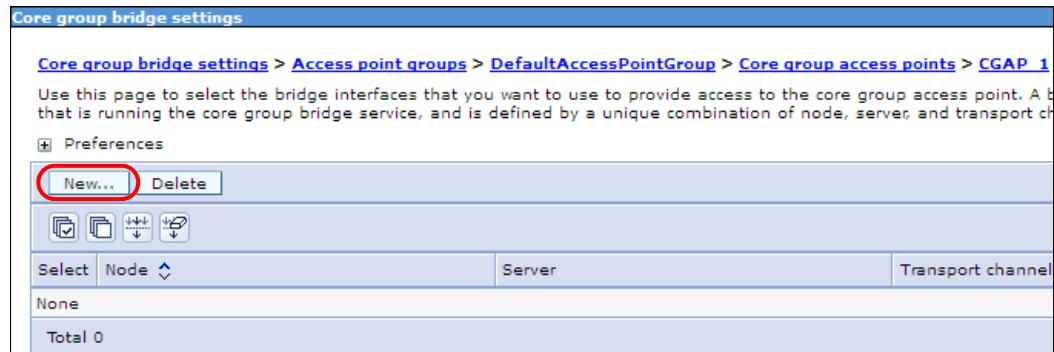


Figure 17-21 CGAP_1 Bridge interfaces window

- Select a process from the Bridge interfaces drop-down menu, as shown in Figure 17-22. You should consider administrative processes first. Click **Apply**.

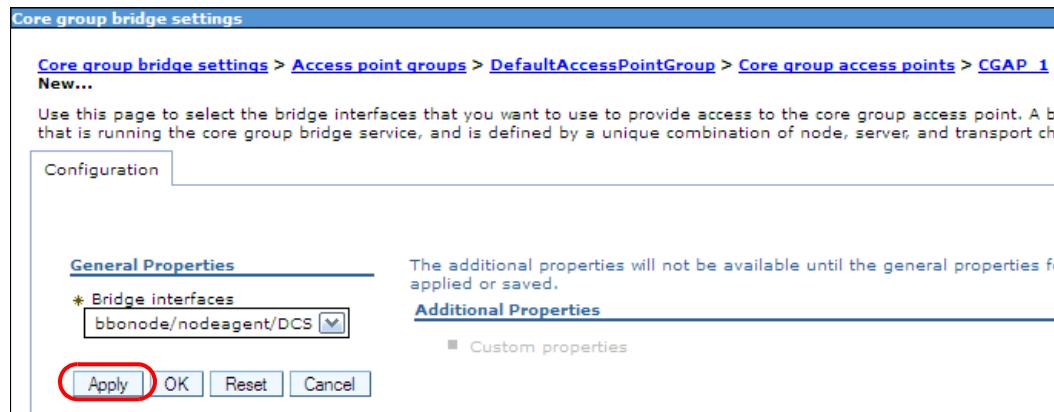


Figure 17-22 New Bridge interfaces window

- Click **Review**, click **Synchronize changes with nodes**, and click **Save**. Alternatively, click **System administration** → **Nodes** and click **Synchronize** with the appropriate node or nodes selected or enable the **Synchronize changes with Nodes** option in **System administration** → **Console Preferences**.
- For high availability purposes, define one more bridge interface by repeating steps 7 to 9 for the first core group.

11. Back on the Core group access points window, select **CGAP_2\Coregroup2** and click **Show Detail**, as shown in Figure 17-23.

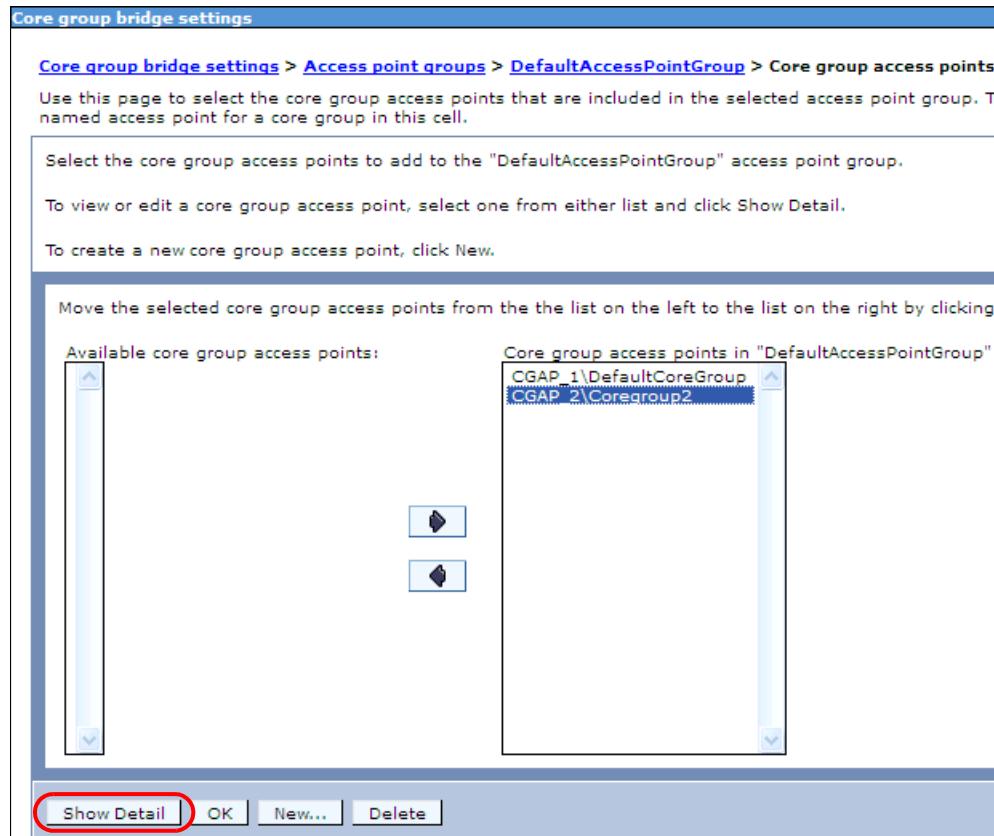


Figure 17-23 Core group access points window

12. Select **Bridge interfaces** from Additional Properties, as shown in Figure 17-24.

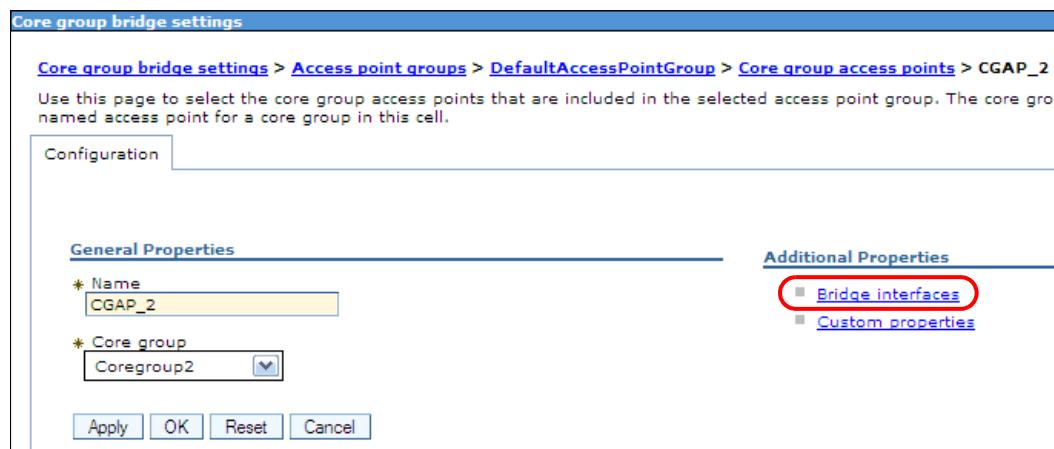


Figure 17-24 CGAP_2 Core group access point window

13. Click **New** to create a new bridge interface for the CGAP_2 Core group access point, as shown in Figure 17-25.



Figure 17-25 CGAP_2 Bridge interfaces window

14. Select a process from the **Bridge interfaces** drop-down menu, as shown in Figure 17-26. Consider administrative processes first. Click **Apply**.

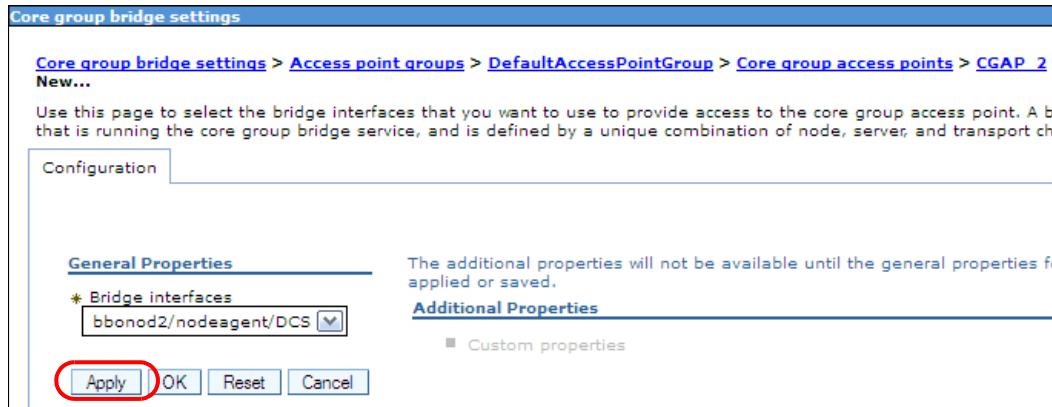


Figure 17-26 New Bridge interfaces window

15. Click **Review**, click **Synchronize changes with nodes**, and click **Save**. Alternatively, click **System administration** → **Nodes** and click **Synchronize** with the appropriate node or nodes selected or enable the **Synchronize changes with Nodes** option in **System administration** → **Console Preferences**.

16. For high availability purposes, define one more bridge interface by repeating steps 13 to 15.

17. Verify your topology setup, as shown in Figure 17-27.

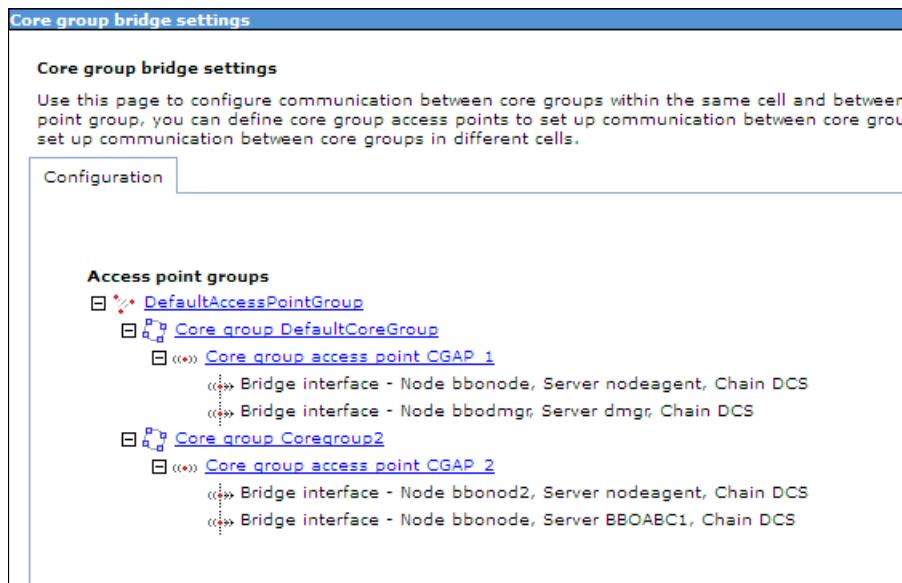


Figure 17-27 Access point groups overview

18. Click **Review**, click **Synchronize changes with nodes**, and click **Save**. Alternatively, click **System administration** → **Nodes** and click **Synchronize** with the appropriate node or nodes selected or enable the **Synchronize changes with Nodes** option in **System administration** → **Console Preferences**.
19. Fully shut down and then restart all core group bridges in the affected access point groups by clicking either **Servers** → **Server Types** → **WebSphere application servers** or **System administration** → **Node agents** or **System administration** → **Deployment manager**. (You must start the deployment manager manually from the system console, using the SDSF command prompt, or using your automation procedures.)

Complete a full shutdown: When you make a change in the core group bridge configuration, including the addition of a new bridge or the removal of an existing bridge, you must *fully shut down* and then restart all core group bridges in the affected access point groups.

Heap size note: If you have multiple large core groups bridged together, consider increasing the heap size of the bridge interfaces and high availability coordinators by 512 MB (as a starting point) and fine-tune heap settings using verbose GC trace monitoring. To increase core group memory settings, consider setting the IBM_CS_DATASTACK_MEG and transport buffer size to 100.

Bridge interfaces: Do not create more than two bridge interfaces per core group or use productive servers as bridge interfaces. Ensure that these interfaces are on different nodes for high availability purposes.

Use administrative processes, such as node agents, deployment manager, or, where possible, dedicated application servers. Process utilization can be high for core bridge interfaces during a core group start.

For more information about core group bridge service, go to the following website:
http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=/com.ibm.websphere.nd.doc/info/ae/ae/trun_ha_coregroupbridge.html

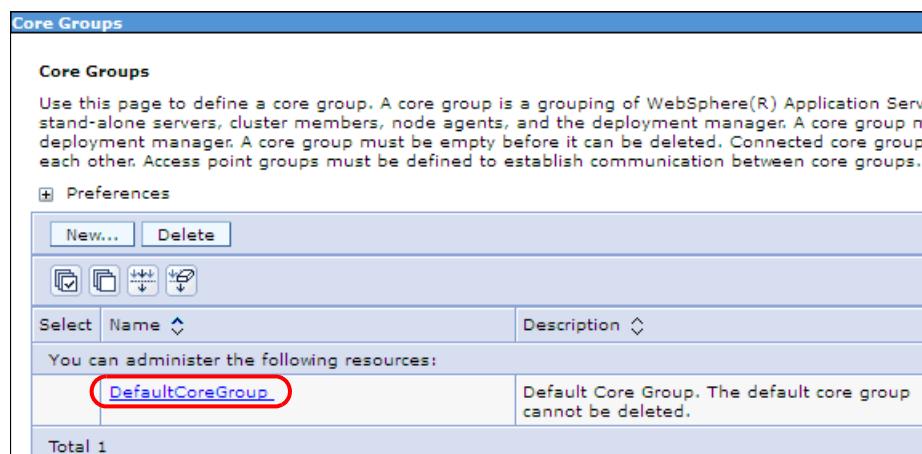
Setting up preferred coordinator servers

By default, the high availability manager elects the lexically lowest named core group process to be the coordinator. The name of the process consists of *cell name*, *node name*, and *process name*.

Because a coordinator takes up additional resources in the JVM, you might want to override the default election mechanism by providing your own list of preferred coordinator servers in the WebSphere Administrative Console.

To set preferred coordinator servers, complete the following steps:

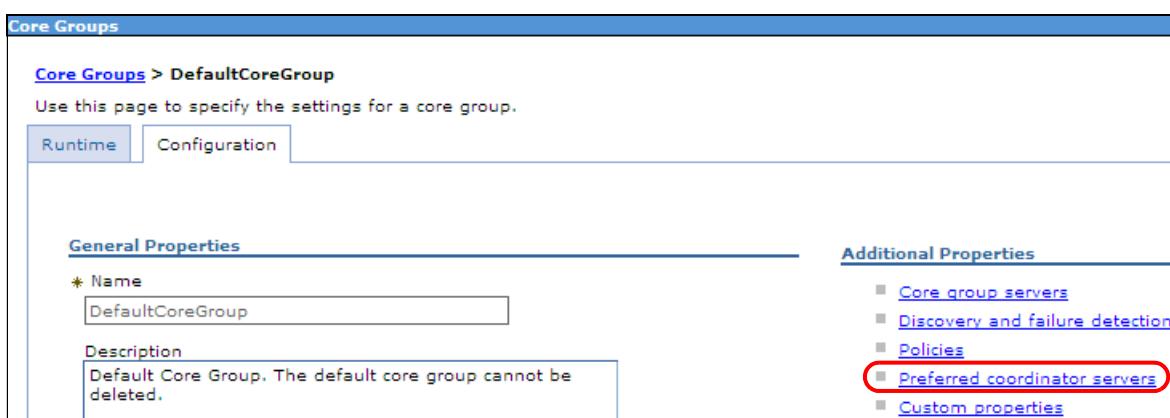
1. Click **Servers** → **Core Groups** → **Core group settings**.
2. Select your desired core group, as shown in Figure 17-28.



The screenshot shows the 'Core Groups' page in the WebSphere Administrative Console. At the top, there is a brief description of what a core group is. Below it is a toolbar with 'New...', 'Delete', and other icons. A table lists core groups, with 'DefaultCoreGroup' highlighted by a red circle. The table columns are 'Select', 'Name', and 'Description'. The 'Name' column for DefaultCoreGroup contains the text 'DefaultCoreGroup'. The 'Description' column states 'Default Core Group. The default core group cannot be deleted.' At the bottom of the table, it says 'Total 1'.

Figure 17-28 Selecting the core group

3. Click **Preferred coordinator servers** under Additional Properties, as shown in Figure 17-29.



The screenshot shows the 'Core Groups > DefaultCoreGroup' page. It has tabs for 'Runtime' and 'Configuration', with 'Configuration' selected. On the left, there's a 'General Properties' section with a 'Name' field containing 'DefaultCoreGroup' and a 'Description' field with the same text. On the right, there's an 'Additional Properties' section with several options: 'Core group servers', 'Discovery and failure detection', 'Policies', 'Preferred coordinator servers' (which is highlighted with a red circle), and 'Custom properties'.

Figure 17-29 Specify preferred coordinator servers

- Add the core group servers to the list of preferred coordinator servers and organize their sequence, as shown in Figure 17-30. Click **OK**.

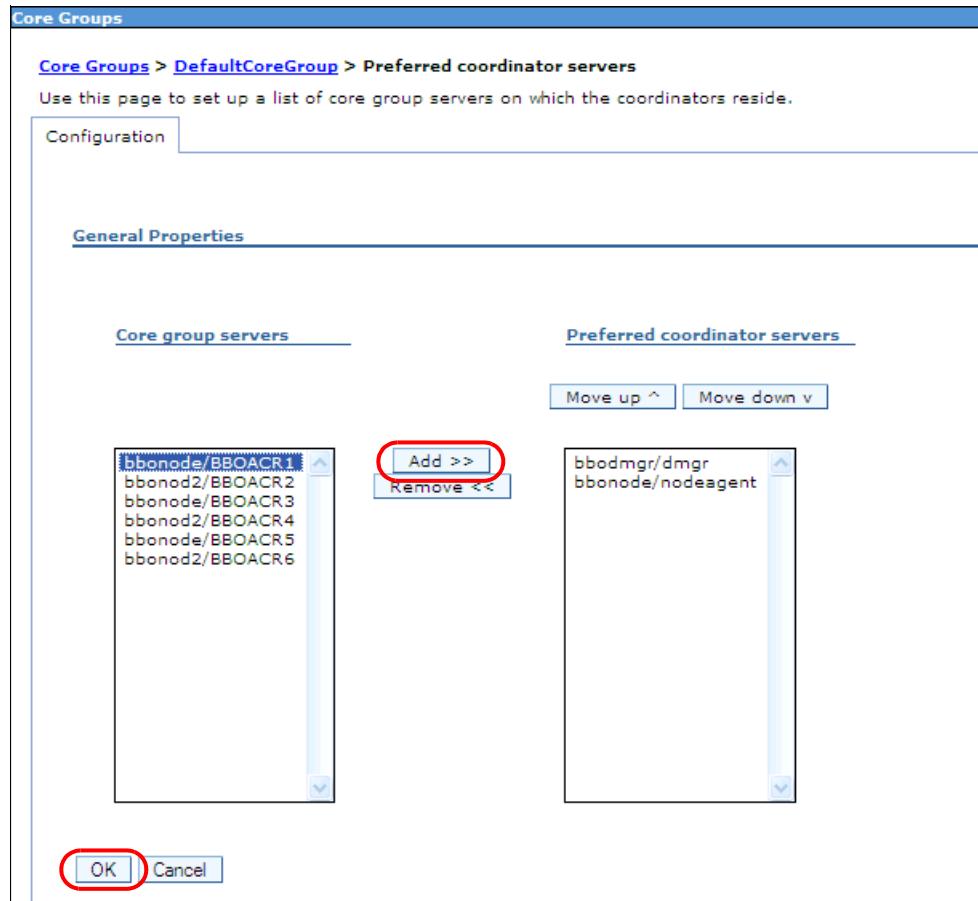


Figure 17-30 Specific core group preferred coordinator servers configuration window

- Click **Review**, click **Synchronize changes with nodes**, and click **Save**. Alternatively, click **System administration** → **Nodes** and click **Synchronize** with the appropriate node or nodes selected or enable the **Synchronize changes with Nodes** option in **System administration** → **Console Preferences**.

DCS alternate protocol providers

In general, alternate protocol providers, such as the z/OS cross-system coupling facility (XCF)-based provider, uses less system resources than the default Discovery Protocol and Failure Detection Protocol, especially during times when the core group members are idle.

An alternate protocol provider generally uses less system resources because it does not perform the member-to-member TCP/IP pinging that the default protocol providers use to determine whether a core group member is still active. If you decide to use the z/OS XCF-based protocol provider, understand that at system start, the server process is joined as a member to an XCF group. The XCF group contains all of the active members for the core group. XCF provides notification to all of the members of this group when a member joins the group and when a member can no longer be contacted because the server shut down or because XCF determines that the server process has terminated.

To enable high availability manager DCS signalling using XCF as alternate protocol, complete the following steps:

1. Click **Servers** → **Core Groups** → **Core group settings**.
2. Select your desired core group, as shown in Figure 17-31.



Figure 17-31 Selecting the core group

3. Click **Discovery and failure detection** from Additional Properties, as shown in Figure 17-32.

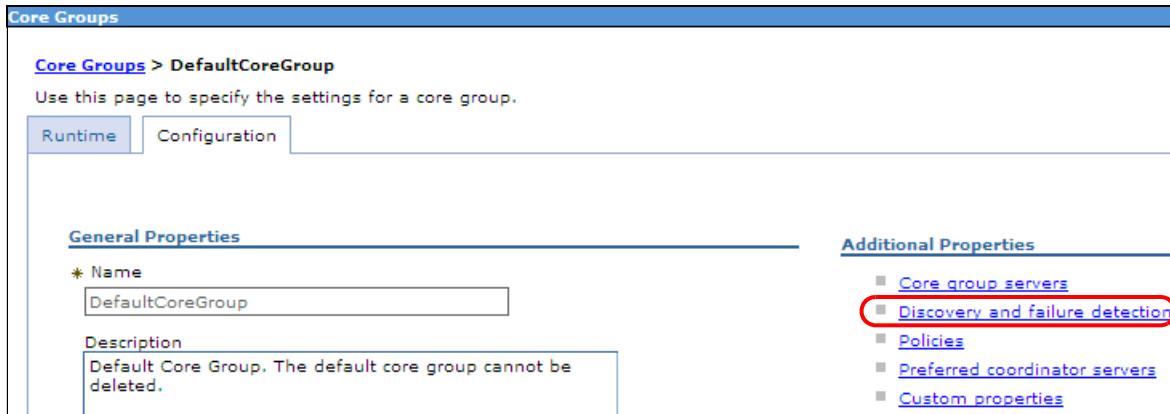


Figure 17-32 Discovery and failure detection

4. Select the **Use alternative protocol providers** option, and enter the class name com.ibm.ws.xcf.groupservices.LivenessPluginZoSFactory, as shown in Figure 17-33. Click **Apply**.

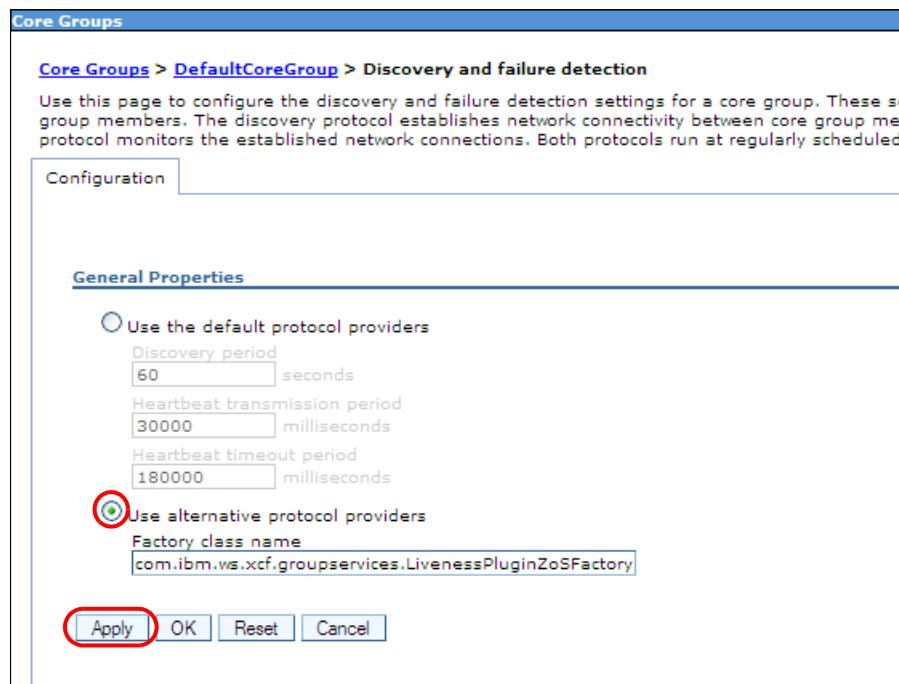


Figure 17-33 Discovery and failure detection window

5. Click **Review**, click **Synchronize changes with nodes**, and click **Save**. Alternatively, click **System administration** → **Nodes** and click **Synchronize** with the appropriate node or nodes selected or enable the **Synchronize changes with Nodes** option in **System administration** → **Console Preferences**.
6. Restart all the core group members by clicking **Servers** → **Server Types** → **WebSphere application servers** or **System administration** → **Node agents** or **System administration** → **Deployment manager**. (You must start the deployment manager manually from the system console, using the SDSF command prompt, or using your automation procedures.)

Consideration: Ensure that the core group includes only servers of Version 7 or higher and that all the members are running on the same z/OS operating system (homogenous environment). Bridged core groups also need to adhere to this restriction because they need to use the same DCS protocol.

Verify that IBM VTAM® starts with the XCFINIT=YES option in ATCSTRxx to use XCF services.

For more information about XCF signalling, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/fep/index.jsp?topic=/com.ibm.websphere.nd.multiplatform.doc/info/ae/ae/trun_ha_cfg_alternate_protocol.html

17.2.3 High availability policies and groups

High availability groups are part of the high availability manager framework. A high availability group provides the mechanism for building a highly available component and enables the component to run in one of several different processes. A high availability group cannot extend beyond the boundaries of a core group.

A high availability group is associated with a specific component. The members of the group are the set of processes where it is possible to run that component. Therefore, a product administrator cannot directly configure or define a high availability group and its associated set of members. Instead, high availability groups are created dynamically at the request of the components for which they need to provide a highly available function.

Every high availability group has an associated policy. The policy is used to determine which members of a high availability group are active at a given point in time. The policies that are available for high availability groups to use are stored as part of the core group configuration. The same policy can be used by several different high availability groups, but all of the high availability groups to which the policy applies must be part of the same core group.

Policy rules are applied in the following circumstances:

- ▶ A member joins or leaves a high availability group
- ▶ The state of a member changes, for example, from idle to disabled

High availability policies

To work with high availability policies, complete the following steps:

1. Click **Servers** → **Core Groups** → **Core group settings**.
2. Select your desired core group, as shown in Figure 17-34.

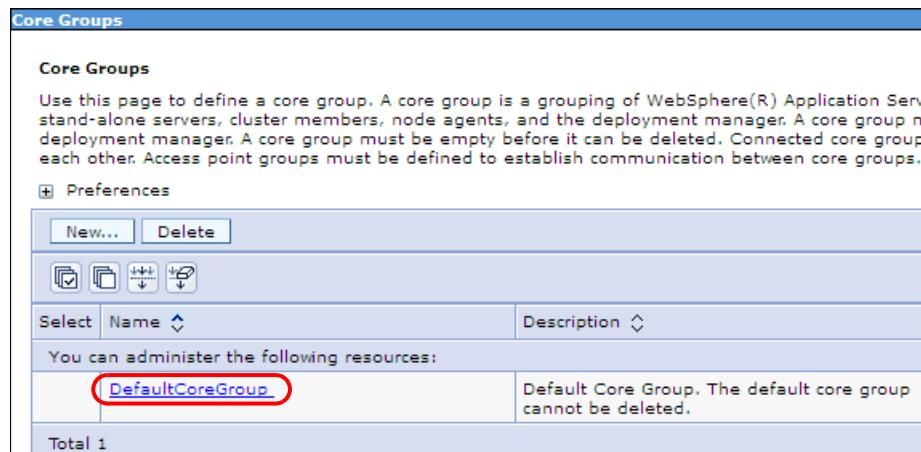


Figure 17-34 Select a core group

3. Click **Policies** from Additional Properties, as shown in Figure 17-35.

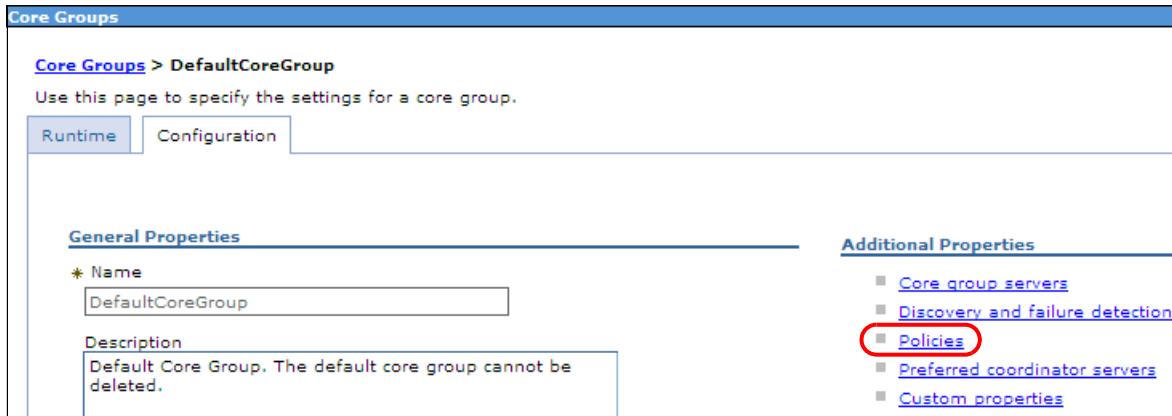


Figure 17-35 Policies

4. You can create new policies to use with components. Figure 17-36 shows the default available policies.

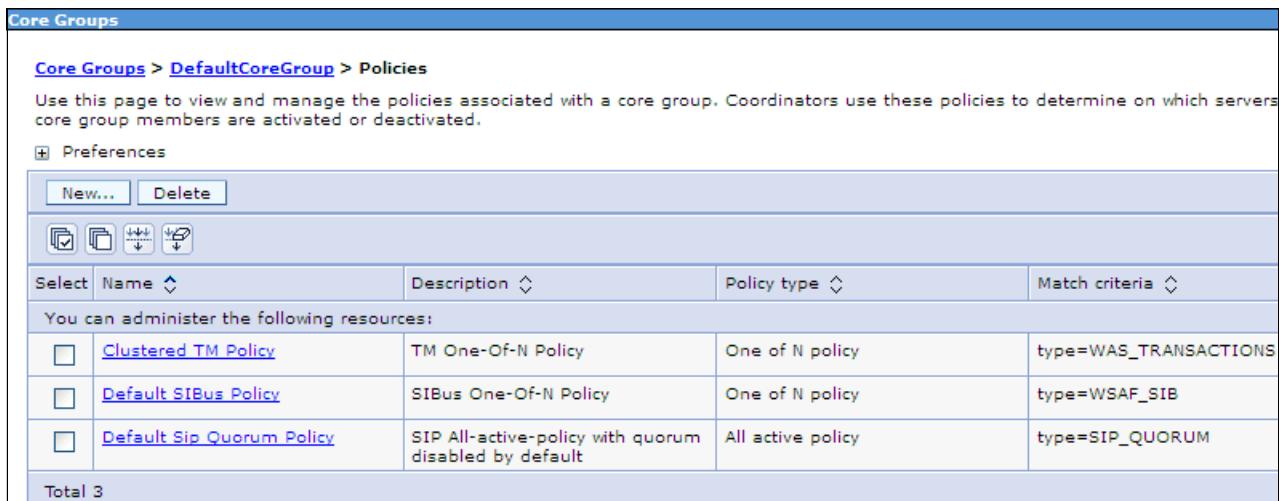


Figure 17-36 Policies window

Configuration changes: The high availability manager dynamically detects policy configuration changes. Therefore, policy setting changes go into effect as soon as you save and propagate these changes. You do not need to restart the server.

Policies note: Do not delete the IBM default provided policies or change the existing policy. Configure a new policy with more name=value pair matching. Ensure that the component that matches a policy supports that policy type.

At least one policy must match and that policy must be the policy with the most number of matches.

Do not configure identical match criteria for multiple policies in the same core group. If there is an ambiguous match, high availability manager puts the high availability group in an error state and does not make any of the group members active.

For instructions about how to create a new high availability policy, go to the following website:
http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=/com.ibm.websphere.nd.doc/info/ae/ae/trun_ha_newpolicy.html

High availability groups

To view high availability groups, complete the following steps:

1. Click **Servers** → **Core Groups** → **Core group settings**.
2. Select your desired core group, as shown in Figure 17-37.



Figure 17-37 Select a core group

3. Go to the **Runtime** tab, as shown in Figure 17-38.

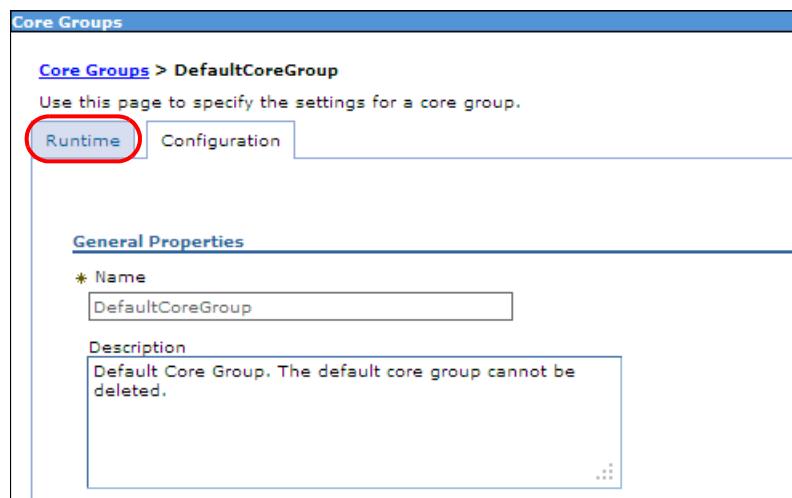


Figure 17-38 Runtime properties

- Click **Show servers** to display core group servers that are hosting active high availability group members. To display high availability groups, specify the name=value pairs or * in the “Group name properties” field and click **Show groups**, as shown in Figure 17-39.

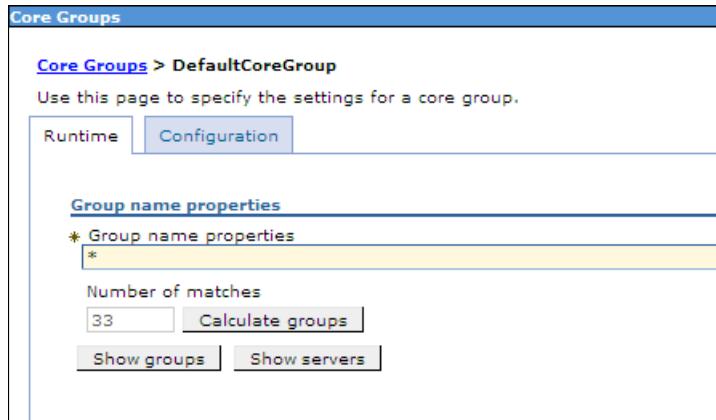


Figure 17-39 Group name properties

17.3 Failover and fallback

Failover is one of the techniques of fault tolerance. It is used to make the system to some degree survive faults and failures.

In this section, we discuss WebSphere Application Server for z/OS failover capabilities.

17.3.1 High availability and failover of singletons

Singleton failover is a cluster-based service. High availability manager has to be enabled on all the members. Failover and subsequent failback depend on the options selected in the high availability service policy. The Preferred servers list option is provided to allow the service run exclusively on the servers in the list in a preferred order. External resources and any other dependencies must be available to all the members that are to run the service in case of a failure of the primary. External clustering software can be used to complement the failover and failback processing.

WebSphere Application Server provides failover and recovery for the following singleton services:

- ▶ Transaction Service
- ▶ Service integration bus

Transaction Service

Transaction service peer recovery processing can only take place between members of the same server cluster. The Transaction Manager supports three different high availability policies to achieve the recovery of transaction logs in a highly available manner:

- ▶ One of N policy

This policy is the default style of peer recovery initiation. If an application server fails, the high availability manager selects another server to perform peer recovery processing on behalf of the failed server.

- ▶ Static policy

This style of peer recovery must be explicitly configured. If an application server fails, the operator can use the Administrative Console to select another server to perform the recovery processing.

- ▶ No Operation policy

This style of peer recovery must be explicitly configured. It indicates that external clustering software is monitoring the Transaction Manager and will fail over to an application server that is configured by the external clustering software to perform the recovery processing.

For more information about transactional high availability, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.zseries.doc/info/zseries/ae/cjta_trans_ha.html

To enable transaction service log recovery, complete the following steps:

1. Click **Servers** → **Server types** → **WebSphere application servers**.
2. Select your desired application server, as shown in Figure 17-40.

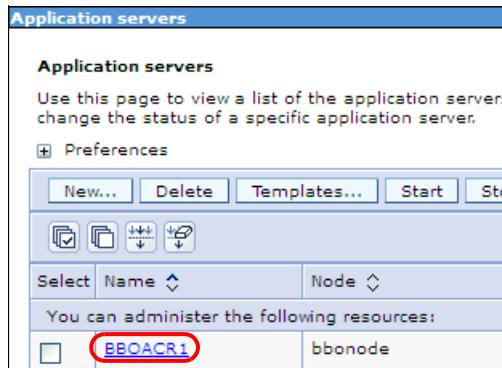


Figure 17-40 Select the application server

3. Under the Container Settings section, expand **Container Services**, and then click **Transaction service**, as shown in Figure 17-41.

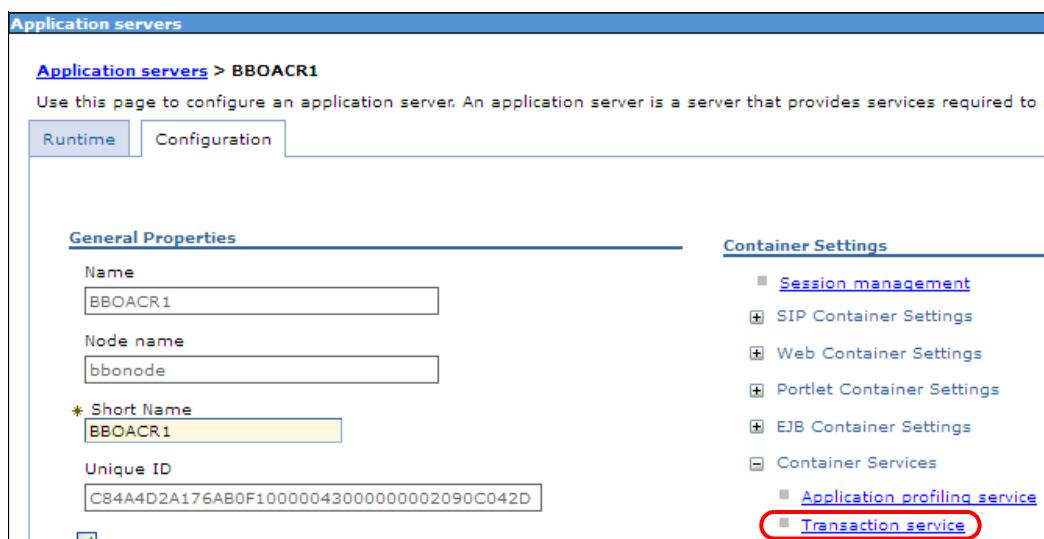


Figure 17-41 Transaction service

- Specify the **Transaction log directory** value, as shown in Figure 17-42. The directory specified should be unique in the cluster and accessible to all the cluster members. It should be allocated on a fast disk.

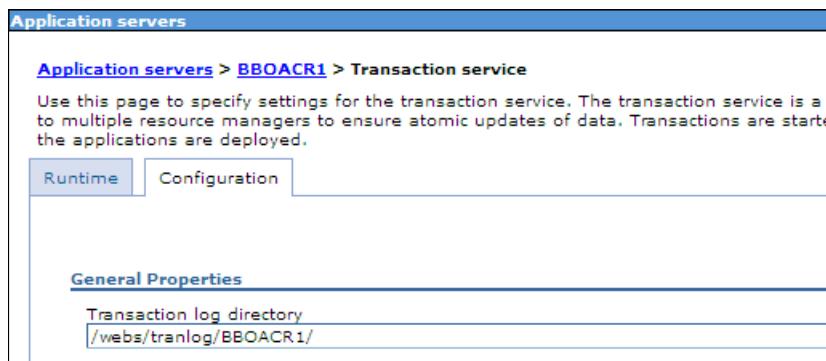


Figure 17-42 Specify the Transaction log directory

- Click **Apply**.
- Click **Review**, click **Synchronize changes with nodes**, and click **Save**. Alternatively, click **System administration** → **Nodes** and click **Synchronize** with the appropriate node or nodes selected or enable the **Synchronize changes with Nodes** option in **System administration** → **Console Preferences**.
- Repeat steps 2 on page 677 to 6 for all the cluster members.
- Click **Servers** → **Clusters** → **WebSphere application server clusters**.
- Select your desired application server cluster, as shown in Figure 17-43.



Figure 17-43 Select application server cluster

10. Enable the **Enable failover of transaction log recovery** option, as shown in Figure 17-44. Click **Apply**.

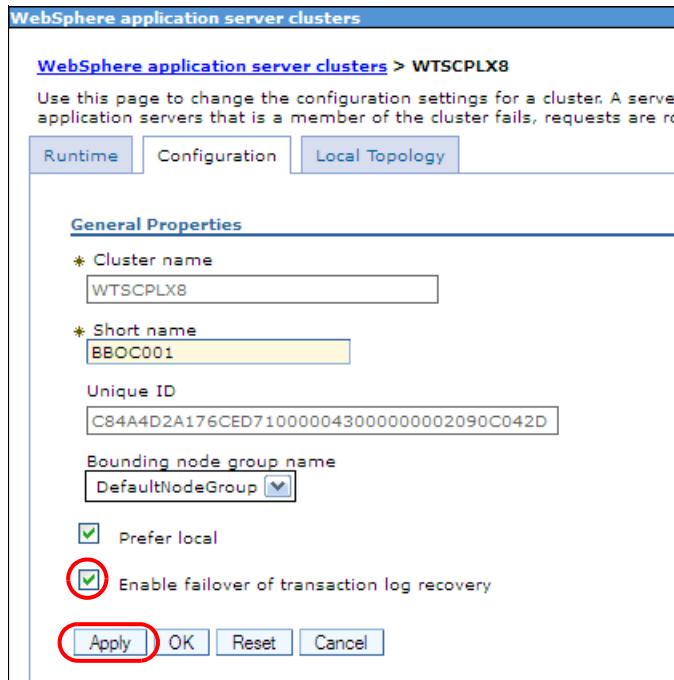


Figure 17-44 Enable failover

11. Click **Review**, click **Synchronize changes with nodes**, and click **Save**. Alternatively, click **System administration** → **Nodes** and click **Synchronize** with the appropriate node or nodes selected or enable the **Synchronize changes with Nodes** option in **System administration** → **Console Preferences**.
12. Restart the affected servers by clicking **Servers** → **Server Types** → **WebSphere application servers**.

Service integration bus

For bus messaging engines, a policy type of One of N should be used. Thus, although the messaging engine can be defined on every server in the cluster, the high availability manager ensures that it is active only on one of the servers in the group and will always be active on one of the servers as long as one is available.

For more information about making the SIBus highly available, complete the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.zseries.doc/info/zseries/ae/cjt0010_.html

To configure a service integration bus (SIBus) with data store persistence, complete the following steps:

1. Click **Service integration** → **Buses**.

- Click **New** to create a new SIBus, as shown in Figure 17-45.

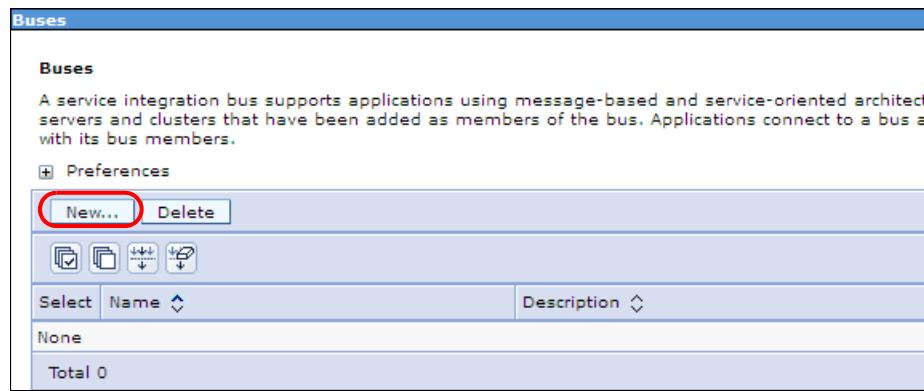


Figure 17-45 New bus

- Enter the name for the new bus and indicate the bus security, as shown in Figure 17-46. Click **Next**.



Figure 17-46 Create a new bus

- Confirm the information, and then click **Finish**, as shown in Figure 17-47.

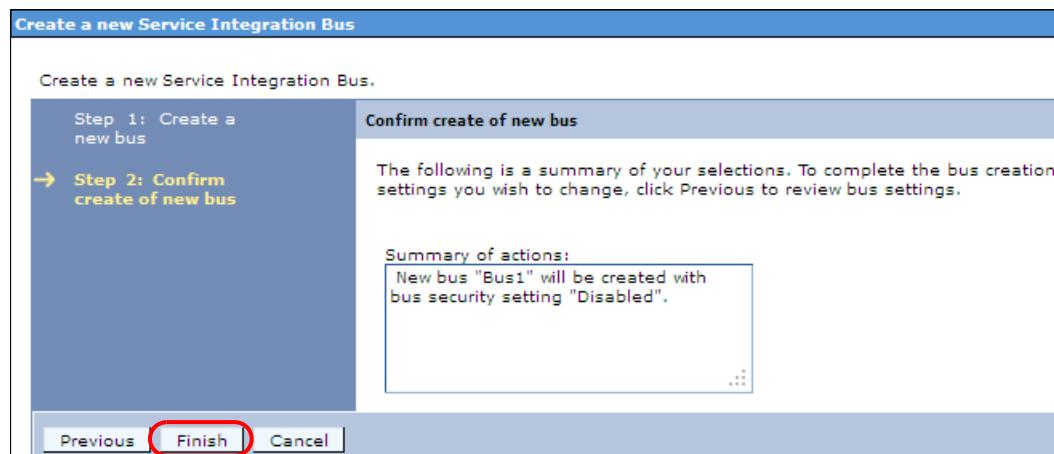


Figure 17-47 Confirm the creation of a new bus

- Click **Review**, click **Synchronize changes with nodes**, and click **Save**. Alternatively, click **System administration** → **Nodes** and click **Synchronize** with the appropriate node or nodes selected or enable the **Synchronize changes with Nodes** option in **System administration** → **Console Preferences**.
- Select the newly created SIBus, as shown in Figure 17-48.

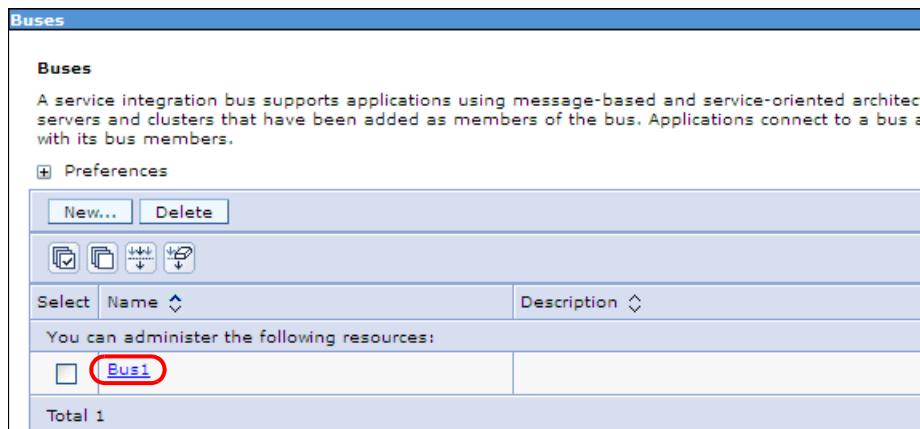


Figure 17-48 Buses view

- In the Topology section, select **Bus members**, as shown in Figure 17-49.

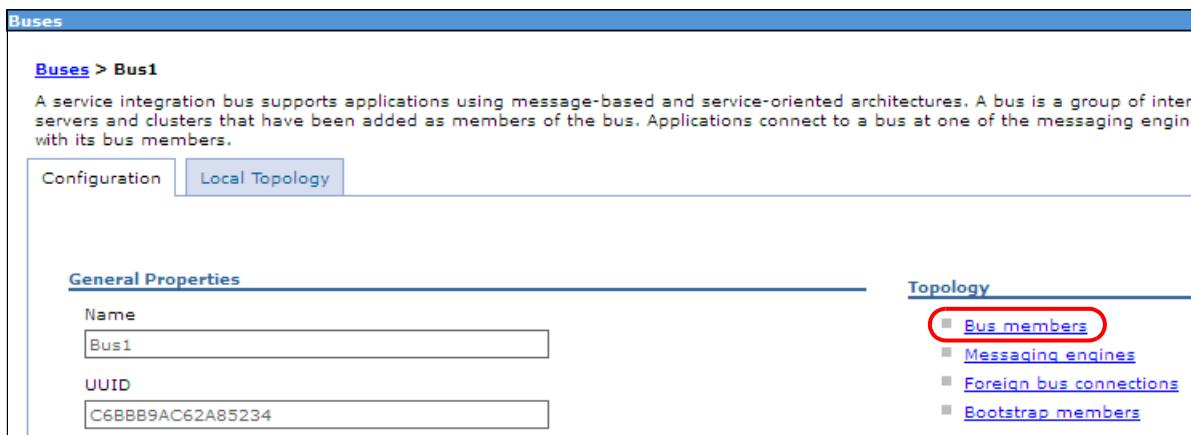


Figure 17-49 Bus members

- Click **Add** to create a new bus member, as shown in Figure 17-50.

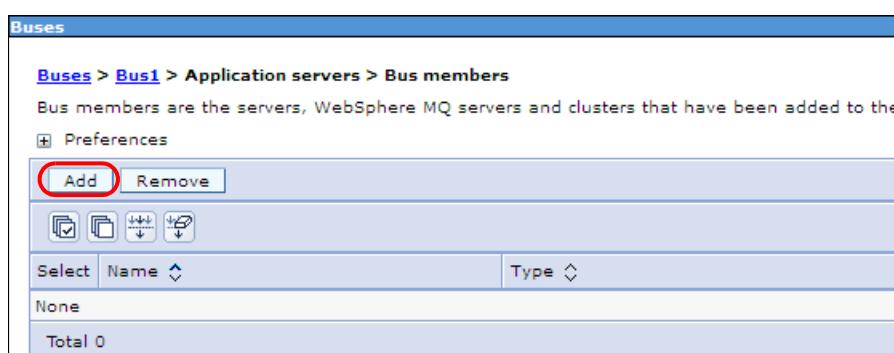


Figure 17-50 Add a new bus member

9. Select the **Cluster** option, and then select the cluster name from the drop-down menu, as shown in Figure 17-51. Click **Next**.

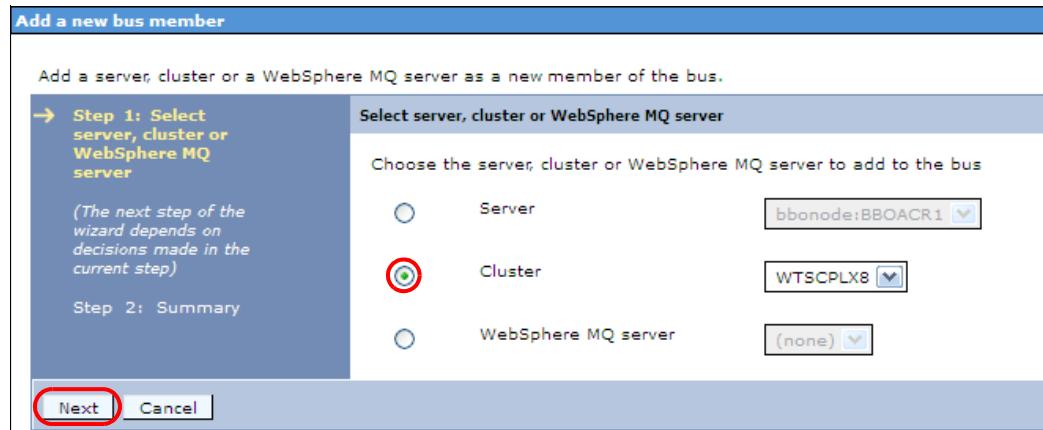


Figure 17-51 Select server, cluster, or WebSphere MQ server window

10. Ensure that the **Messaging engine policy assistance settings** option is enabled, and then select the **Custom** policy type, as shown in Figure 17-52. Click **Next**.

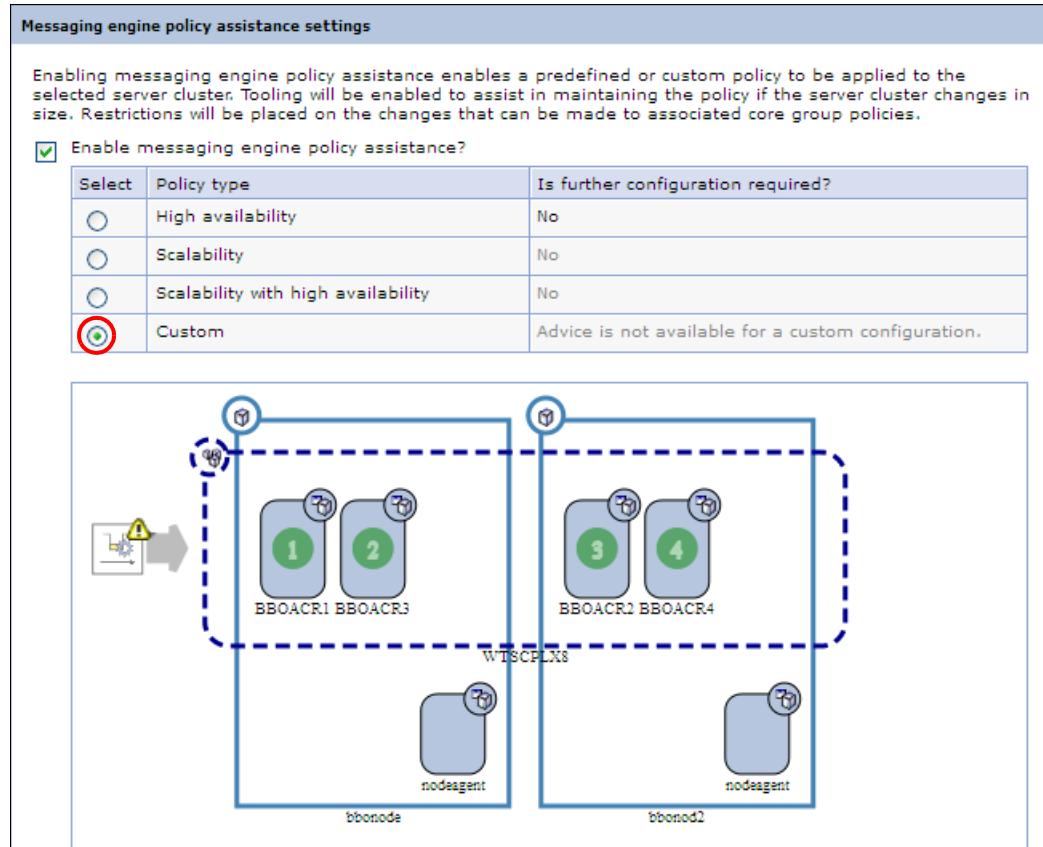


Figure 17-52 Select the Custom messaging engine policy assistance

11. Select the **Data store** option, as shown in Figure 17-53.

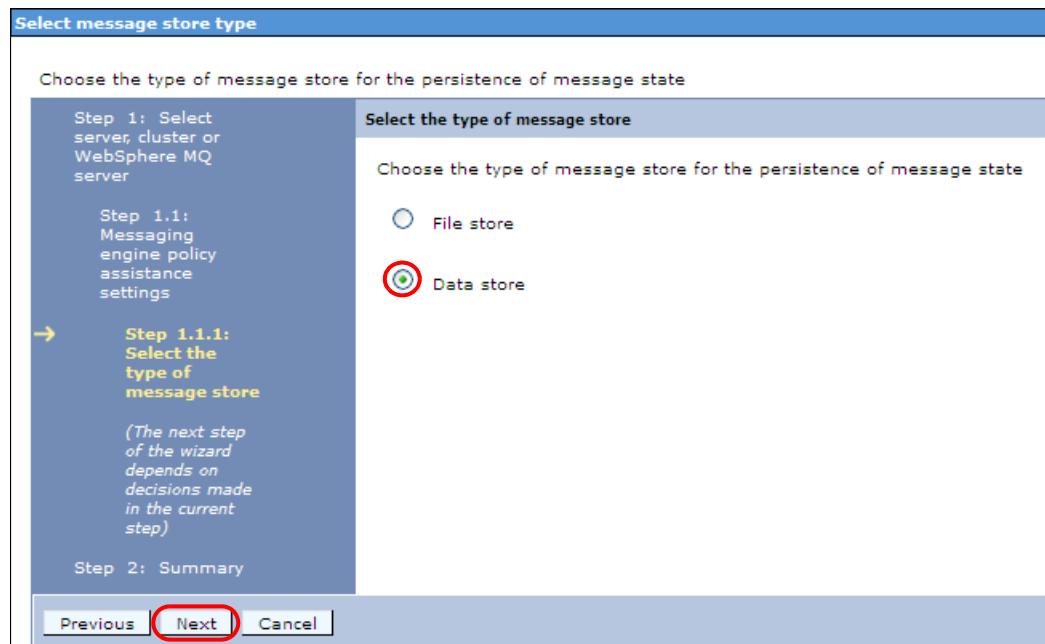


Figure 17-53 Select Data store type of message store

12. Click **Add** to create a new messaging engine, as shown in Figure 17-54.

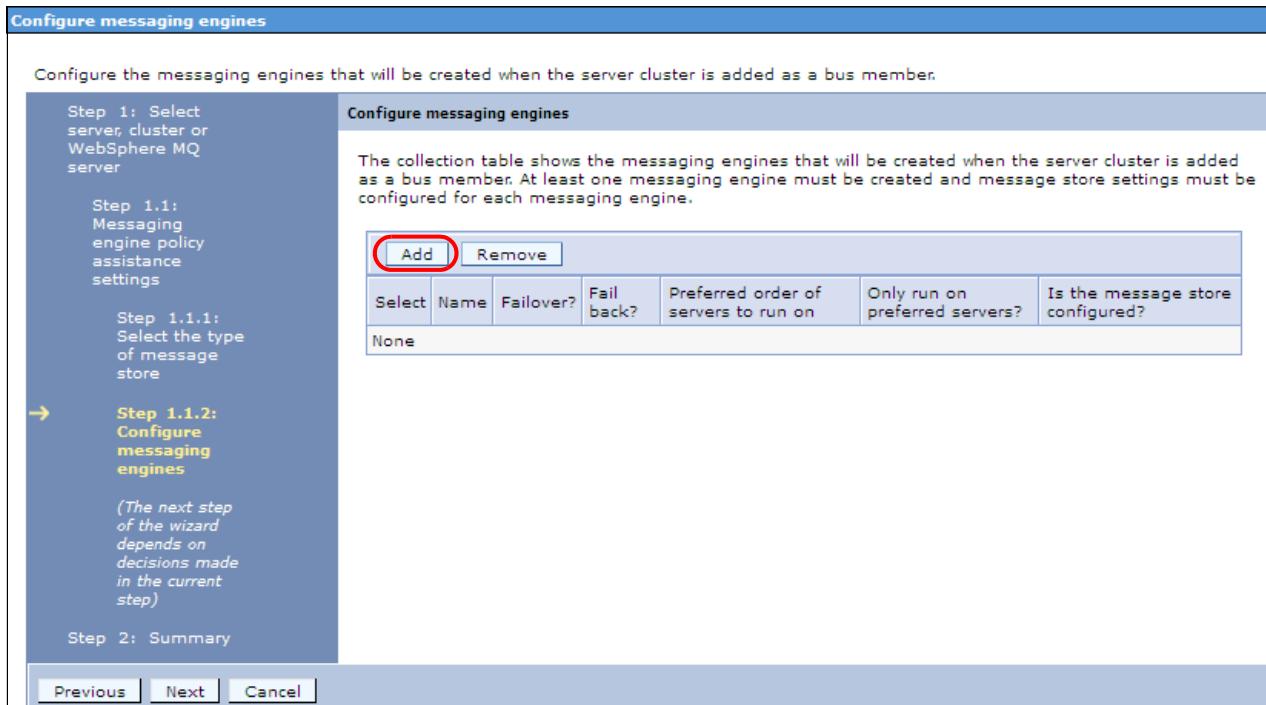


Figure 17-54 Add a new messaging engine

13. We have chosen to enable fallback and failover of the messaging engine by selecting the following options:

- The messaging engine should fail over to other servers in the server cluster.
- The messaging engine should fail back to a preferred server if one is available (as specified in the preferred servers list).

Move suitable servers from the available servers list to the preferred servers list by selecting them and clicking the **Add** button. When you are done, click **Next**, as shown in Figure 17-55.

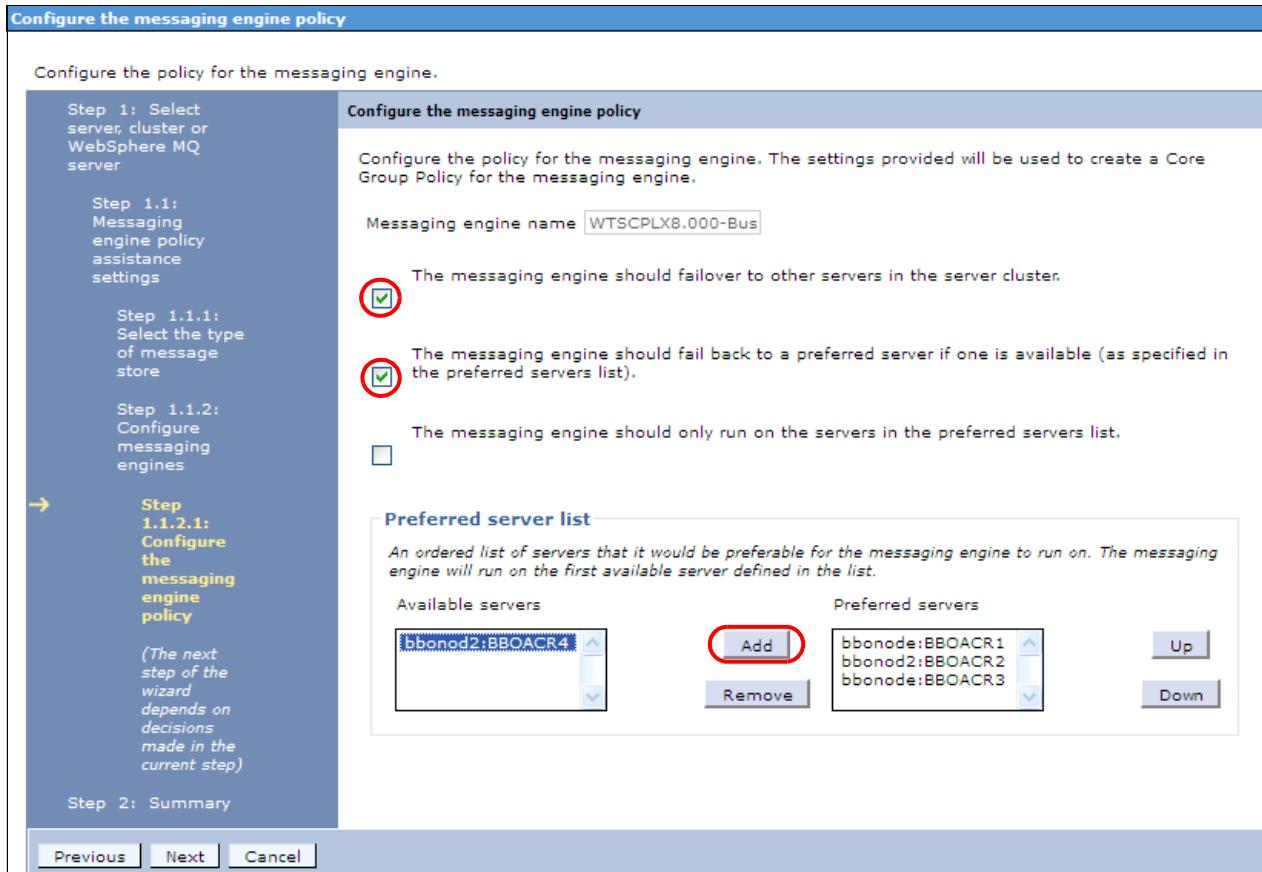


Figure 17-55 Configure the messaging engine policy

14. Enter the Data source JNDI name and Schema name, and select an Authentication alias. Clear **Create tables** and click **Next**, as shown in Figure 17-56.

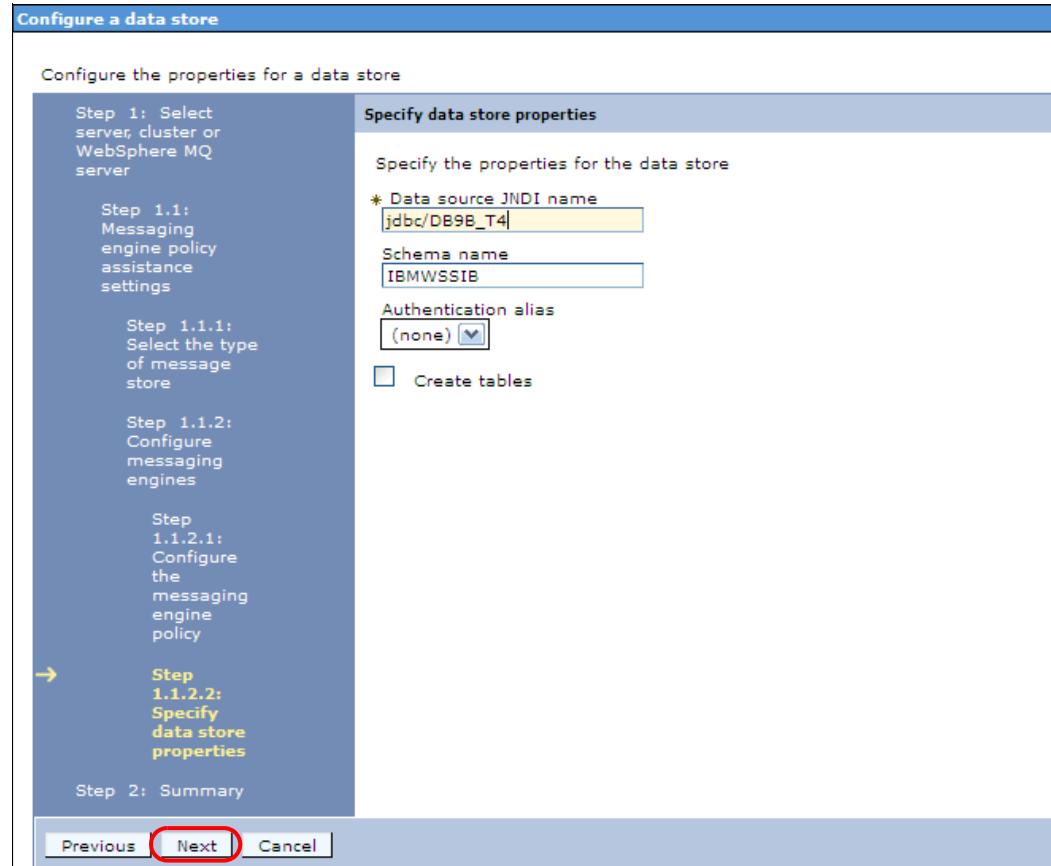


Figure 17-56 Specify data store properties

15. Verify the messaging engine configuration and click **Next**, as shown Figure 17-57.

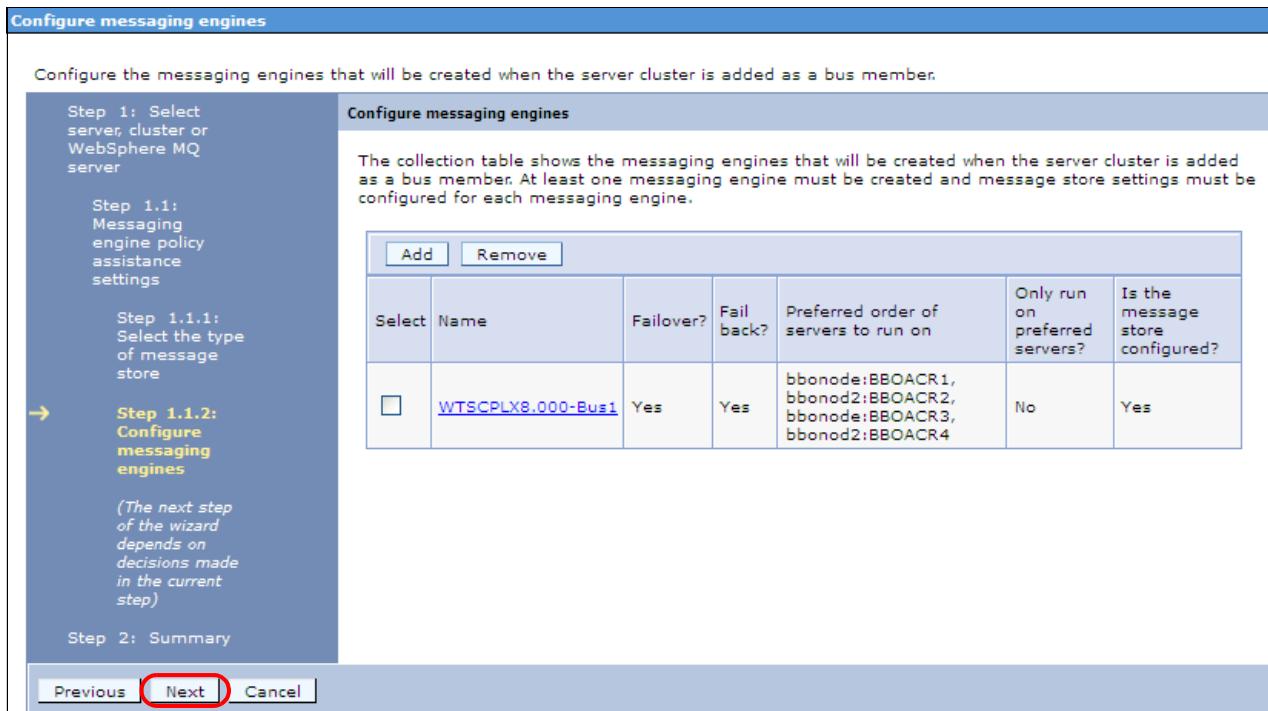


Figure 17-57 Configure messaging engines

16. Click **Finish**, as shown in Figure 17-58.

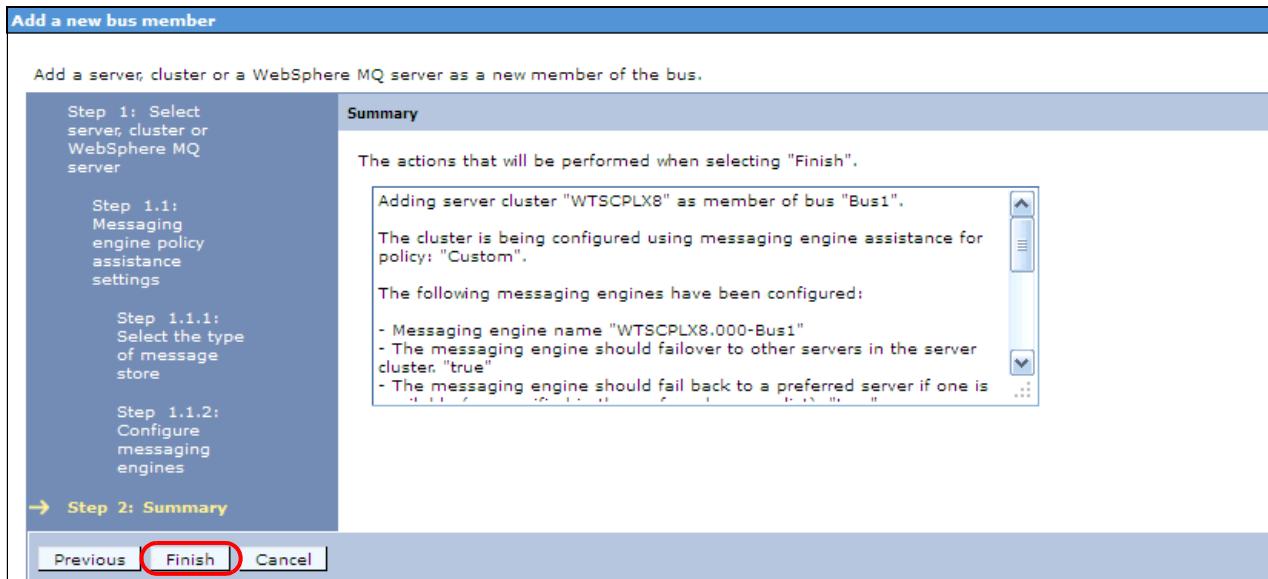


Figure 17-58 Summary window

17. Click **Review**, click **Synchronize changes with nodes**, and click **Save**. Alternatively, click **System administration** → **Nodes** and click **Synchronize** with the appropriate node or nodes selected or enable the **Synchronize changes with Nodes** option in **System administration** → **Console Preferences**.

18. Restart the affected servers by clicking **Servers** → **Server Types** → **WebSphere application servers**.

SIBus note: SIBus data store tables on z/OS have to be created manually using the **sibDDLGenerator.sh** script, as shown in the Example 17-3.

Example 17-3 sibDDLGenerator.sh DB2 for z/OS example

```
 ${WAS_INSTALL_ROOT}/bin/sibDDLGenerator.sh -system db2 -version 9.1 -platform zos  
 -schema schemaname -user userid -create -database databasename -storagegroup  
 storagename -statementend ";"
```

To configure external high availability frameworks for messaging engine with a “No operation” policy, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.zseries.doc/info/zseries/ae/tjt0031_.html

WebSphere Application Server V8 provides a feature to reconnect to a standby gateway MQ queue manager when an active queue manage fails or becomes unavailable. For information about high availability of messaging engines that are connected to WebSphere MQ, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.zseries.doc/info/zseries/ae/cjt0003_.html#cjt0003_

For information about how to create a WebSphere MQ link between SIB and WebSphere MQ queue manager, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.zseries.doc/info/zseries/ae/tjc0002A_.html

17.3.2 Data replication domains

Replication domains are used for replication by the HTTP session manager, dynamic cache service, and stateful session bean failover components. All components that need to share information must be in the same replication domain. Use different replication domains for each type of consumer. You can use same replication domain for HTTP sessions and stateful session beans. Configuring one replication domain in this case ensures that the backup state information is located on the same backup application servers. Cache replication should use separate replication domain.

To create a new replication domain, complete the following steps:

1. Click **Environment** → **Replication domains**.

- Click **New** to create a new replication domain, as shown in Figure 17-59.

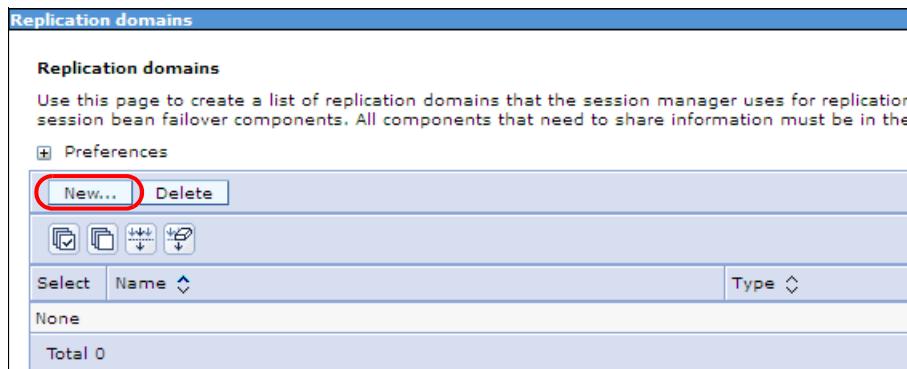


Figure 17-59 New Replication domains

- Specify the Name, Request timeout value, and the Number of replicas, as shown in Figure 17-60. Click **Apply**.

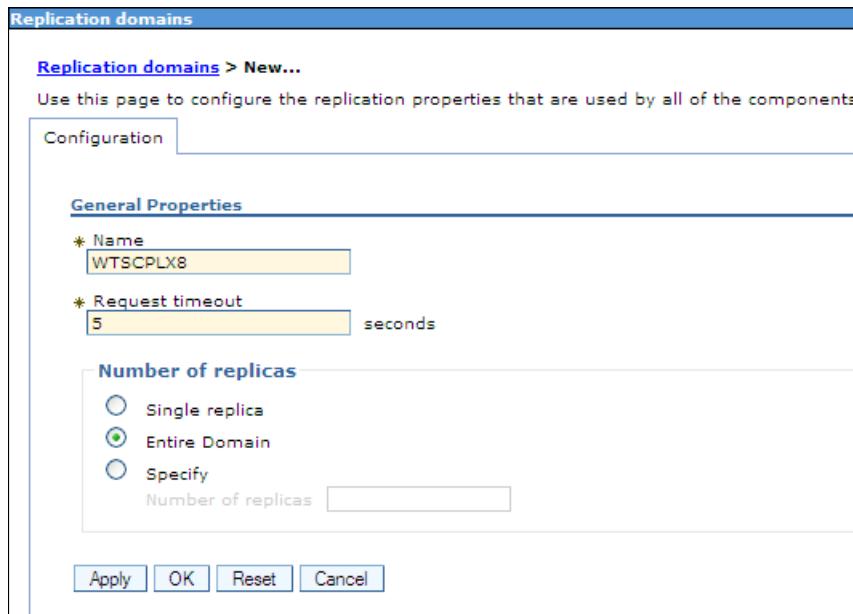


Figure 17-60 Replication domain configuration

- Click **Review**, click **Synchronize changes with nodes**, and click **Save**. Alternatively, click **System administration** → **Nodes** and click **Synchronize** with the appropriate node or nodes selected or enable the **Synchronize changes with Nodes** option in **System administration** → **Console Preferences**.
- Restart the affected application servers by clicking **Servers** → **Server Types** → **WebSphere application servers**.

For more information about data replication, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.zseries.doc/info/zseries/ae/crun_drs_replication.html

17.3.3 Session management replication

For information about how to enable persistent session management using a data store, refer to 25.8.1, “Enabling database persistence” on page 905. If the session persistence database is hosted on local data sharing DB2 for z/OS, the performance is comparable or better to memory-to-memory replication scheme.

To enable memory-to-memory session data replication in the cluster, complete the following steps:

1. Click **Servers** → **Server Types** → **WebSphere application servers**.
2. Select your desired application server, as shown in Figure 17-61.

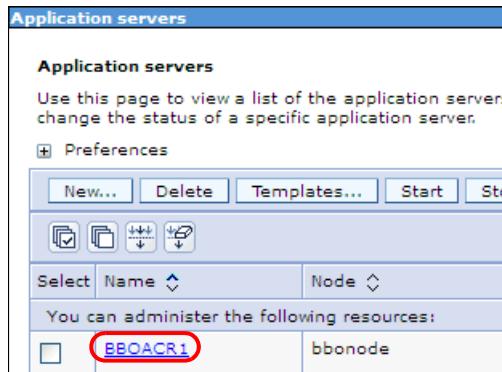


Figure 17-61 Specify the application servers

3. In the Container Settings section, select **Session management**, as shown in Figure 17-62.

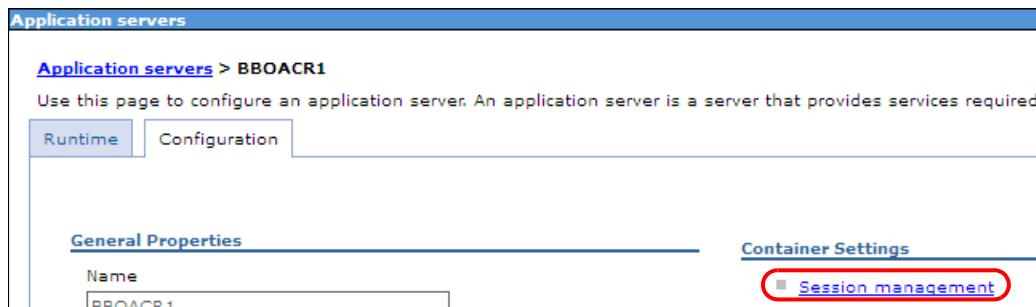


Figure 17-62 Session management

- Under Additional Properties, select **Distributed environment settings**, as shown in Figure 17-63.

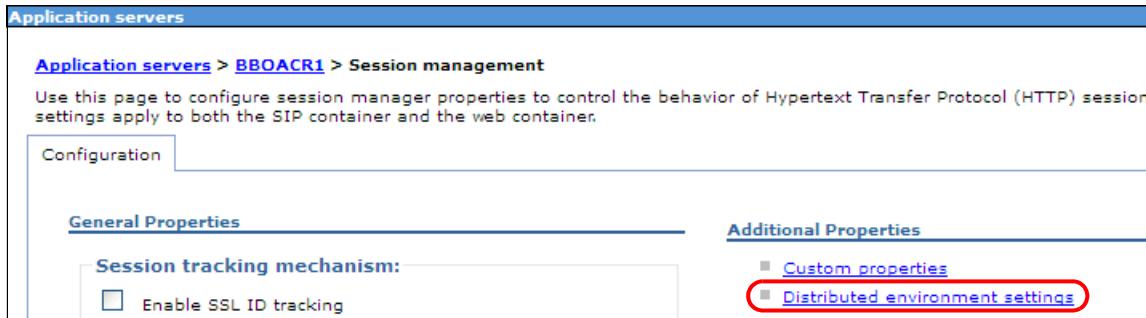


Figure 17-63 Distributed environment settings

- Click **Memory-to-memory replication**, as shown in Figure 17-64.

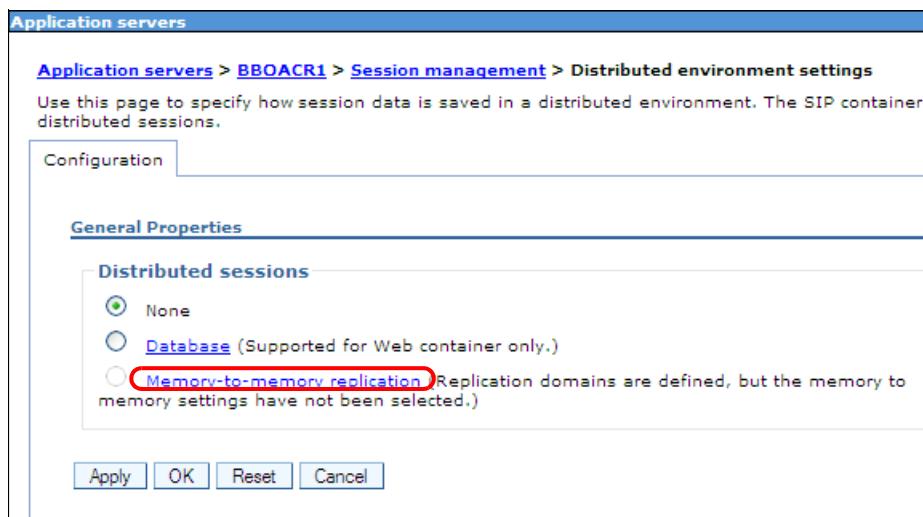


Figure 17-64 Memory-to-memory replication

- Select the desired Replication domain and Replication mode. Click **Apply**, as shown in Figure 17-65.

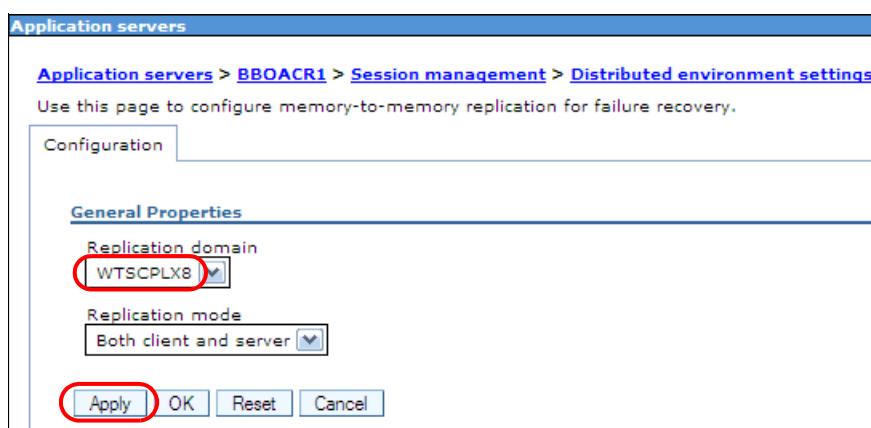


Figure 17-65 Memory-to-memory replication configuration

7. Click **Review**, click **Synchronize changes with nodes**, and click **Save**. Alternatively, click **System administration** → **Nodes** and click **Synchronize** with the appropriate node or nodes selected or enable the **Synchronize changes with Nodes** option in **System administration** → **Console Preferences**.
8. Repeat enablement for every application server in the cluster.
9. Restart the affected application servers by clicking **Servers** → **Server Types** → **WebSphere application servers**.

17.3.4 EJB stateful session bean replication

To enable stateful EJB sessions replication, complete the following steps:

1. Click **Servers** → **Server Types** → **WebSphere application servers**.
2. Select the desired application server, as shown in Figure 17-66.

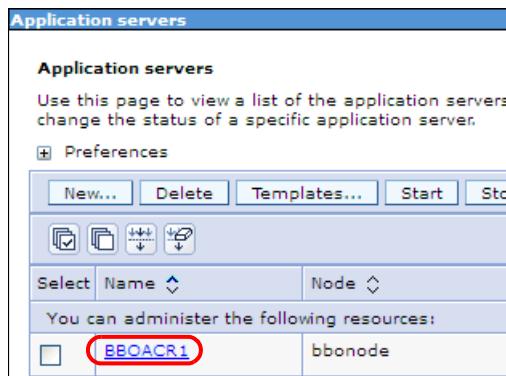


Figure 17-66 Select the application server

3. In the Container Settings section, expand **EJB Container Settings**, and then select **EJB container**, as shown in Figure 17-67.

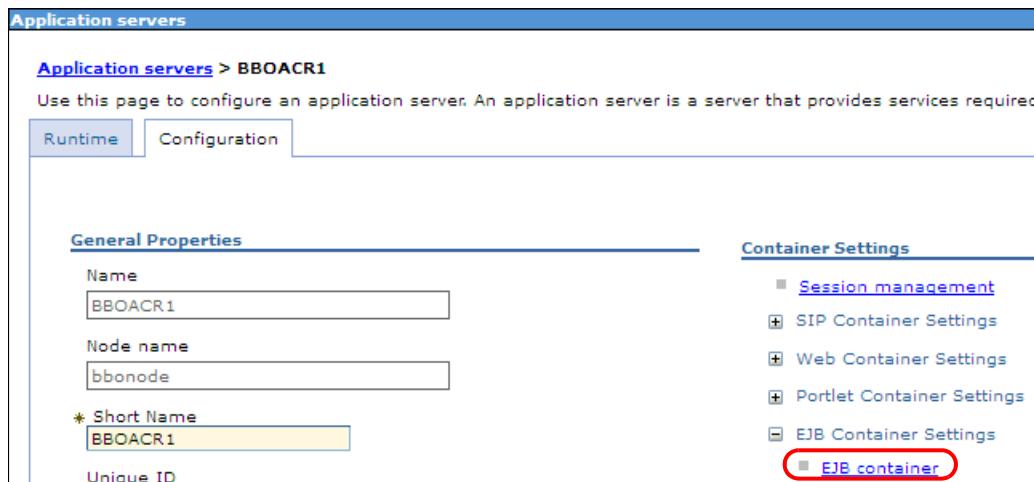


Figure 17-67 EJB container

- Click the **memory-to-memory replication** link, as shown in Figure 17-68.

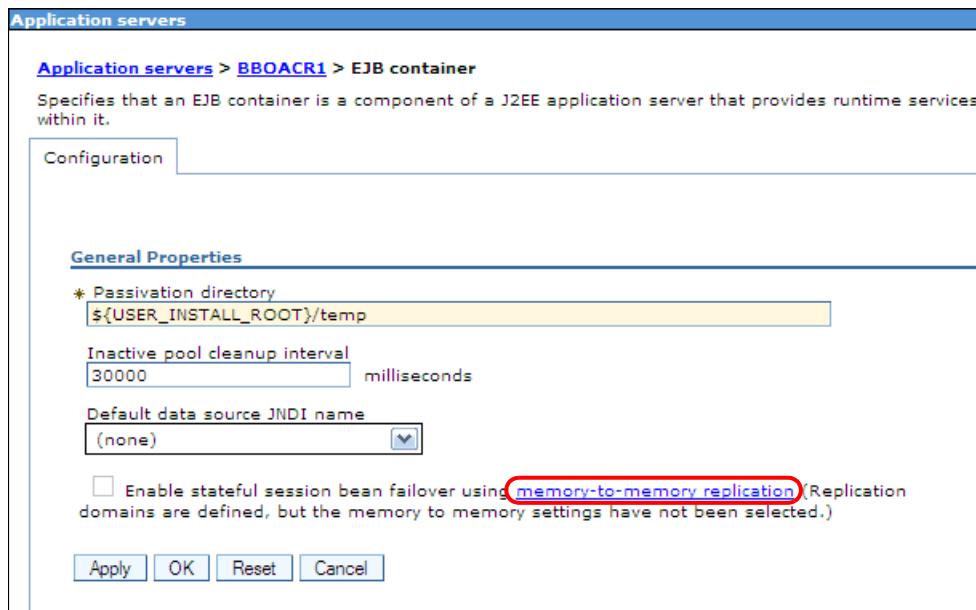


Figure 17-68 Select memory-to-memory link

- Select the desired Replication domain and Replication mode. Click **Apply**, as shown in Figure 17-69.

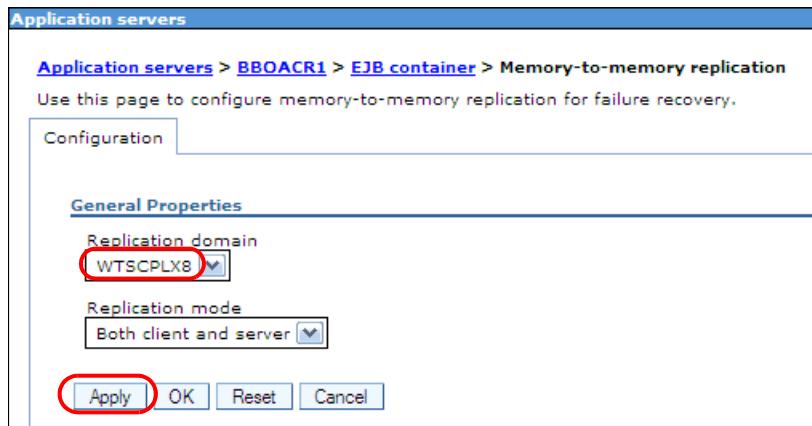


Figure 17-69 Memory-to-memory replication configuration

- Click **Review**, click **Synchronize changes with nodes**, and click **Save**. Alternatively, click **System administration** → **Nodes** and click **Synchronize** with the appropriate node or nodes selected or enable the **Synchronize changes with Nodes** option in **System administration** → **Console Preferences**.

7. In the EJB container window, click **Apply**, as shown in Figure 17-70.

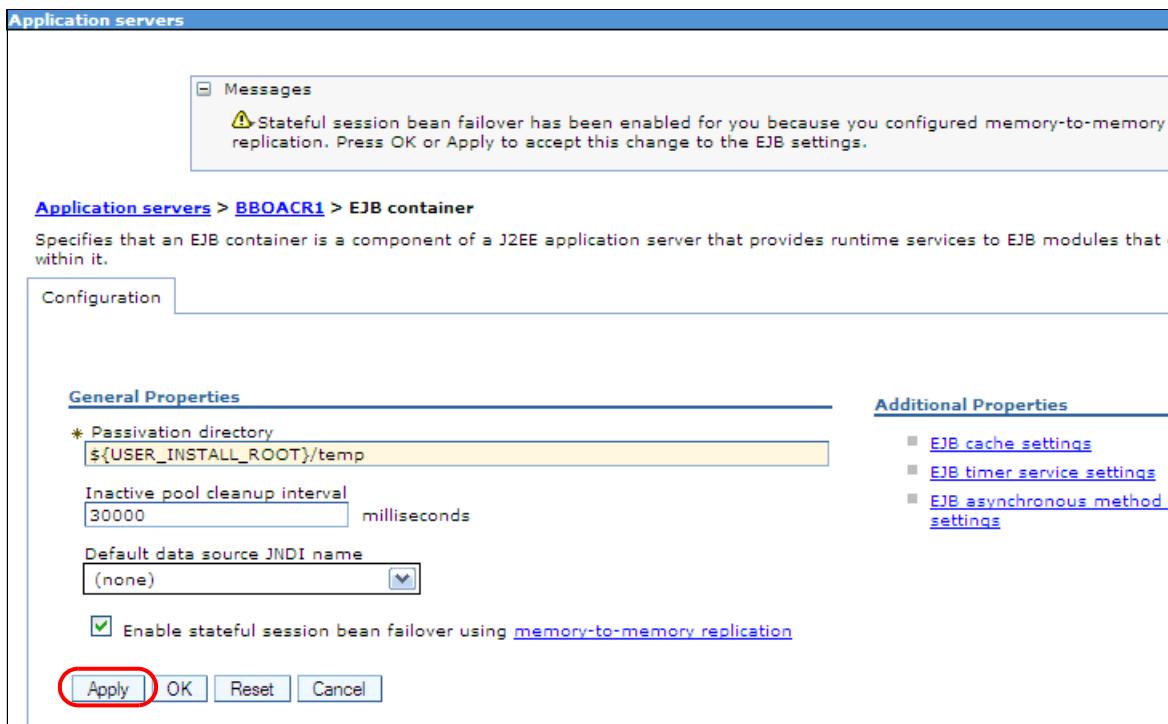


Figure 17-70 EJB container window

8. Click **Review**, click **Synchronize changes with nodes**, and click **Save**. Alternatively, click **System administration** → **Nodes** and click **Synchronize** with the appropriate node or nodes selected or enable the **Synchronize changes with Nodes** option in **System administration** → **Console Preferences**.
9. Repeat enablement for every application server in the cluster.
10. Restart the affected application servers by clicking **Servers** → **Server Types** → **WebSphere application servers**.

Attention: Remember that with failover enabled, your application should never use both a local (EJBLocalObject) and remote (EJBObject) reference to the same stateful session bean instance.

To enable failover of only some stateful session beans, override the global EJB container settings at either the application or EJB module level.

For more information about Stateful session bean failover for the EJB container, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.base.doc/info/aes/ae/cejb_sfsbf.html

To learn how to enable servant failover of EJB container in an unmanaged server, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=%2Fcom.ibm.websphere.nd.multiplatform.doc%2Finfo%2Fae%2Ftejb_sfsbfzos.html

17.3.5 Cache replication

You can make cache elements highly available by using memory-to-memory replication. To enable cache replication in WebSphere Application Server, complete the following steps:

1. Click **Servers** → **Server Types** → **WebSphere application servers**.
2. Select your desired application server, as shown in Figure 17-71.

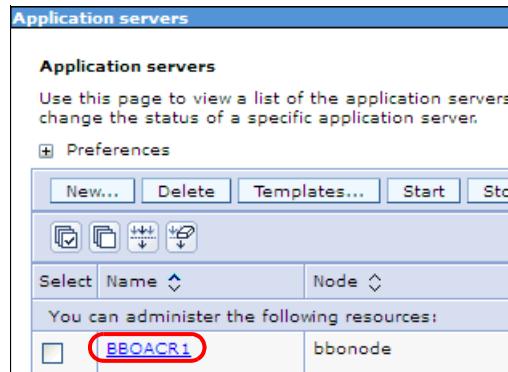


Figure 17-71 Select the application server

3. In the Container Settings section, expand **Container Services**, and then select **Dynamic cache service**, as shown in Figure 17-72.

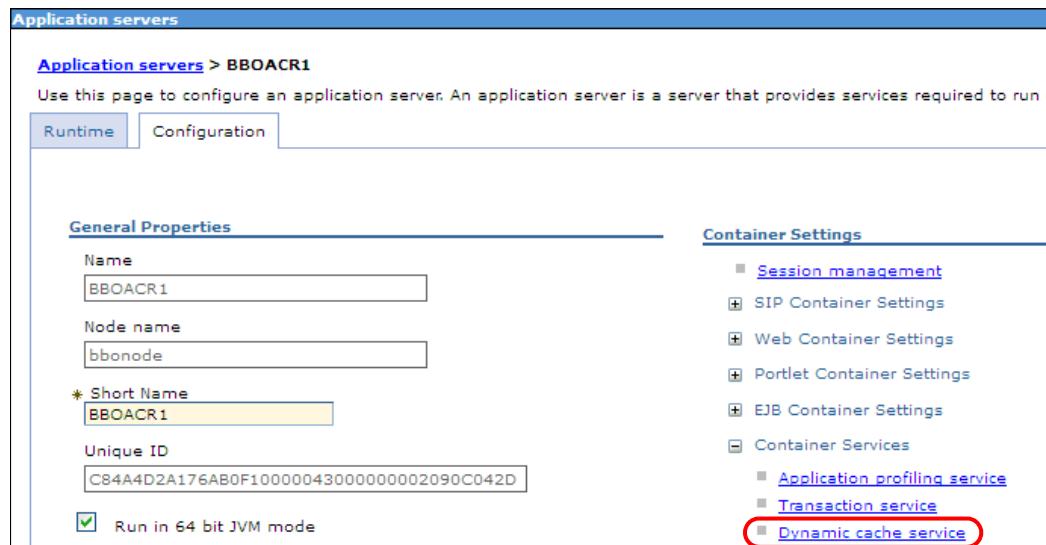


Figure 17-72 Dynamic cache service

4. Select the **Enable cache replication** option, and then select the desired Replication domain and Replication type. Click **Apply**, as shown in Figure 17-73.

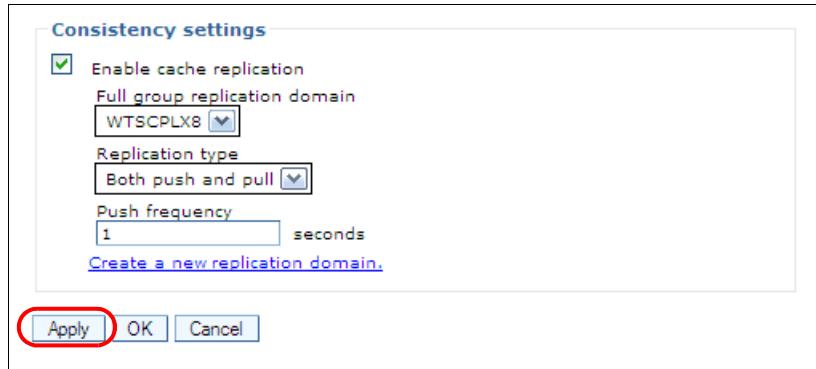


Figure 17-73 Consistency settings

5. Click **Review**, click **Synchronize changes with nodes**, and click **Save**. Alternatively, click **System administration** → **Nodes** and click **Synchronize** with the appropriate node or nodes selected or enable the **Synchronize changes with Nodes** option in **System administration** → **Console Preferences**.
6. Repeat enablement for every application server in the cluster.
7. Restart the affected application servers by clicking **Servers** → **Server Types** → **WebSphere application servers**.

Summary: To summarize the sharing policy, it is important to understand that the push only replication type is good for workloads with a high probability of other clusters handling cache hits. The both push and pull replication type is good for unknown or variable probability of cross-cluster cache hits.

The “Number of replicas” property for any replication domain that is used by the dynamic cache service must be set to **Entire domain**.

To learn how to enable cache replication on a single server in a non-clustered environment, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.zseries.doc/info/zseries/ae/tdyn_cachereplication.html

17.3.6 Resource workload routing

The resource workload routing feature works for JCA J2C connection factories, including WebSphere optimized local adapters and JDBC data sources. It includes failover and subsequent failback from a predefined alternate or backup resource. The same enablement procedure for resource workload routing applies for both JCA and JDBC resources.

To enable resource workload routing for JDBC Data source, complete the following steps:

1. Click **Resources** → **JDBC** → **Data sources**.

2. Select your desired data source, as shown in Figure 17-74.

The screenshot shows the 'Data sources' configuration page. At the top, there is a section titled 'Data sources' with a note about editing settings for a datasource. Below this, a scope is selected: 'Scope: Cell=bbocell, Cluster=WTSCPLX8'. A checked checkbox indicates 'Show scope selection drop-down list with the all scopes option'. A dropdown menu shows 'Cluster=WTSCPLX8'. Under 'Preferences', there are buttons for 'New...', 'Delete', 'Test connection', and 'Manage state...'. Below these are icons for creating, deleting, and managing resources. A table lists resources: 'DB9B_T4' and 'DB9G_T4'. The 'DB9B_T4' row is highlighted with a red circle around its name. The table columns are 'Select', 'Name', 'JNDI name', 'Scope', and 'Provider'. The 'Scope' column for DB9B_T4 shows 'Cluster=WTSCPLX8'. The 'Provider' column for both rows shows 'DB2 Universal JDBC Driver Provider (XA)'.

Figure 17-74 Select data source

3. In the Additional Properties section, select **Connection pool properties**, as shown in Figure 17-75.

The screenshot shows the 'Data sources > DB9B_T4' configuration page. It includes a note about editing datasource settings. Below is a 'Configuration' section with a 'Test connection' button. The 'General Properties' tab is active, showing a 'Scope' field containing 'cells:bbocell:clusters:WTSCPLX8'. The 'Additional Properties' tab is also visible, with a red circle around the 'Connection pool properties' link.

Figure 17-75 Connection pool properties

4. Select the **Connection pool custom properties** link, as shown in Figure 17-76.

The screenshot shows the 'Data sources > DB9B_T4 > Connection pools' configuration page. It includes a note about setting properties for connection management. Below is a 'Configuration' section with a 'General Properties' tab showing a 'Scope' field containing 'cells:bbocell:clusters:WTSCPLX8'. The 'Additional Properties' tab is active, with a red circle around the 'Connection pool custom properties' link.

Figure 17-76 Connection pool custom properties

- Click **New** to add a new custom property, as shown in Figure 17-77.

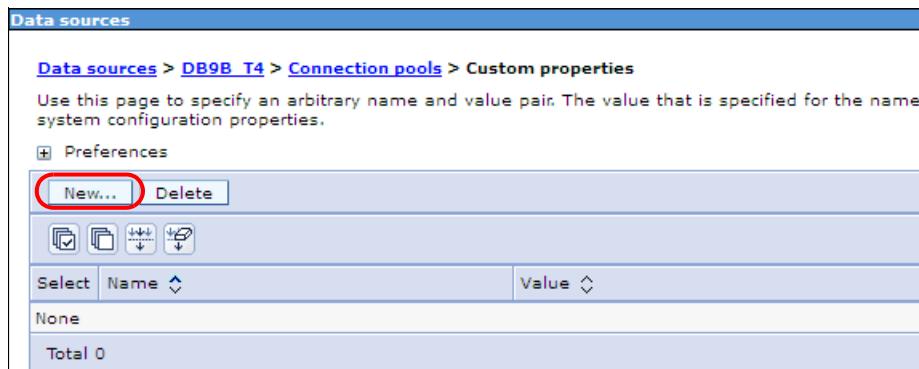


Figure 17-77 Add new custom property

- Enter a name of alternateResourceJNDIName, and enter a value of your mirror failover data source JNDI name. Click **Apply**, as shown in Figure 17-78.

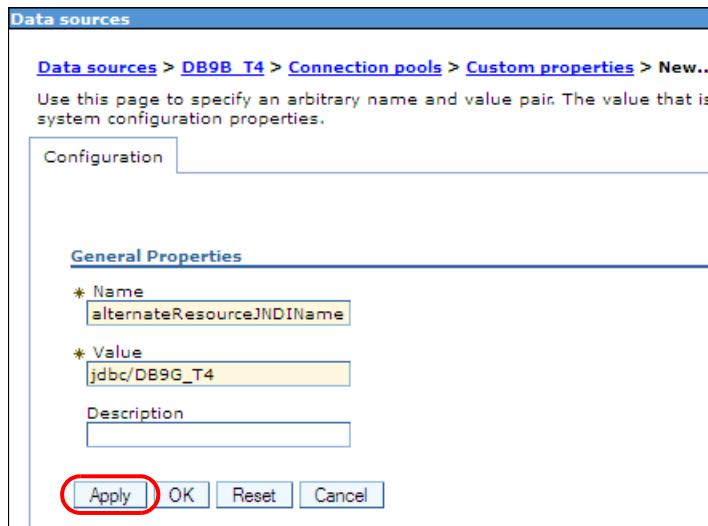


Figure 17-78 New Connection pools custom property configuration

- Click **Review**, click **Synchronize changes with nodes**, and click **Save**. Alternatively, click **System administration** → **Nodes** and click **Synchronize** with the appropriate node or nodes selected or enable the **Synchronize changes with Nodes** option in **System administration** → **Console Preferences**.

8. Repeat steps 5 on page 697 to 7 on page 697 to create additional custom properties to tailor the failover options, as shown in Figure 17-79.

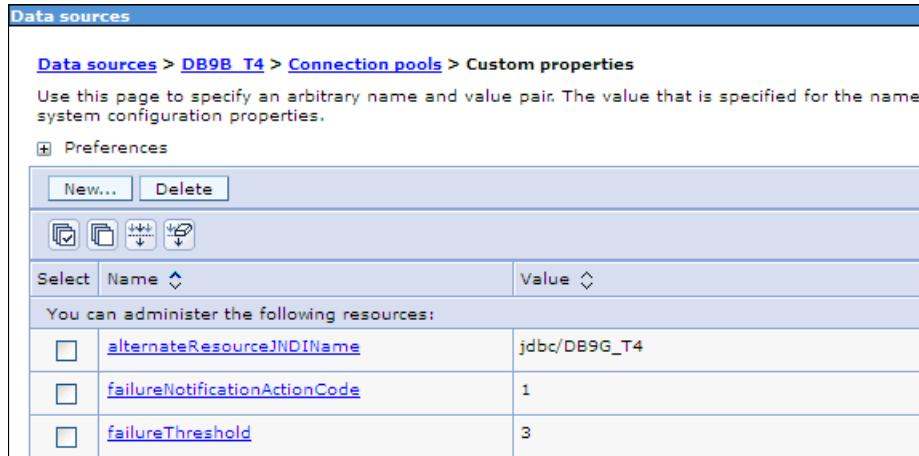


Figure 17-79 Connection pools custom properties window

9. Repeat steps 2 on page 696 to 8 on your mirrored failover data source to enable failback to the primary data source.
10. Restart the affected application servers by clicking **Servers** → **Server Types** → **WebSphere application servers**.

Test: You are encouraged to test the suitability of this feature with your system environment and resources before enabling failover. Mbean support for resource workload routing is provided. The alternate resource configuration should mirror the primary resource configuration.

You can use the commands in the following examples to manually initiate or to enable or disable failover and failback for the specified JNDI name:

- ▶ **F servername,FAILOVER,'jdbc/DB9B_T4'**

The output is shown in Example 17-4.

Example 17-4 Output of the F servername,FAILOVER command

ExtendedMessage: BB000222I: J2CA0690I: The FailOverToAlternateResource operation issued for the resource with a JNDI name of jdbc/DB9B_T4 completed successfully.

- ▶ **F servername,FAILBACK,'jdbc/DB9B_T4'**

The output is shown in Example 17-5.

Example 17-5 Output of the F servername,FAILBACK command

ExtendedMessage: BB000222I: J2CA0690I: The FailBackToPrimaryResource operation issued for the resource with a JNDI name of jdbc/DB9B_T4 completed successfully.

- ▶ **F servername,DISABLEFAILOVER,'jdbc/DB9B_T4'**

The output is shown in Example 17-6.

Example 17-6 Output of the F servername,DISABLEFAILOVER command

ExtendedMessage: BB000222I: J2CA0690I: The DisableResourceFailOver operation issued for the resource with a JNDI name of jdbc/DB9B_T4 completed successfully.

- ▶ **F servername,ENABLEFAILOVER,'jdbc/DB9B_T4'**

The output is shown in Example 17-7.

Example 17-7 Output of the F servername,ENABLEFAILOVER command

ExtendedMessage: BB000222I: J2CA0690I: The EnableResourceFailOver operation issued for the resource with a JNDI name of jdbc/DB9B_T4 completed successfully.

For more information about the resource workload routing feature, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Finfo%2Fzseries%2Fae%2Fcdat_dsfailover.html

17.3.7 High availability application update rollout

To allow for automatic pause or resume of server during application update rollout, complete the following steps:

1. Click **System administration → Node agents**.
2. Select the node agent that is related to the cluster that will be involved in the high availability rollout, as shown in Figure 17-80.

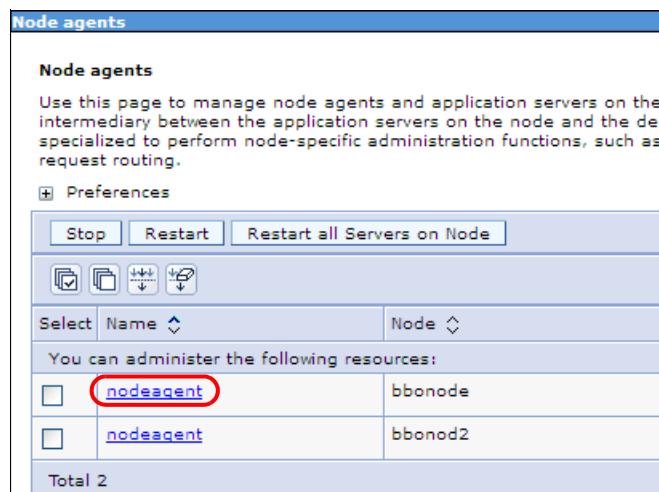


Figure 17-80 Select node agent

3. In the Additional Properties section, select **Administration services**, as shown in Figure 17-81.

Figure 17-81 Administration services

4. In the Additional Properties section, select **Custom properties**, as shown in Figure 17-82.

Figure 17-82 Custom properties

- Click **New** to add a new custom property, as shown in Figure 17-83.



Figure 17-83 New custom property

- Enter a name of com.ibm.websphere.zos.rollout.pauseresume with a value of true. Click **Apply**, as shown in Figure 17-84.

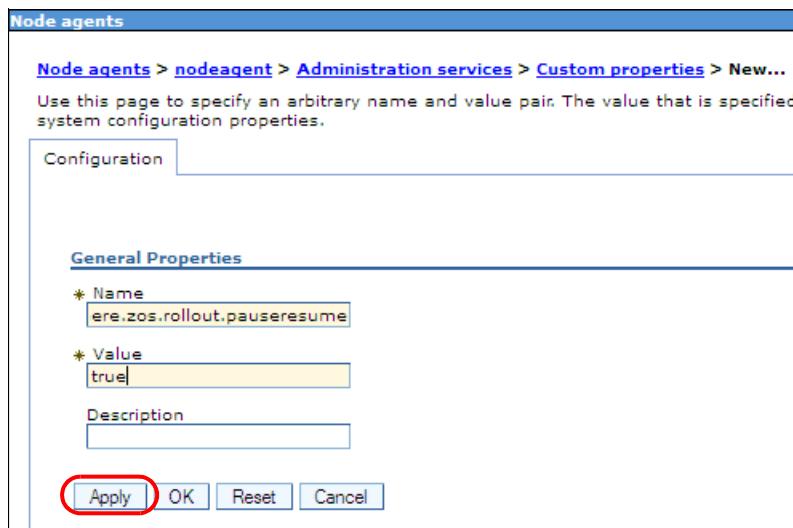


Figure 17-84 New administration service custom property

- Click **Review**, click **Synchronize changes with nodes**, and click **Save**. Alternatively, click **System administration** → **Nodes** and click **Synchronize** with the appropriate node or nodes selected or enable the **Synchronize changes with Nodes** option in **System administration** → **Console Preferences**. Click **Save** and **OK**.

- Repeat steps 5 on page 701 to 7 on page 701, but enter a custom property with a name of com.ibm.websphere.zos.mvsservices.enable and a value of true, so that the Administration services Custom properties view contains the properties shown in Figure 17-85.

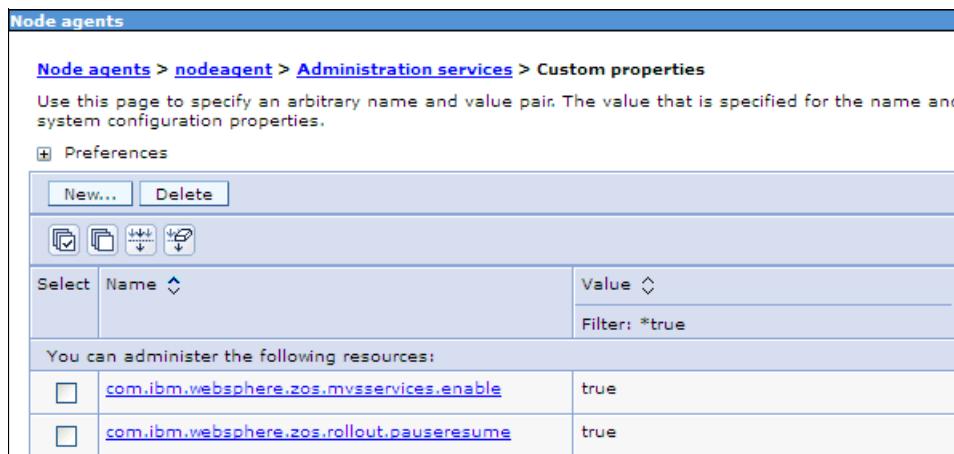


Figure 17-85 Administration services custom properties window

- Repeat steps 2 on page 699 to 8 for all the node agents tied to your cluster.
- Restart the affected node agents.

Manual rollout procedure: For the manual rollout procedure, disable nodeagent **Automatic synchronization** and **Startup synchronization** under **System administration** → **Node agents** → **Node agent name** → **File synchronization service**.

For information about a staged application deployment rollout, go to the following website:
<http://www-03.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP101641>

For information about updating a high availability application manually with a server stop, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v7r0/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Finfo%2Fzseries%2Fae%2Fcontainer_update_ha_apps_zos.html

17.3.8 Additional resources

For more information about high availability, consult the following resources:

- For more information about support for client reroute for applications that use DB2, go to the following website:
http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.zseries.doc/info/zseries/ae/tdat_clientreroute.html
- For more information about support for client affinities for applications that use DB2, go to the following website:
http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.zseries.doc/info/zseries/ae/tdat_clientaffinity.html

- ▶ To learn more about achieving node isolation with intermediate symbolic links, go to the following website:
<http://www-03.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP100396>
- ▶ To learn more about planning to make the deployment manager mobile, go to the following website:
<http://www-03.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP101140>
- ▶ For instructions about how to change cell's host name and system name for a disaster recovery process test, go to the following website:
<http://www-03.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP100792>
- ▶ For information about how to manage large WebSphere Application Server installations, go to the following website:
<http://www.redbooks.ibm.com/abstracts/sg247536.html>
- ▶ For an overview about high availability and a summary on routing mechanisms capabilities, go to the following website:
<http://share.confex.com/share/116/webprogram/Session8381.html>
- ▶ For more information about business resiliency in a Parallel Sysplex environment, go to the following website:
<http://www-03.ibm.com/systems/z/advantages/resiliency/datadriven/network.html>



Monitoring z/OS systems

Being able to measure and monitor system interactions helps IT to provide business continuity. Monitoring capabilities play a key role in successfully managing enterprise systems. In WebSphere Application Server, there are a number of tools that can contribute to the monitoring strategy of an organization and to provide insights into the performance of the application server.

In this chapter, we provide an introduction to these toolsets and talk about the additional tools that are available for WebSphere Application Server on the z/OS platform.

We cover the following topics:

- ▶ Overview
- ▶ Monitoring from the administrative console
- ▶ IBM Tivoli Composite Application Manager for WebSphere Application Server
- ▶ Additional resources for monitoring

18.1 Overview

IT environments are complex, involving many different servers working together to deliver the electronic functions of business. In a single user interaction, it is typical that information can be retrieved from many systems. Consider the simple WebSphere Application Server for z/OS environment in Figure 18-1.

The stars in Figure 18-1 highlight that even a simple web application request can pass through a whole series of dependent servers to successfully complete a request. JEE is a component based architecture, requiring that a request interact and use n number of these components to complete. Monitoring system components and their performance can become complex, yet is critical to understanding the overall performance of an application.

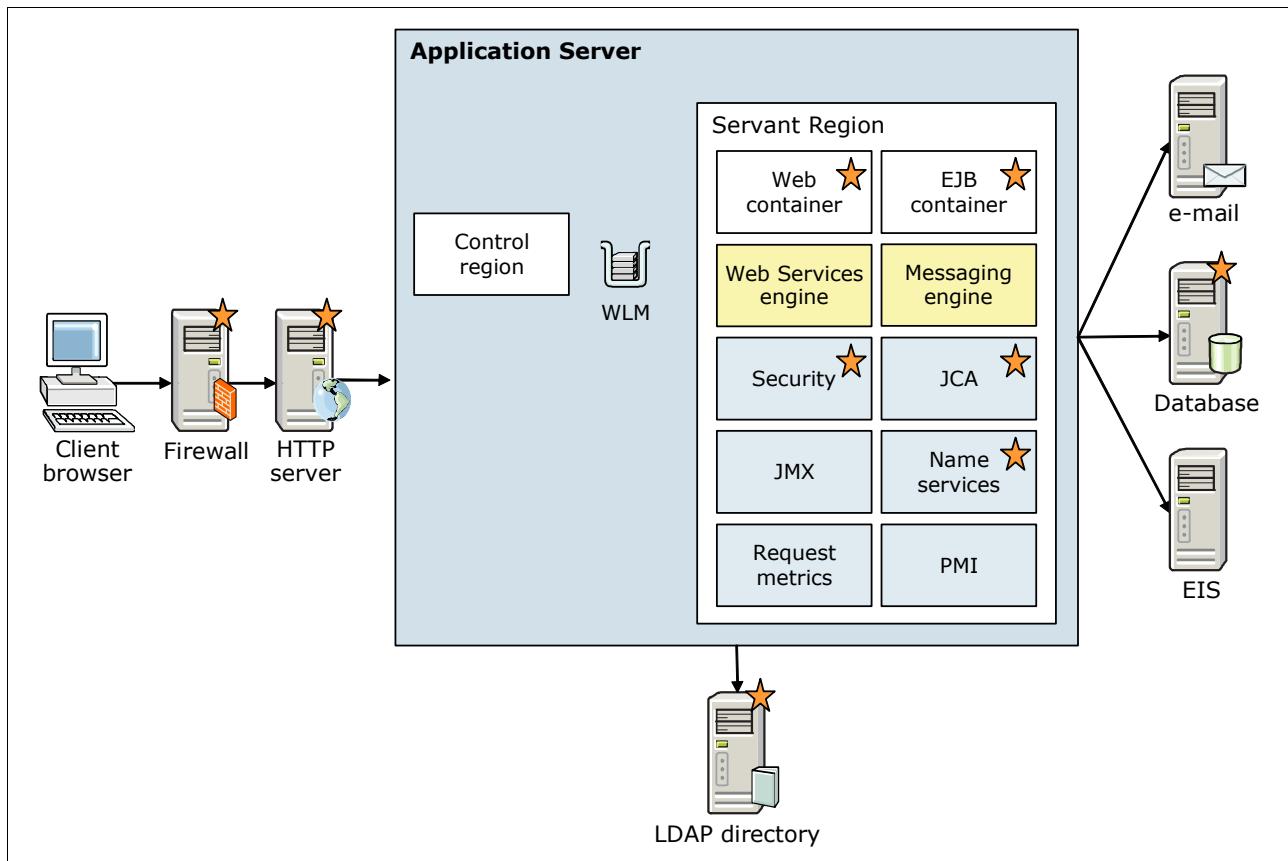


Figure 18-1 Simple web system topology

Monitoring the systems contributes to overall systems management by:

- ▶ Establishing an understanding of the performance baseline and the runtime behaviors that constitute normal operations
- ▶ Measuring performance and identifying poorly performing systems and components
- ▶ Identifying service failures, and assisting in root cause identification

WebSphere Application Server monitoring tools rely primarily on information gathered from two core data infrastructures:

- ▶ Performance Monitoring Infrastructure (PMI), which is a collection of statistical agents scattered throughout the application server that gather statistical data on the performance of the application server components. For more information, go to the following website:
http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Finfo%2Fzseries%2Fae%2Fcprf_pmidata.html
- ▶ Request metrics, which are primarily a set of timing agents that track a request as it navigates the components of the application server. A key differentiation of request metrics is that they are measured at the request level. The focus of a request metric is to record the time spent by individual requests in different components of the application and at the end of the request, provide a record of where the time was spent in the request. For more information, go to the following website:
http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Finfo%2Fzseries%2Fae%2Ftprf_requestmetrics.html

Other information structures on which WebSphere Application Server V8 monitoring tools rely are:

- ▶ WebSphere Application Server for z/OS relies on Work Load Manager (WLM) to collect some of the accounting and performance data.
- ▶ Resource Measurement Facility (RMF) and System Management Facility (SMF) records provide accounting and performance information to WebSphere.

18.2 Monitoring from the administrative console

The PMI and request metrics tools can be used from the administrative console to monitor WebSphere Application Server for z/OS. For instructions, see the following sections in Chapter 15, “Monitoring distributed systems” on page 541:

- ▶ 15.2, “Enabling monitoring infrastructures” on page 545
- ▶ 15.3, “Viewing the monitoring data” on page 556
- ▶ 15.4, “Monitoring scenarios” on page 566

18.3 IBM Tivoli Composite Application Manager for WebSphere Application Server

IBM Tivoli Composite Application Manager (ITCAM) for WebSphere is shipped with WebSphere Application Server V8. Once installed, it is embedded in the application server and can be configured to monitor application performance, providing real-time status information in the WebSphere console. More information about ITCAM for WebSphere Application Server is available at the following website:

http://publib.boulder.ibm.com/infocenter/tivihelp/v24r1/index.jsp?topic=/com.ibm.itcamfad.doc_7101/ecam.html

ITCAM for WebSphere can easily be integrated with ITCAM for Application Diagnostics to provide deep dive diagnostics data, real-time monitoring, and management. ITCAM for Application Diagnostics requires additional licensing. More information is available at the following website:

http://publib.boulder.ibm.com/infocenter/tivihelp/v24r1/index.jsp?topic=/com.ibm.itcamfad.doc_7101/planning_an_installation/overview_of_itcam_for_application_diagnostics.html

18.3.1 Installing the data collector

The first step in enabling ITCAM is to install and configure a data collector for the monitored servers. ITCAM for WebSphere can be installed from WebSphere media using the IBM Installation Manager. For information about the installation, refer to *ITCAM Program Directory*, GI11-8919-01.

18.3.2 Configuring IBM Tivoli Composite Application Manager for WebSphere metrics

After installing ITCAM for WebSphere, complete the following steps:

1. Navigate to the bin installation root directory and run the `createcfg.sh` command (or job CYEZISRT), as shown in Example 18-1, using the following parameters:

-config	Specifies the itcam configuration directory.
-install	Specifies the itcam installation directory.
-owner	Specifies the user and group owners.

Example 18-1 Creating an ITCAM configuration directory

```
/opt/zWebSphere/V8R0/itcamdc/WebSphere/DC/bin # createcfg.sh -config  
/wasv8config/wpcel1/wpdmmnode/ecam -install  
/opt/zWebSphere/V8R0/itcamdc/WebSphere/DC -owner WPADMIN:WPCFG
```

```
+-----+  
| ITCAM 7.2 for WebSphere Application Server      HCYE720 210878 |  
| Create runtime configuration directory          |  
+-----+
```

Create a ITCAM 7.2 for WebSphere Application Server runtime configuration home directory.

You only need to run this script once. All Data Collectors can share the same configuration home.

```
Thu Jul 21 20:24:20 EDT 2011  
uid=0(STC) gid=0(TSO)
```

```
-config /wasv8config/wpcel1/wpdmmnode/ecam  
-install /opt/zWebSphere/V8R0/itcamdc/WebSphere/DC  
-owner WPADMIN:WPCFG
```

For the ITCAM installation directory path you should enter the logical path (symbolic link) instead of the physical path

(canonical path name). Doing this will provide flexibility in cloning and for service migration.

Enter ITCAM 7.2 for WebSphere Application Server installation directory path
"/opt/zWebSphere/V8R0/itcamdc/WebSphere/DC":

Answer: ""

Enter directory name in which to create the ITCAM 7.2 for WebSphere Application Server configuration root
"/wasv8config/wpcell/wpdmnode/ecam":

Answer: ""

You may set a new owner for the configuration home files and directories.

The owner may be entered as "user" or "user:group".
An owner of "none" will bypass setting the owner.

Enter owner for the configuration home "/WPADMIN:WPCFG":

Answer: ""

You may set permissions for the configuration home files and directories.

Permissions are entered in chmod format, which may be either numeric, like "664", or symbolic, like "a+rX,u+w,g+w,o-w".
Permissions specified as "none" will bypass setting permissions.

Enter permissions for the configuration home "/none":

Answer: ""

```
+-----+  
| Summary of chosen configuration parameters |  
+-----+
```

createcfg will create an ITCAM 7.2 for WebSphere Application Server configuration home with the following parameters:

- 1) ITCAM installation path : /opt/zWebSphere/V8R0/itcamdc/WebSphere/DC
- 2) Config home path : /wasv8config/wpcell/wpdmnode/ecam

Existing runtime : No
Owner will be : WPADMIN:WPCFG

Do you wish to proceed with configuration home creation? (y|n|q): y

Answer: "y"

For directory /wasv8config/wpcell/wpdmnode/ecam:

```
Successfully created runtime subdirectory
Successfully created runtime/custom subdirectory
Successfully created bin subdirectory
Successfully created itcamdc symbolic link.
Successfully created toolkit symbolic link.
Successfully created plugins symbolic link.
Successfully created symbolic link for bin/amupdate.sh.
Successfully created symbolic link for bin/cye_collector.sh.
Successfully created symbolic link for bin/reconfig_dc_was.sh.
Successfully created symbolic link for bin/setmode.sh.
Successfully created symbolic link for bin/setupdc.sh.
Successfully created symbolic link for bin/unconfig.sh.
Successfully created symbolic link for bin/configDataCollector.jacl.
Successfully created symbolic link for bin/findServers.jacl.
Successfully created symbolic link for bin/reconfig_dc_was.jacl.
Successfully created symbolic link for bin/unconfigDataCollector.jacl.
Successfully edited setmodecntl
Successfully created itcam.properties file.
Successfully created toolkit_global_custom.properties file.
```

Setting ownership of /wasv8config/wpcell/wpdmnode/ecam to WPADMIN:WPCFG

Configuration home directory created:

```
/wasv8config/wpcell/wpdmnode/ecam
```

```
*** Successful completion of createcfg script. ***
```

2. Navigate to the **wsadmin.sh** command directory, and ensure that you can connect to the WebSphere stand-alone or deployment manager at the SOAP or RMI port.

Note: Ensure that the user who runs the **wsadmin.sh** command has enough region size, as the command starts a Java Virtual Machine and demands some memory. Verify the SIZE parameter in the RACF TSO segment.

3. Navigate to your ITCAM configuration directory, and run the **setupdc.sh** command in prompt mode to configure the data collector on your WebSphere, as shown in Example 18-2:

Note: To run the **setupdc.sh** command, you need at least 512 MB of virtual storage. Check the parameter ASSIZEMAX on the OMVS segment for the user who will run it.

Example 18-2 Configuring the data collector

```
/wasv8config/wpcell/wpdmnode/ecam/bin # setupdc.sh
```

```
+-----+
| ITCAM 7.2 for WebSphere Application Server      HCYE720 211506 |
|                                                               |
|               Data Collector Configuration          |
+-----+
```

```
+-----+
```

```
This script will configure a Data Collector instance for a  
WebSphere Application Server
```

```
Thu Jul 21 21:20:37 EDT 2011  
uid=0(DFS) gid=0(TSO)
```

```
Default local host name detected: WTSC58.ITSO.IBM.COM  
Default local IP address detected: 9.12.4.8  
Local host name used for local node resolution: WTSC58.ITSO.IBM.COM  
Local IP address used for local node resolution: 9.12.4.8
```

```
Enter the path of the WAS user install root `u/WAS80`:  
/wasv8config/wpcell/wpnodea
```

```
Answer: "/wasv8config/wpcell/wpnodea"
```

```
Searching for wsadmin.sh. Please wait...  
Found wsadmin.sh in the following locations:
```

- 1) /wasv8config/wpcell/wpnodea/AppServer/bin/wsadmin.sh
wsadmin.sh WAS home is /wasv8config/wpcell/wpnodea/AppServer
Node wpnodea in cell wpcell
- 2) /wasv8config/wpcell/wpnodea/AppServer/profiles/default/bin/wsadmin.sh
wsadmin.sh WAS home is /wasv8config/wpcell/wpnodea/AppServer
Node wpnodea in cell wpcell

```
Choose a wsadmin.sh under an Application Server node  
containing the Application Server to be configured,  
preferably one under the profiles/default/bin directory.
```

```
Enter the number of the wsadmin.sh to use `2`:
```

```
Answer: "2"
```

```
Local WAS node is wpnodea in cell wpcell  
Using wsadmin:  
/wasv8config/wpcell/wpnodea/AppServer/profiles/default/bin/wsadmin.sh  
Searching for local Application Servers  
This may take some time. Please wait...
```

```
Realm/Cell Name: <default>  
Username: wpadmin  
Password: passw0rd  
WASX7209I: Connected to process "dmgr" on node wpdmnode using SOAP connector;  
The type of process is: DeploymentManager  
WASX7303I: The following options are passed to the scripting environment and  
are  
available as arguments that are stored in the argv variable:  
"/tmp/am_servers5  
0333959"
```

```
#####
# findServers for ITCAM 7.2 HCYE720 210878 Thu Jul 21 21:29:28 EDT 2011 #
#
#####

```

Node: wpnodea

1) wpsr01a

Server "wpsr01a" has been selected
Checking WebSphere Application Server version...
Found WAS Version 8.0.0.0 in package ND

The following runtime directory will be created:

/wasv8config/wpcell/wpdmnode/ecam/runtime/was80.wpnodea.wpsr01a

```
+-----+
|   ITCAM for WebSphere DC Configuration
|
|   The ITCAM for WebSphere data collector can be configured to
|   apply the required PMI settings and enable ITCAM. Otherwise
|   ITCAM will be configured as disabled; it may be enabled at a
|   later time using the WebSphere Integrated Console. Enabling
|   at a later time will require a restart of the server.
+-----+
```

Do you wish to apply the required PMI settings and enable ITCAM? (y|n): y

Answer: "y"

Required PMI settings will be applied and ITCAM will be enabled.

```
+-----+
|   ITCAM for WebSphere DC Configuration
|
|   The ITCAM for WebSphere data collector can be configured to
|   use ITCAM Managing Server for additional deep-dive analysis.
|   The ITCAM Managing Server is installed separately on a UNIX
|   or Windows host.
|
|   See the product documentation for more information.
+-----+
```

Do you wish to configure the Data Collector
to use the ITCAM Managing Server? (y|n):

Answer: ""

JVM Garbage Collection information will not be gathered.

```
+-----+  
| Summary of chosen configuration parameters |  
+-----+
```

Setup will create an ITCAM runtime with the following parameters:

1) wsadmin script :

/wasv8config/wpcell/wpnnodea/AppServer/profiles/default/bin/wsadmin.sh

2) WAS server name : wpsr01a

Cell Name : wpcell
Node Name : wpnodea
WAS Version : 8.0.0.0 ND
WAS Platform : was80

Deployment : Network Deployment
64-bit mode : 64bit

3) Product : ITCAM 7.2 for WebSphere Application Server

Product Home : /wasv8config/wpcell/wpdmnode/ecam

ApplyPMI/
Enable ITCAM : Yes

Managing Server : No

Enter 'y' to continue, item number to respecify, or 'q' to quit: y

Answer: "y"

```
ITCAM configuration for wpsr01a created in  
/wasv8config/wpcell/wpdmnode/ecam/runtime/was80.wpnnodea.wpsr01a  
Configuring Application Server. This may take some time. please wait...  
Realm/Cell Name: <default>  
Username: wpadmin  
Password: passw0rd  
WASX7209I: Connected to process "dmgr" on node wpdmnode using SOAP connector;  
The type of process is: DeploymentManager  
WASX7303I: The following options are passed to the scripting environment and  
are available as arguments that are stored in the argv variable:  
"/tmp/input.properties50333959"
```

```
#####
#  
# configDataCollector Version ITCAM 7.2 211371 Thu Jul 21 21:33:40 EDT 2011 #
#  
#####
response file /tmp/input.properties50333959
```

product=eCAM

```

server.platform=z/OS

server.id=wpsr01a(cells/wpcell/nodes/wpnodea/servers/wpsr01a|server.xml#Server_1184194176402)
server.version=80
server.runtime=${ITCAMDCHOME}/runtime/was80.wpnodea.wpsr01a
server.variable=ITCAMDCHOME=/wasv8config/wpcell/wpdmnode/ecam
server.variable=AM_HOME=/wasv8config/wpcell/wpdmnode/ecam/itcamdc
server.genericJvmArguments=-Xshareclasses:none -Xverify:none
-agentlib:am=${ITCAMDCHOME}/runtime/was80.wpnodea.wpsr01a/

server.genericJvmArguments=-Xbootclasspath/p:${ITCAMDCHOME}/toolkit/lib/bcm-bootstrap.jar:${ITCAMDCHOME}/itcamdc/lib/ppe.probe-bootstrap.jar

server.genericJvmArguments=-Djava.security.policy=${ITCAMDCHOME}/itcamdc/etc/datcollector.policy
server.systemproperty=am.home=/wasv8config/wpcell/wpdmnode/ecam/itcamdc

server.environment=LIBPATH=${ITCAMDCHOME}/runtime/was80.wpnodea.wpsr01a/lib:${ITCAMDCHOME}/toolkit/lib:
server.classpath=
server.environment=NLS PATH=${ITCAMDCHOME}/toolkit/msg/%L/%N.cat
server.genericJvmArguments=
server.systemproperty=TEMAGCCollector.gclog.path=
server.enablePMI=Yes
server.pmiservice.statisticSet=custom
control.systemproperty=ITCAM_DC_ENABLED=true

control.systemproperty=ws.ext.dirs=${ITCAMDCHOME}/itcamdc/lib:${ITCAMDCHOME}/itcamdc/lib/ext:${ITCAMDCHOME}/itcamdc/lib/ext/was:${ITCAMDCHOME}/toolkit/lib:${ITCAMDCHOME}/toolkit/lib/ext

product=eCAM
serverid=wpsr01a(cells/wpcell/nodes/wpnodea/servers/wpsr01a|server.xml#Server_1184194176402)
server=wpsr01a
node=wpnodea
product home=/wasv8config/wpcell/wpdmnode/ecam
product platform=z/OS
Server major version is 80.

Start setting attributes for wpsr01a
Configuring JVM Args
Modify JVM command line arguments
create new variable AM_OLD_JVM_ARGS

Final merged generic JVM arguments:

-Xshareclasses:none
-Xverify:none
-agentlib:am=${ITCAMDCHOME}/runtime/was80.wpnodea.wpsr01a/
-Xbootclasspath/p
:${ITCAMDCHOME}/toolkit/lib/bcm-bootstrap.jar

```

```

:${ITCAMDCHOME}/itcamdc/lib/ppe.probe-bootstrap.jar
-Djava.security.policy=${ITCAMDCHOME}/itcamdc/etc/datacollector.policy

create new variable AM_CONFIG_JVM_ARGS
Configuring System Properties
Create/Modify JVM system properties
newProperty={name am.home} {value /wasv8config/wpcell/wpdmnode/ecam/itcamdc}
new am.home = "/wasv8config/wpcell/wpdmnode/ecam/itcamdc"
create new property am.home = "/wasv8config/wpcell/wpdmnode/ecam/itcamdc"
newProperty={name TEMAGCCollector.gclog.path} {value {}}
new TEMAGCCollector.gclog.path = ""
create new property TEMAGCCollector.gclog.path = ""
Configuring Environment Variables
Create/Modify java process with 2 environment variables
Creating new environment
LIBPATH=${ITCAMDCHOME}/runtime/was80.wpnodea.wpsr01a/lib:${ITCAMDCHOME}/toolkit
/lib:
Creating new environment NLSPATH=${ITCAMDCHOME}/toolkit/msg/%L/%N.cat
Configuring server variable ITCAMDCHOME
create new variable ITCAMDCHOME
Configuring server variable AM_HOME
create new variable AM_HOME
Configuring server region classpath
Configuring Control Region system properties
Create/Modify JVM system properties
newProperty={name ITCAM_DC_ENABLED} {value true}
new ITCAM_DC_ENABLED = "true"
create new property ITCAM_DC_ENABLED = "true"
newProperty={name ws.ext.dirs} {value
${ITCAMDCHOME}/itcamdc/lib:${ITCAMDCHOME}
/itcamdc/lib/ext:${ITCAMDCHOME}/itcamdc/lib/ext/was:${ITCAMDCHOME}/toolkit/lib:
${ITCAMDCHOME}/toolkit/lib/ext}}
new ws.ext.dirs =
"${ITCAMDCHOME}/itcamdc/lib:${ITCAMDCHOME}/itcamdc/lib/ext:${I
TCAMDCHOME}/itcamdc/lib/ext/was:${ITCAMDCHOME}/toolkit/lib:${ITCAMDCHOME}/tool
kit/lib/ext"
create new property ws.ext.dirs =
"${ITCAMDCHOME}/itcamdc/lib:${ITCAMDCHOME}/itcamdc/lib/ext:${ITCAMDCHOME}/itcam
dc/lib/ext/was:${ITCAMDCHOME}/toolkit/lib:${ITCAMDCHOME}/toolkit/lib/ext"
PMI service is already enabled
Configuring PMI Service statistics_set
Disabling SMF data collection
Removing WebSphere variable AM_HOME
server_SMF_server_interval_enabled control region environment property retained
server_SMF_container_interval_enabled control region environment property
retained
server_SMF_interval_length control region environment property retained

Validating state of server wpsr01a on node wpnodea
Validation completed successfully
WAS validation log file =
/wasv8config/wpcell/wpnodea/AppServer/profiles/default/logs/wsadmin.valout

Saving changes to server wpsr01a on node wpnodea
Saving complete

```

```
Synchronizing with node wpnodea
Synchronization completed successfully on wpnodea

Successfully configured data collector for server wpsr01a
wsadmin.sh return code is 0
*INFO* ITCAM plugin gpex.bundle_manager_was.jar copied

Transform by wsc2n.sh in progress
Transform complete, log at
/wasv8config/wpcell/wpdmnode/ecam/runtime/was80.wpnodela.wpsr01a/logs/transform.log

ITCAM 7.2 for WebSphere Application Server setupdc.sh configuration completed
for /wasv8config/wpcell/wpdmnode/ecam/runtime/was80.wpnodela.wpsr01a
```

4. After configuring the data collector, navigate to your heap definitions and increase the maximum heap size field, adding 128 MB to the current value. If no value is set, assume it is 256 MB and set it to 384 MB. In the console, click **Servers** → **Server Types** → **WebSphere application servers** → <your server> → **Server Infrastructure** → **Java and process management** → **Process definition** → **Servant** → **Additional Properties** → **Java Virtual Machine**.
5. Restart the WebSphere Application Server.

Note: More details about the ITCAM configuration for WebSphere Application Server data collector can be found at the following website:

http://publib.boulder.ibm.com/infocenter/tivihelp/v24r1/topic/com.ibm.itcamfad.doc_7101/itcam_ecam_installation_72.pdf

Refer to Chapter 6, “Installing and Configuring ITCAM for WebSphere Application Server on IBM z/OS” in the PDF.

- After data collector configuration is complete, use the WebSphere administration console to navigate to the PMI window for the configured server (Figure 18-2). A new ITCAM for WebSphere Application Server link is now in the Additional Properties section of the window.

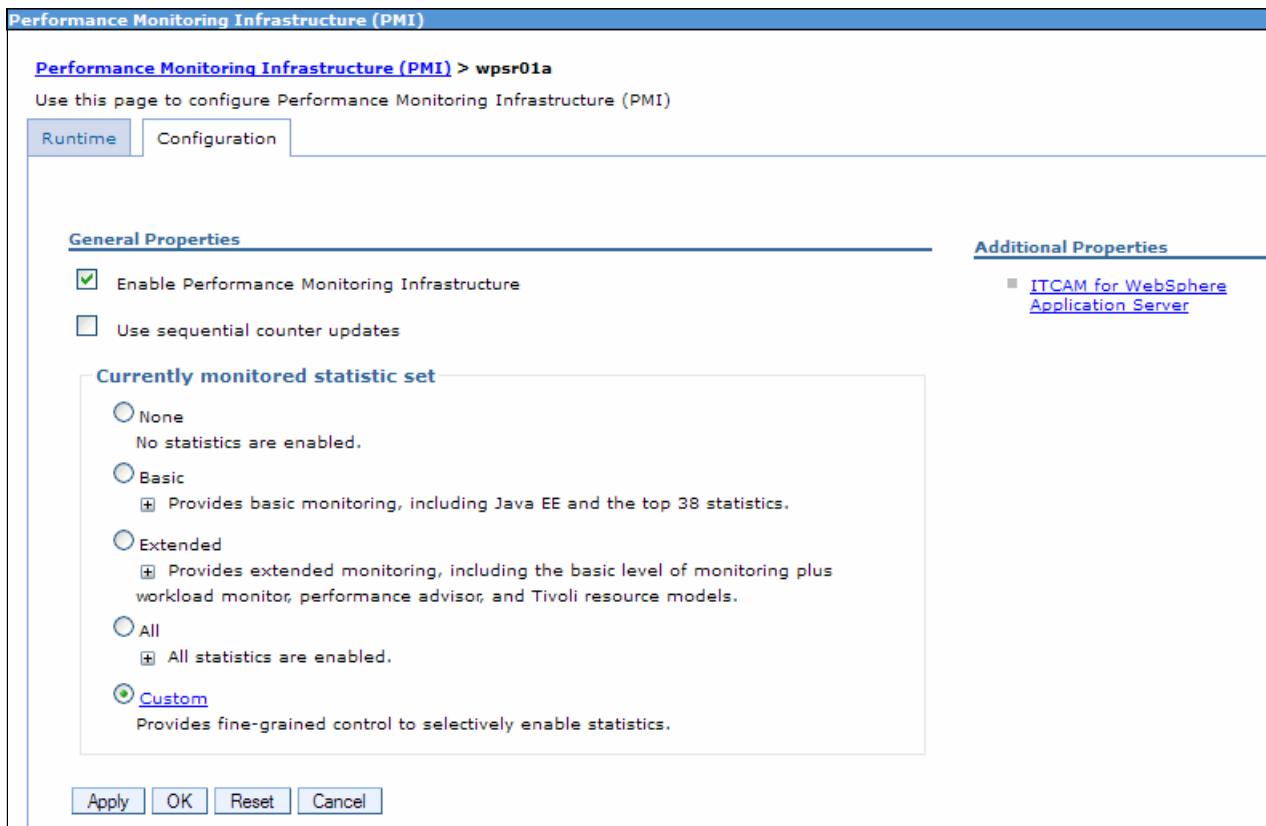


Figure 18-2 PMI configuration with ITCAM for WebSphere Application Server

- Click the **ITCAM for WebSphere Application Server** link to open the window shown in Figure 18-3. This window contains the setting that enables the data collector.

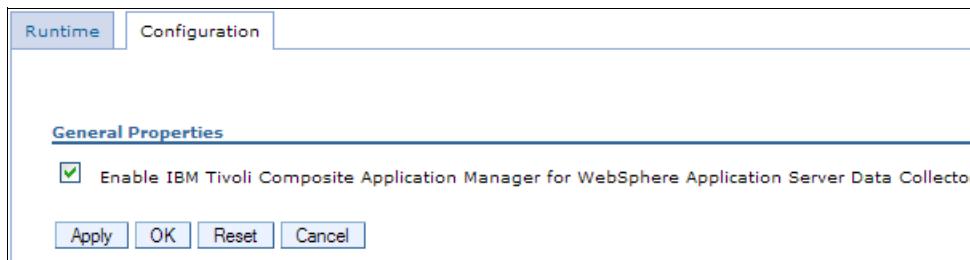


Figure 18-3 ITCAM for WebSphere Application Server window

Note: This setting will only be enabled if you choose to apply the required PMI settings during data collector configuration. If not, enable the ITCAM for WebSphere Application Server data collector and restart the server.

The PMI settings must also be configured at the custom level.

18.3.3 Viewing IBM Tivoli Composite Application Manager for WebSphere data

For information about how to view the data monitored by ITCAM for WebSphere Application Server, refer to 15.5.3, “Viewing IBM Tivoli Composite Application Manager for WebSphere data” on page 582. The process is the same for z/OS and distributed platforms.

18.4 Additional resources for monitoring

In this section, we discuss additional resources to be used while monitoring your WebSphere environment.

18.4.1 IBM Support Assistant

IBM Support Assistant (ISA) is a software workbench provided by IBM to help you diagnose and solve questions about IBM software. It offers several tools as add-ons to help you analyze logs, heap dumps, Java dumps, and so on. For details about the tools and features available for z/OS and how to install and configure them, see *Introducing the IBM Support Assistant for WebSphere Application Server on z/OS* at the following website:

<http://www-03.ibm.com/support/techdocs/atstrmstr.nsf/WebIndex/WP101575>

18.4.2 Verbose garbage collection

Verbose garbage collection was mentioned in 15.4.3, “JVM memory usage” on page 570. Again, assuming there is appropriate disk space, a monitoring strategy should include verbose garbage collection. To enable verbose garbage collection, click **Servers** → **Server Types** → **WebSphere application servers** → <your server> → **Server Infrastructure** → **Java and process management** → **Process definition**, as shown in Figure 18-4:

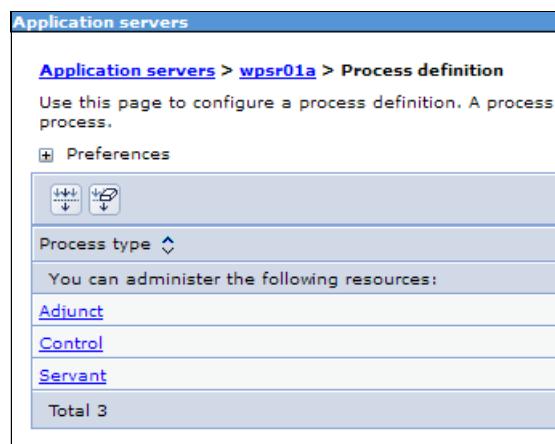


Figure 18-4 Process definition type

You can define the garbage collection for any of the above processes. However, it is more efficient to set it to run in a servant process, in which the application runs and most of the tuning occurs automatically. Click **Servant** → **Additional Properties** → **Java Virtual Machine**, and select the **Verbose garbage collection** box, as shown on Figure 18-5.

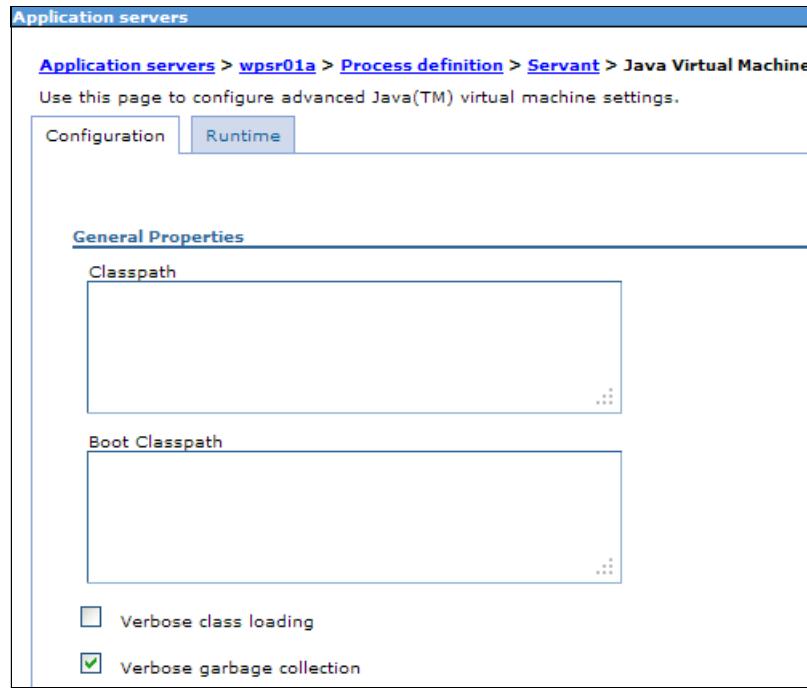


Figure 18-5 Enable verbose garbage collection

Verbose garbage collection can also be enabled using the Runtime tab, as shown in Figure 18-6.

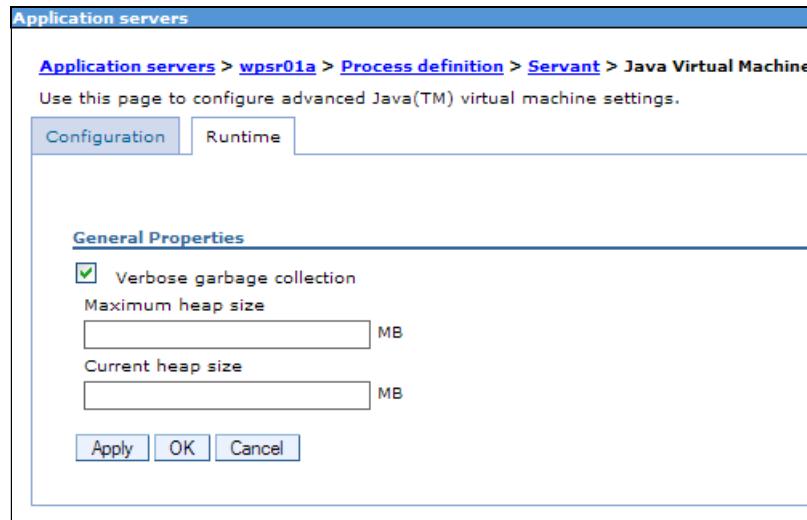


Figure 18-6 Enabling verbose garbage collection at run time

Note: After enabling verbose garbage collection, restart the server to validate the change.

When verbose garbage collector is enabled, a report is written to the output stream each time the garbage collector runs. The report is typically located in SYSOUT DD from servant output. In Version 7, WebSphere used an optimal throughput (**optthrput**) algorithm for garbage collection. Currently, generational concurrent garbage (**gencon**) collection is used, allowing for performance improvements. As **optthrput** uses a large contiguous heap shared by all threads, when garbage collection is invoked, all of this area is scanned. This policy is beneficial for applications that demand optimal throughput, but it has long pause times. Generational concurrent garbage collection divides the heap memory in two pieces:

- ▶ Nursery, for new objects
- ▶ Tenured, for aged objects

So, in this policy, the time spent to scan one of the areas is shorter.

Example 18-3 shows a sample garbage collection entry in SYSOUT.

Example 18-3 Garbage collection entry

```
<exclusive-start id="274" timestamp="2011-07-20T22:44:56.947"
intervalms="34510.096">
    <response-info timems="0.046" idlems="0.046" threads="0"
lastid="0000004808DCB900" lastname="WebSphere non-WLM Dispatch Thread t=007c7c68">
/>
</exclusive-start>
<af-start id="275" totalBytesRequested="40" timestamp="2011-07-20T22:44:56.948"
intervalms="34510.089" />
<cycle-start id="276" type="scavenge" contextid="0"
timestamp="2011-07-20T22:44:56.948" intervalms="34510.087" />
<gc-start id="277" type="scavenge" contextid="276"
timestamp="2011-07-20T22:44:56.948">
    <mem-info id="278" free="157892256" total="261947392" percent="60">
        <mem type="nursery" free="0" total="60620800" percent="0" />
        <mem type="tenure" free="157892256" total="201326592" percent="78" />
            <mem type="soa" free="147826336" total="191260672" percent="77" />
            <mem type="loa" free="10065920" total="10065920" percent="100" />
        </mem>
        <remembered-set count="17359" />
    </mem-info>
</gc-start>
<allocation-stats totalBytes="42822544" >
    <allocated-bytes non-tlh="94144" tlh="42728400" />
    <largest-consumer threadName="WebSphere non-WLM Dispatch Thread t=007c7c68"
threadId="0000004808DCB900" bytes="40718768" />
</allocation-stats>
<gc-op id="279" type="scavenge" timems="105.492" contextid="276"
timestamp="2011-07-20T22:44:57.053">
    <scavenger-info tenureage="6" tiltratio="70" />
    <memory-copied type="nursery" objects="226786" bytes="14839960"
bytesdiscarded="4904" />
    <memory-copied type="tenure" objects="65747" bytes="4425048"
bytesdiscarded="1512" />
    <finalization candidates="976" queued="572" />
    <references type="soft" candidates="4850" cleared="0" queued="0"
dynamicThreshold="29" maxThreshold="32" />
    <references type="weak" candidates="567" cleared="45" queued="16" />
    <references type="phantom" candidates="2" cleared="0" queued="0" />
</gc-op>
```

```

<gc-end id="280" type="scavenge" contextid="276" durationms="105.833"
timestamp="2011-07-20T22:44:57.053">
  <mem-info id="281" free="200176936" total="263258112" percent="76">
    <mem type="nursery" free="46784264" total="61931520" percent="75" />
    <mem type="tenure" free="153392672" total="201326592" percent="76" />
      <mem type="soa" free="143326752" total="191260672" percent="74" />
      <mem type="loa" free="10065920" total="10065920" percent="100" />
    </mem>
    <pending-finalizers system="506" default="66" reference="16" classloader="0" />
    <remembered-set count="13724" />
  </mem-info>
</gc-end>
<cycle-end id="282" type="scavenge" contextid="276"
timestamp="2011-07-20T22:44:57.054" />
<allocation-satisfied id="283" threadId="0000004808DCB900" bytesRequested="40" />
<af-end id="284" timestamp="2011-07-20T22:44:57.054" />
<exclusive-end id="285" timestamp="2011-07-20T22:44:57.054" durationms="106.246" />

```

These and other tools that make garbage collection analysis easier are located on the IBM Support Assistant website at:

<http://www-01.ibm.com/software/support/isa/download.html>

For more information about Java memory management, refer to the Java Information Center at the following website:

http://publib.boulder.ibm.com/infocenter/javasdk/v6r0/index.jsp?topic=%2Fcom.ibm.java.doc.diagnostics.60%2Fdiag%2Funderstanding%2Fmemory_management.html

18.4.3 Java dump and core files

Java core files present, in essence, a picture of what is occurring inside the Java Virtual Machine. These files are helpful for analyzing situations, such as when CPU utilization is nearing 100%, when threads are hanging, or when performance is slow.

Java dump and system dump files are a picture of the objects that were in Java Virtual Machine memory. These files are helpful in diagnosing memory-related problems, such as memory leaks.

Both kinds of files can be generated at the WebSphere console as follows:

1. Click **Troubleshooting** → **Java dumps and cores**.
2. Select the desired server.

3. Click one of the available buttons, as shown in Figure 18-7:

- a. **Heap dump**
- b. **Java core**
- c. **System dump**

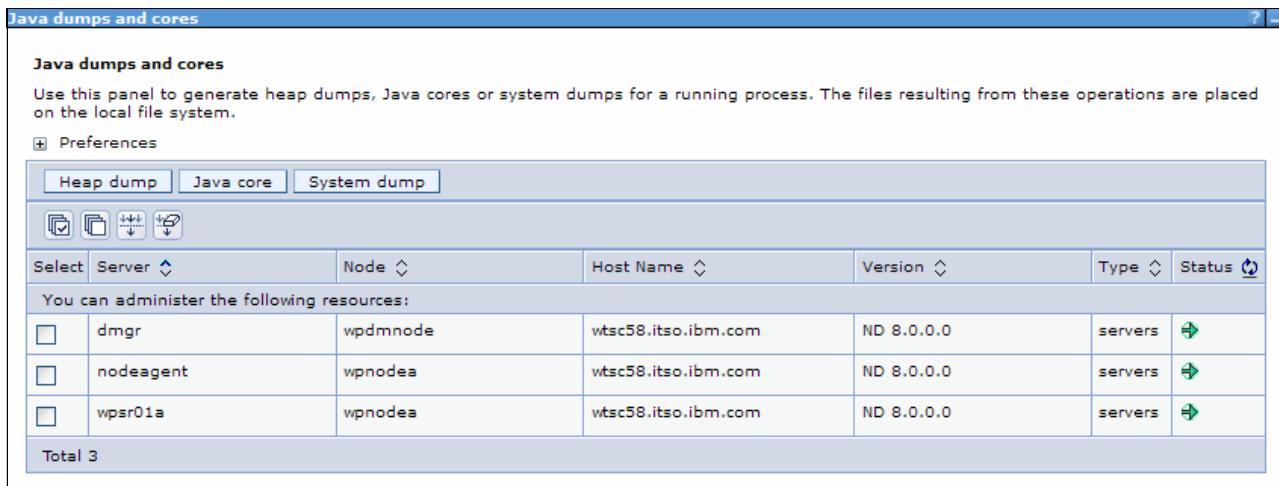


Figure 18-7 Core and dump generation

Check the joblog for information about the files that were generated.

For more information about javadumps, refer to the Java Information Center at the following website:

<http://publib.boulder.ibm.com/infocenter/javasdk/v6r0/index.jsp?topic=%2Fcom.ibm.java.doc.diagnostics.60%2Fdiag%2Ftools%2Fjavadump.html>

For more information about heapdumps, refer to the Java Information Center at the following website:

<http://publib.boulder.ibm.com/infocenter/javasdk/v6r0/index.jsp?topic=%2Fcom.ibm.java.doc.diagnostics.60%2Fdiag%2Ftools%2Fheapdump.html>

For more information about system dumps, refer to the Java Information Center at the following website:

http://publib.boulder.ibm.com/infocenter/javasdk/v6r0/index.jsp?topic=%2Fcom.ibm.java.doc.diagnostics.60%2Fdiag%2Ftools%2Fdump_viewer.html

18.4.4 Basic logging

WebSphere Application Server V8 provides many logging options. The most significant environment incidents are logged automatically. Basic logging provides information in the Job Entry SubSystem (JES) spool, which can be seen in SYSOUT and SYSPRINT cards from process **sysout**. The IBM Tivoli Log Analyzer cannot be used to analyze basic logs in distributed platforms. Different from basic logs in distributed platforms, the IBM Tivoli Log Analyzer cannot be used to analyze them.

18.4.5 Advanced logging

In WebSphere Application Server V8, an alternative to the basic log and trace facility is offered, called High Performance Extensible Logging (HPEL). It provides three repositories:

- ▶ Log data repository: A storage facility for log records, typically information stored in `SystemOut.log`, `SystemErr.log`, or `java.util.logging` at level detail or higher.
- ▶ Trace data repository: A storage facility for trace records, typically information written to `java.util.logging` below level detail.
- ▶ Text log: Plain text file for log and trace records. Provided for convenience.

Note: All data that is written to the log and trace repositories is parsed and formatted to be stored in the text log file. For this reason, consider disabling the log file as soon as possible to enhance server performance.

Figure 18-8 depicts the three HPEL repositories.

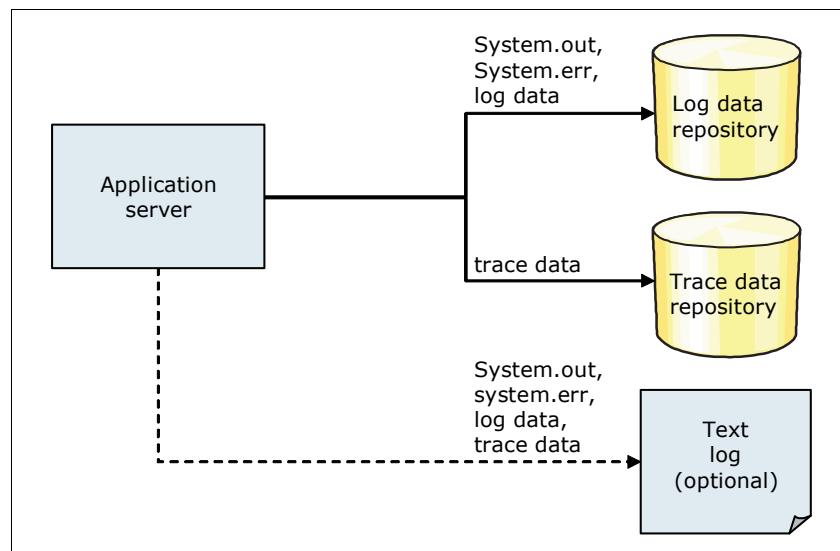


Figure 18-8 HPEL repositories

Now the data is stored in proprietary binary format, rather than text format as in basic logging. (The only exception is for the text log repository.) In this way, the following benefits are achieved:

- ▶ There is no more text parsing.
- ▶ More data is available due to the fact that truncation does not need to be done.
- ▶ Data is not formatted unless necessary.
- ▶ There is no need to clear log files before server start, for example, to diagnosis a problem.
- ▶ Trace speed is improved and more data can be available (it has half of the impact of basic tracing).
- ▶ It is a common solution between z/OS and distributed platform.
- ▶ Applications running with HPEL run faster than with basic logging.

To read the log and trace records in this new format, a new command called **logViewer.sh** was introduced. It reads the data from repositories, formats it, and displays it to the administrator, as shown in Figure 18-9.

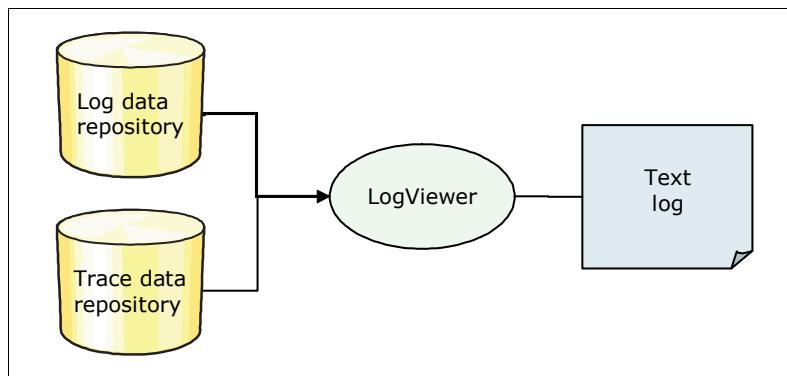


Figure 18-9 *logViewer.sh* command

To activate HPEL logging and tracing, in the administrative console, click **Troubleshooting** → **Logs and trace** → <your_server> → **Change log and trace mode** → **Switch to HPEL Mode** (button), as shown in Figure 18-10.

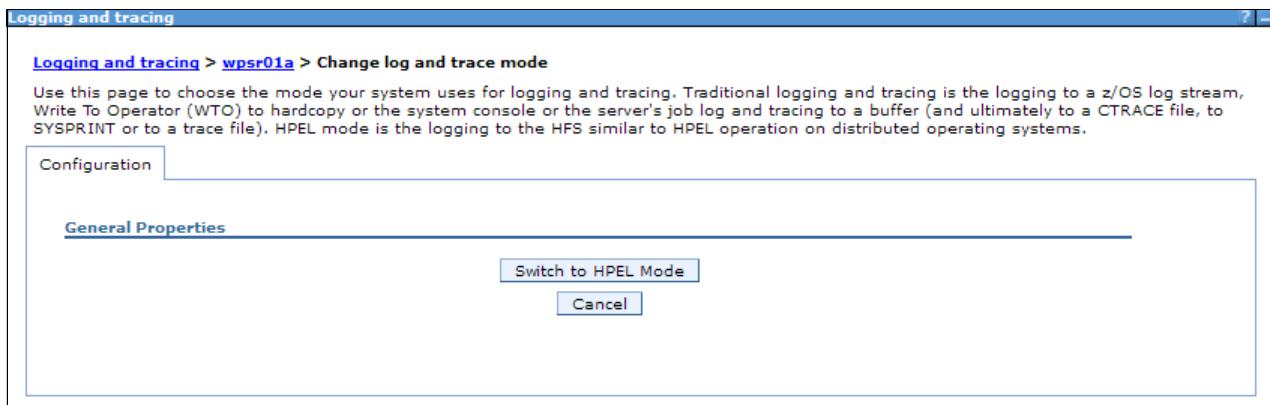


Figure 18-10 *Enabling HPEL*

Note: A server restart is needed after enabling HPEL logging and tracing.

Viewing data from HPEL repositories

After configuring HPEL, the data can be viewed using the **logViewer.sh** command from the bin subdirectory of the profile, as shown in Example 18-4.

Example 18-4 *logViewer.sh* output

Using /wasv8config/wpc11/wpnnodea/AppServer/profiles/default/logs/wpsr01a as repository directory.

Instance ID	Start Date
1311360474568	07/22/11
14:47:54.568 EDT	
1311360474568/000001B400000003-WPSR01AS_STC05860	07/22/11
14:48:52.526 EDT	

As a single z/OS server is composed of several Java Virtual Machines (controller, servant, and adjunct); if the instance is not specified, use `logViewer.sh` to list the available instances instead of listing the logs and traces, as in distributed environments. This output is the same as that obtained from the `logViewer.sh -listInstances` command. In Example 18-4 on page 724, the first instance listed is the controller and the other is the servant.

After enabling HPEL, the log format for WebSphere z/OS is similar to that used for a distributed platform, making it easier for administrators from distributed platforms to analyze issues on the z/OS. In Example 18-5, we see the result of the `logViewer.sh -instance 1311360474568` command (control process).

Example 18-5 WebSphere control log

```
Using /wasv8config/wpcell/wpnodea/AppServer/profiles/default/logs/wpsr01a as repository
directory.
***** Start Display Current Environment *****
WebSphere Platform 8.0.0.0 [ND 8.0.0.0 n1118.03] running with process name WPSR01A and process
id 0000019800000005
Host Operating System is z/OS, version 01.12.00
Java version = 1.6.0, Java Compiler = j9jit26, Java VM name = IBM J9 VM
was.install.root = /wasv8config/wpcell/wpnodea/AppServer
user.install.root = /wasv8config/wpcell/wpnodea/AppServer/profiles/default
Java Home = /wasv8config/wpcell/wpnodea/AppServer/java64
ws.ext.dirs =
/wasv8config/wpcell/wpnodea/AppServer/java64/lib:/wasv8config/wpcell/wpnodea/AppServer/classes:/
wasv8config/wpcell/wpnodea/AppServer/lib:/wasv8config/wpcell/wpnodea/AppServer/installedChannels
:/wasv8config/wpcell/wpnodea/AppServer/lib/ext:/wasv8config/wpcell/wpnodea/AppServer/web/help:/w
asv8config/wpcell/wpnodea/AppServer/deploytool/itp/plugins/com.ibm.etools.ejbdeploy/runtime:/was
v8config/wpcell/wpnodea/AppServer/java64/jre/lib:/wasv8config/wpcell/wpdmnode/ecam/itcamdc/lib:/
wasv8config/wpcell/wpdmnode/ecam/itcamdc/lib/ext:/wasv8config/wpcell/wpdmnode/ecam/itcamdc/lib/e
xt/was:/wasv8config/wpcell/wpdmnode/ecam/toolkit/lib:/wasv8config/wpcell/wpdmnode/ecam/toolkit/l
ib/ext
Classpath =
/wasv8config/wpcell/wpnodea/AppServer/profiles/default/properties:/wasv8config/wpcell/wpnodea/Ap
pServer/properties:/wasv8config/wpcell/wpnodea/AppServer/lib/bootstrap.jar:/wasv8config/wpcell/w
pnodea/AppServer/lib/bootstrapws390.jar:/wasv8config/wpcell/wpnodea/AppServer/lib/lmproxy.jar:/w
asv8config/wpcell/wpnodea/AppServer/lib/startup.jar:/wasv8config/wpcell/wpnodea/AppServer/java64
/lib/tools.jar
Java Library path =
/wasv8config/wpcell/wpnodea/AppServer/patches:/wasv8config/wpcell/wpnodea/AppServer/lib/s390-com
mon:/wasv8config/wpcell/wpnodea/AppServer/lib/s390x-64:/wasv8config/wpcell/wpnodea/AppServer/jav
a64/bin/classic:/wasv8config/wpcell/wpnodea/AppServer/lib/s390-common:/wasv8config/wpcell/wpnod
ea/AppServer/java64/bin:/wasv8config/wpcell/wpnodea/AppServer/java64/bin/j9vm:/wasv8config/wpcel
l/wpnodea/AppServer/java64/lib/s390/j9vm:/wasv8config/wpcell/wpnodea/AppServer/java64/lib/s390:/
wasv8config/wpcell/wpnodea/AppServer/java64/lib/s390x/j9vm:/wasv8config/wpcell/wpnodea/AppServer
/java64/lib/s390x:/wasv8config/wpcell/wpnodea/AppServer/lib:/wasv8config/wpcell/wpnodea/AppSer
ver/java64/jre/bin:/wasv8config/wpcell/wpnodea/AppServer/java64/jre/lib/s390/j9vm:/wasv8config/wpc
ell/wpnodea/AppServer/java64/jre/lib/s390x/j9vm:/wasv8config/wpcell/wpnodea/AppServer/java64/jre
/lib/s390x
Orb Version = IBM Java ORB build orb626fp1-20110419.00
***** End Display Current Environment *****
[7/22/11 18:47:54:568 GMT] 00000000 ManagerAdmin I TRAS0017I: The startup trace state is
*=info.
[7/22/11 18:47:54:583 GMT] 00000000 ManagerAdmin I TRAS0111I: The message IDs that are in use
are deprecated
```

```

[7/22/11 18:47:54:713 GMT] 00000000 ModelMgr      I  WSVR0800I: Initializing core configuration
models
[7/22/11 18:47:55:600 GMT] 00000000 ComponentMeta I  WSVR0179I: The runtime provisioning
feature is disabled. All components will be started.
[7/22/11 18:47:55:694 GMT] 00000000 CommonBridge A  BBOJ0011I: JVM Build is JRE 1.6.0 IBM J9
2.6 z/OS s390x-64 20110418_80450 (JIT enabled, AOT enabled)
J9VM - R26_Java626_GA_FP1_20110418_1915_B80450
JIT - r11_20110215_18645ifx8
GC - R26_Java626_GA_FP1_20110418_1915_B80450
J9CL - 20110418_80450.
[7/22/11 18:47:55:697 GMT] 00000000 CommonBridge A  BBOJ0051I: PROCESS INFORMATION:
STC05855/WPSR01A , ASID=102(0x66), PID=2497(0x9c1)
...
Lines removed
...
[7/22/11 18:50:03:770 GMT] 00000020 authz      I  CWWIM2000I Initialization of the
authorization component completed successfully.
[7/22/11 18:50:03:787 GMT] 00000020 UserManagemen I  CWWIM6003I Initialization of the dynamic
reload manager completed successfully.
[7/22/11 18:50:03:789 GMT] 00000019 WsServerImpl A  WSVR0001I: Server CONTROL PROCESS wpsr01a
open for e-business
Operation Complete
Processed 412 records in 0.382 seconds (1,078.534 records per second).

```

The above example shows all of the log details available in the repository, because **logViewer.sh** was invoked without any parameters. To see the messages from the most recent server, run **logViewer.sh -instance <your_instance> -latestInstance**.

There are several options to be used with the **logViewer.sh** command, making it a powerful tool to filter events from log and trace repositories. Some possibilities that can be explored are:

- ▶ To show messages starting at a specific level or higher, run:

```
logViewer.sh -instance <your_instance> -minlevel <message_level>
```

- ▶ To show messages from log and trace for a specific thread, run:

```
logViewer.sh -instance <your_instance> -Thread <thread_id>
```

- ▶ To show messages from log and trace in advanced format, run:

```
logViewer.sh -instance <your_instance> -format advanced
```

- ▶ To showing messages from log and trace from a specific time range, run:

```
logViewer.sh -instance <your_instance> -startDate <date/time/timezone>
-stopDate <date/time/timezone>
```

An example of the advanced format view is shown in Example 18-6.

Example 18-6 Advanced format view

```

...
[7/22/11 18:47:56:351 GMT] 00000000 A UOW= source=com.ibm.ws.management.AdminInitializer class=
method= org=IBM prod=WebSphere component=Application Server thread=[main]
        ADMN0015I: The administration service is initialized.
[7/22/11 18:47:57:127 GMT] 00000000 I UOW=
source=com.ibm.ws.management.component.PluginConfigServiceImpl class= method= org=IBM
prod=WebSphere component=Application Server thread=[main]
        PLGC0057I: The plug-in configuration service started successfully.

```

```
[7/22/11 18:47:57:177 GMT] 00000000 I UOW= source=com.ibm.ws.ssl.core.SSLComponentImpl class=
method= org=IBM prod=WebSphere component=Application Server thread=[main]
    CWPKI0001I: SSL service is initializing the configuration
[7/22/11 18:47:57:202 GMT] 00000000 W UOW= source=com.ibm.ws.ssl.config.WSKeyStore class=
method= org=IBM prod=WebSphere component=Application Server thread=[main]
    CWPKI0041W: One or more key stores are using the default password.
[7/22/11 18:47:57:220 GMT] 00000000 I UOW= source=com.ibm.ws.ssl.config.SSLConfigManager class=
method= org=IBM prod=WebSphere component=Application Server thread=[main]
    CWPKI0027I: Disabling default hostname verification for HTTPS URL connections.
[7/22/11 18:47:57:229 GMT] 00000000 I UOW= source=com.ibm.ws.ssl.core.SSLDiagnosticModule
class= method= org=IBM prod=WebSphere component=Application Server thread=[main]
    CWPKI0014I: The SSL component's FFDC Diagnostic Module
com.ibm.ws.ssl.core.SSLDiagnosticModule registered successfully: true.
[7/22/11 18:47:57:230 GMT] 00000000 I UOW= source=com.ibm.ws.ssl.core.SSLComponentImpl class=
method= org=IBM prod=WebSphere component=Application Server thread=[main]
    CWPKI0002I: SSL service initialization completed successfully
[7/22/11 18:47:57:246 GMT] 00000000 I UOW= source=com.ibm.wsspi.rasdiag.DiagnosticConfigHome
class=com.ibm.wsspi.rasdiag.DiagnosticConfigHome method=setStateCollectionSpec org= prod=
component= thread=[main]
    RASD0012I: Updating State Collection Spec from Uninitialized Value to .*.*=0
[7/22/11 18:47:57:257 GMT] 00000000 A UOW= source=com.ibm.ws.pmi.component.PMIImpl class=
method= org=IBM prod=WebSphere component=Application Server thread=[main]
    CWPMI1001I: PMI is enabled
...

```

More information about the **logViewer.sh** command is available at the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Finfo%2Fzseries%2Fae%2Frtrb_logviewer.html

Another way to monitor logs and traces is by using the WebSphere console. Click **Troubleshooting → Logs and trace → <your_server> → View HPEL logs and trace**, as shown in Figure 18-11.

Viewing log records from server instance July 22, 2011 18:48:52 1311360532526					
Number of records to show: 20					
TimeStamp	Thread ID	Logger	Level	Message	
7/22/11 18:47:54.568	00000000	loggerAdmin	INFO	TRAS0017I :	The startup trace state is *=info.
7/22/11 18:47:54.583	00000000	loggerAdmin	INFO	TRAS0111I :	The message IDs that are in use are deprecated
7/22/11 18:47:54.713	00000000	g.ModelMgr	INFO	WSVR08001 :	Initializing core configuration models
7/22/11 18:47:55.600	00000000	etaDataMgr	INFO	WSVR0179I :	The runtime provisioning feature is disabled. All components will be started.
7/22/11 18:47:55.694	00000000	nmonBridge	AUDIT	BBOJ0011I :	JVM Build is JRE 1.6.0 IBM J9 2.6 z/OS s390x-64 20110418_80450 (JIT enabled, AOT enabled) J9VM - R26_Java626_GA_FP1_20110418_1915_B80450 JIT - r11_20110215_18645ifx8 GC - R26_Java626_GA_FP1_20110418_1915_B80450 J9CL - 20110418_80450.
7/22/11 18:47:55.697	00000000	nmonBridge	AUDIT	BBOJ0051I :	PROCESS INFORMATION: STC05855/WPSR01A , ASID=102(0x66), PID=2497(0x9c1)
7/22/11 18:47:55.699	00000000	nmonBridge	AUDIT	com.ibm.ws390.orb.CommonBridge printProperties	BBOJ0077I : org.osgi.framework.executionenvironment = OSGi/Minimum-/Minimum-1.1.OSGi/Minimum-1.2.JRE-1.1.J2SE-1.2.J2SE-1.3.J2SE-1.4.J2SE-1.5.JavaSE-1.6
7/22/11 18:47:55.700	00000000	nmonBridge	AUDIT	com.ibm.ws390.orb.CommonBridge printProperties	BBOJ0077I : osgi.framework = file:/wasv8config/wpcell/wpnodea/AppServer/org.eclipse.osgi_.jar
7/22/11 18:47:55.701	00000000	nmonBridge	AUDIT	com.ibm.ws390.orb.CommonBridge printProperties	BBOJ0077I : java.home = /wasv8config/wpcell/wpnodea/AppServer/java
7/22/11 18:47:55.702	00000000	nmonBridge	AUDIT	com.ibm.ws390.orb.CommonBridge printProperties	BBOJ0077I : traceSettingsFile = /wasv8config/wpcell/wpnodea/AppServer/config/cells/wpcell/nodes/wpnodea/servers/wpsr01a/trace.dat
7/22/11 18:47:55.703	00000000	nmonBridge	AUDIT	com.ibm.ws390.orb.CommonBridge printProperties	BBOJ0077I : eclipse.application = com.ibm.ws.bootstrap.WSLauncher
					com.ibm.ws390.orb.CommonBridge printProperties BBOJ0077I : org.osgi.framework.bootdelegation = com.ibm.jvm.com.ibm.lang.management,com.ibm.oti.reflect,com.ibm.oti.shared, com.ibm.oti.util.com.ibm.oti.vn.com.ibm.paaS.com.ibm.paaS.internal, com.ibm.tools.attach.com.ibm.tools.attach.javaSE,com.ibm.tools.attach.spi.org.apache.harmony.kernel.vm, org.apache.harmony.annotation.internal.lns.org.apache.harmony.beans.org.apache.harmony.beans.editors.org.apache.harmony.objectweb.asm.org.objectweb.asm.signature.com.ibm.j9drcom.ibm.j9drcorereaders, com.ibm.j9drcorereaders.aix.com.ibm.j9drcorereaders.debugger.com.ibm.j9ddcorereaders.elf,com.ibm.j9ddcorereaders.com.ibm.j9drcorereaders.minidump.com.ibm.j9drcorereaders.osthread,com.ibm.j9drcorereaders.tdump.com.ibm.j9drcom.ibm.j9drcorereaders.tdump.zbedee.le.com.ibm.j9drcorereaders.tdump.zbedee.mvs.com.ibm.j9drcorereaders.tdump.com.ibm.j9drlibraries.com.ibm.j9drutil.com.ibm.j9drlogging.com.ibm.j9drvm23.events, com.ibm.j9drvm23,j9.com.ibm.j9drvm23,j9.gc.com.ibm.j9drvm23,j9.stackmap.com.ibm.j9drvm23,j9.stackwalker com.ibm.j9drvm23,j9.walkers.com.ibm.j9drvm23.pointer.com.ibm.j9drvm23.pointergenerated.com.ibm.j9drvm23.type com.ibm.j9drvm24.events.com.ibm.j9drvm24,j9.com.ibm.j9drvm24.pointer.com.ibm.j9drvm24.pointergenerated, com.ibm.j9drvm24.types.com.ibm.j9drvm26.events.com.ibm.j9drvm26,j9.com.ibm.j9drvm26,j9.gc,

Figure 18-11 HPEL log and trace viewed in the WebSphere console

Expanding the Content and Filtering Details link, you can see all of the possible filter options for viewing logs and traces, as shown in Figure 18-12.

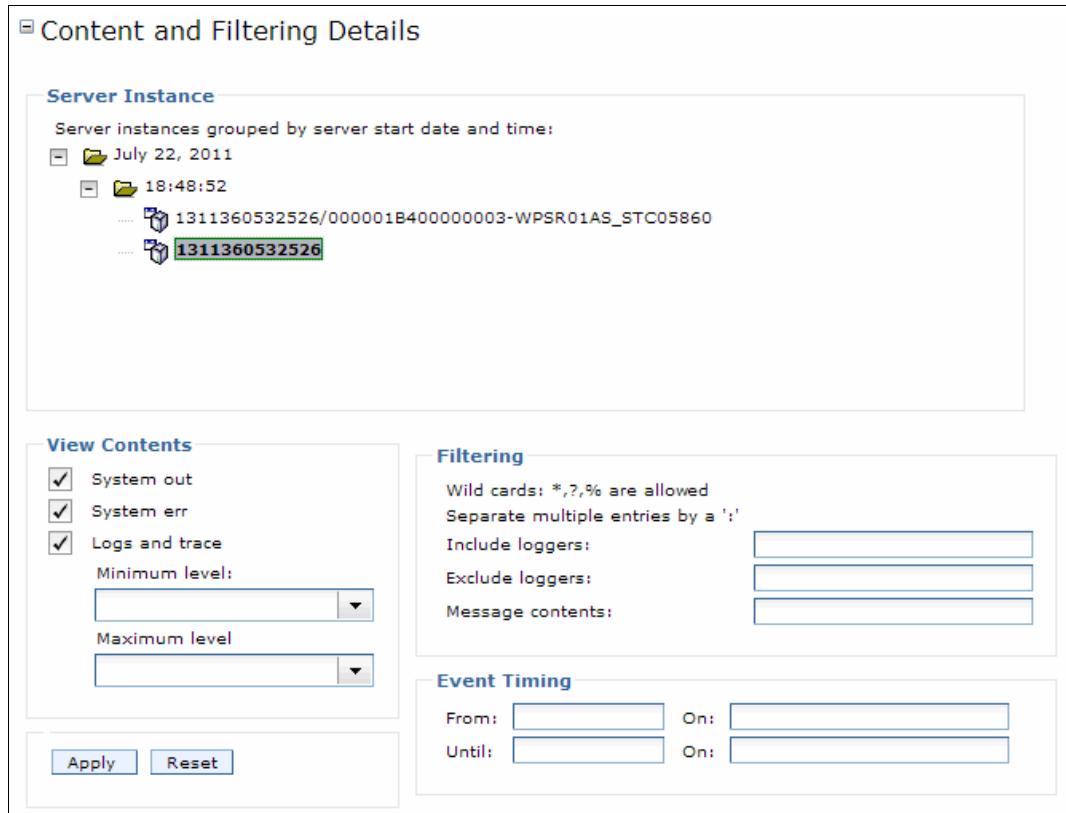


Figure 18-12 HPEL console filtering options

More filtering options are available using the buttons at the top of the console messages, as shown in Figure 18-11 on page 728.

More information about HPEL can be found on the Information Center website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Finfo%2Fzseries%2Fae%2Fctrb_HPELOverview.html

18.4.6 z/OS monitoring

To monitor information on z/OS, several operating systems tools and facilities can be configured and used. We show how to monitor WebSphere from the operating system perspective.

SMF

Information about WebSphere Application Server can be obtained from SMF records. It uses record type 120 to record information about its behavior, so when you start to configure the WebSphere environment, it is important to configure your SMFPRMxx member from PARMLIB to record SMF record type 120. You can change the SYS or SUBSYS statement as shown in Example 18-7.

Example 18-7 SMFPRMxx configuration for record type 120

```
SUBSYS(STC,EXITS(IEFU29,IEFACTRT),INTERVAL(SMF,SYNC),  
TYPE(0,30,70:79,88,89,120,245))
```

Note: After configuring SMFPRMxx, use the SET SMF=xx command to refresh the configuration.

More WebSphere configuration is needed. In the WebSphere console, click **Servers** → **Server Types** → **WebSphere application servers** → <your server name> → **Server Infrastructure** → **Java and Process Management** → **Process definition** → **Control** → **Additional Properties** → **Environment Entries**, and define one or more entries shown in Figure 18-13:

The screenshot shows the 'Environment Entries' page for the 'wpsr01a' server. At the top, there is a header with the path: Application servers > wpsr01a > Process definition > Control > Environment Entries. Below the header, a message says: 'Use this page to specify an arbitrary name and value pair. The value that is specified system configuration properties.' There is a 'Preferences' link. Below that is a toolbar with 'New...', 'Delete', and other icons. A table follows with columns 'Select', 'Name', and 'Value'. The table lists nine SMF properties, each with a checkbox in the 'Select' column:

Select	Name	Value
<input type="checkbox"/>	server_SMF_container_activity_enabled	1
<input type="checkbox"/>	server_SMF_container_interval_enabled	1
<input type="checkbox"/>	server_SMF_interval_length	0
<input type="checkbox"/>	server_SMF_request_activity_CPU_detail	1
<input type="checkbox"/>	server_SMF_request_activity_enabled	1
<input type="checkbox"/>	server_SMF_request_activity_security	1
<input type="checkbox"/>	server_SMF_request_activity_timestamps	1
<input type="checkbox"/>	server_SMF_server_activity_enabled	1
<input type="checkbox"/>	server_SMF_server_interval_enabled	1

Total 9

Figure 18-13 SMF properties

Note: A server restart is needed to validate the SMF configuration.

An overview of SMF record type 120 can be found at the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Finfo%2Fzseries%2Fae%2Frtrb_SMFr120overview.html&resultof=%22smf%22

To gain a better understanding of SMF record type 120, subtype 9, go to the following website:

<http://www.ibm.com/support/techdocs/atstrmstr.nsf/WebIndex/WP101342>

After collecting SMF records, you can use the SMF browser to look at the data. The tool is available at the following website:

<https://www14.software.ibm.com/webapp/iwm/web/preLogin.do?source=zosos390>

You must download the bbomsmfv.jar file and upload it to your z/OS file system. It is a UNIX System Services application. It depends on Java to run, so configure your PATH variable to point to a Java installation directory.

Note: SMF browser testing has been done with Java 5 and later. The browser may work with earlier versions of Java, but this was not tested.

To process the SMF data, first dump the records from the SMF VSAM file to a sequential data set, as shown in Example 18-8.

Example 18-8 Sample job to dump SMF records to sequential data set

```
//SMFDMP JOB (999,POK),'SMFDUMP',MSGLEVEL=(1,1),
// CLASS=A,MSGCLASS=T,NOTIFY=&SYSUID
/*
//STEP      EXEC PGM=IFASMFDP
//INDD      DD DSN=<HLQ>.<your_dataset>,DISP=SHR
//OUTDD     DD DSN=LIST.SMFOUT,DISP=(NEW,CATLG,KEEP),UNIT=SYSDA,
// VOL=SER=TARTS3,SPACE=(CYL,(50,10),RLSE)
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
        INDD(INDD,OPTIONS(DUMP))
        OUTDD(OUTDD,TYPE(120))
/*
```

After generating the sequential data set, go to OMVS in the directory where the bbomsmfv.jar file was previously uploaded. Run the command shown in Example 18-9.

Example 18-9 SMF browser invocation

```
java -cp bbomsmfv.jar com.ibm.ws390.sm.smfview.SMF "INFILE(LIST.SMFOUT)"
"PLUGIN(PERFSUM,/tmp/smf.out)"
```

Where:

- ▶ “INFILE(<dataset name>)” is the data set generated by the IFASMFDP utility.
- ▶ “PLUGIN(PERFSUM,<output file>)” is the file that will be generated by SMF browser analysis.

Another option for analyzing the data is to run an RMF post processor job to analyze the report produced on dumped SMF records.

RMF

RMF is the strategic IBM product for performance analysis, capacity planning, and problem determination in a z/OS host environment. It has three monitors:

- ▶ Monitor I: Provides long-term collection for system workload and resource utilization
- ▶ Monitor II: Provides on-demand measurements for use in solving immediate problems
- ▶ Monitor III: Provides short-term data collection and online reports for continuous monitoring of system status and solving performance problems

For WebSphere Application Server V8 monitoring, you can use Monitor III reports and combine information with Monitor I post processor to generate workload activity reports. Details are available at the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Finfo%2Fzseries%2Fae%2Ftprf_capwar.html

Dispatch progress monitor (DPM)

DPM is able to provide information, at specified time intervals, about a dispatched request if it is still being processed after the time interval elapses. You can monitor the following protocols:

- ▶ IIOP
- ▶ HTTP
- ▶ HTTPS
- ▶ MDB
- ▶ SIP
- ▶ SIPS

Using this tool, you define a time interval (not too short) to capture data about long running requests. When the time has elapsed, you can instruct DPM to generate one of the following:

- ▶ SVC dump
- ▶ Java core dump
- ▶ Heap dump
- ▶ Java transaction dump
- ▶ Traceback data

The default time and action are obtained from the WLM classification file (for details about WLM classification, see 16.10, “Tuning workload management on z/OS systems” on page 625). To enable DPM, use the MODIFY command. Example 18-10 shows how to enable DPM for a server, setting the protocol HTTP to five seconds.

Example 18-10 Enabling DPM for HTTP protocol

F <server>,DPM,HTTP=5

Note: To disable DPM for a specific protocol, set the time to 0.

More details about DPM are located at the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Finfo%2Fzseries%2Fae%2Ftprf_monitor_dispatch_requests.html

18.4.7 Summary of monitoring tips

The following summary lists a simple set of best practices about how to establish a useful monitoring environment:

- ▶ Take the time to understand the applications deployed into the environment. Use this knowledge to plan for the kinds of metrics that will be beneficial to understand application performance.
- ▶ Activate monitoring at the planned level in all testing environments, especially when benchmarking. Although it is vital to your ability to understand an environment, monitoring is not free and uses CPU, memory, and other resources. Monitoring can impact capacity planning.
- ▶ Use monitoring to understand normal operations and gain an appreciation of what is normal for your systems.
- ▶ Check for differences in your systems after all release changes, especially if full performance testing and benchmarks have not been re-established.
- ▶ Use the basic metrics of PMI as a good starting set of metrics, but customize them to your needs.

Monitoring WebSphere Application Server V8 alone is not enough. Application server monitoring needs to be part of an overall monitoring strategy.



Part 5

Working with applications



New features for application development and deployment

In this chapter, we provide an overview of the new features for application development and deployment introduced in WebSphere Application Server V8.0. WebSphere Application Server V8.0 supports Java Platform, Enterprise Edition (Java EE) V6, and it has backwards compatibility with older versions. Programming models that were available as features packs in Version 7.0 are now a part of Version 8.0.

In this chapter, we also introduce new features such as Monitored Directory Support, JAX-RS Support, and Integrated Web Services Support. In addition, there are new deployment and development tools for Version 8.0 that we describe briefly.

19.1 Java Enterprise Edition 6 support

Java EE 6 expands the developer value that was introduced in Java EE 5 and continues to focus on developer productivity and ease-of-use enhancements. Version 6 includes the following new features:

- ▶ Support for the EJB 3.1 specification
- ▶ Support for the Context and Dependency Injection (CDI) 1.0 specification at a runtime level that uses the Apache OpenWebBeans 1.x implementation.
- ▶ Java Persistence API (JPA) 2.0
- ▶ Java Servlet 3.0
- ▶ Java API for RESTful Web Services (JAX-RS) 1.1
- ▶ JavaServer Faces (JSF) 2.0
- ▶ JavaServer Pages (JSP) 2.2
- ▶ Bean Validation 1.0
- ▶ Java Architecture for XML Binding (JAXB) 2.2
- ▶ Enterprise Web Services 1.3
- ▶ Java API for XML-Based Web Services (JAX-WS) 2.2
- ▶ Java EE Connector Architecture 1.6

Table 19-1 includes Java EE and JAVA SE versions supported by the versions of WebSphere Application Server from Version 6.0 to Version 8.0.

Table 19-1 Java EE support

Specification or API	WebSphere Application Server version			
	Version 8.0	Version 7.0	Version 6.1	Version 6.0
Java Platform, Enterprise Edition (Java EE) specification Prior to Java EE 5, the specification name was Java 2 Platform, Enterprise Edition (J2EE)	Java EE 6 (JSR 316), Java EE 5, J2EE 1.4, and J2EE 1.3	Java EE 5, J2EE 1.4, and J2EE 1.3	J2EE 1.4, J2EE 1.3, and J2EE 1.2	J2EE 1.4, J2EE 1.3, and J2EE 1.2
Java Platform, Standard Edition (Java SE) specification Prior to Java SE 6, the specification name was Java 2 Platform, Standard Edition (J2SE)	Java SE 6	Java SE 6	J2SE 5	J2SE 1.4.2

For more information about JEE 6.0 support, refer to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=%2Fcom.ibm.websphere.nd.doc%2Finfo%2Fae%2Fae%2Fwelc_transition.html

19.2 Integrated standards-base programming models and extensions

Many of the core programming models in WebSphere Application Server V8.0 were available through feature packs in Version 7.0. Now these models are built into WebSphere Application Server V8.0. This section describes the available programming models.

19.2.1 Session Initiation Protocol applications

Session Initiation Protocol (SIP) applications are Java programs that use at least one SIP servlet written to the JSR 116 specification. WebSphere Application Server V8.0 also supports SIP Servlet Specification 1.1, also referred to as JSR 289. SIP is used to establish, modify, and terminate multimedia IP sessions. SIP negotiates the medium, the transport, and the encoding for the call. After the SIP call is established, the communication takes place over the specified transport mechanism, independent of SIP. Examples of application types that use SIP include voice over IP (VOIP), click-to-call, and instant messaging.

Rational Application Developer for WebSphere V8 provides special tools for developing SIP applications. SIP applications are packaged as SIP archive (SAR) files, and are deployed to the application server using the standard WebSphere Application Server administrative tools. SAR files can also be bundled in a Java EE application archive (EAR file), similar to other Java EE components.

In the application server, the web container and SIP container are converged and are able to share session management, security, and other attributes. In this model, an application that includes SIP servlets, HTTP servlets, and portlets are able to interact, regardless of the protocol. High availability of these converged applications is made possible because of the integration of HTTP and SIP in the base application server. For more information about SIP applications, see the following resources:

- ▶ JSR 289 SIP Servlet API 1.1 Specification, found at the following website:
<http://www.jcp.org/aboutJava/communityprocess/final/jsr289/index.html>
- ▶ JSR 116, found at the following website:
<http://jcp.org/en/jsr/detail?id=116>
- ▶ RFT 3261, found at the following website:
<http://www.ietf.org/rfc/rfc3261.txt>

19.2.2 Java batch programming model

Batch applications can perform long running bulk transaction processing and computationally intensive work. They run as background jobs, described by a job control language, and are supported by infrastructure components that are aimed to support batch workloads.

The control language for batch jobs is called XML job control language (xJCL). It allows users to describe the job steps that are involved in a batch job. The application runs in batch containers that run in designated WebSphere Application Server environments. The batch container is the heart of the batch application support provided in the WebSphere Application Server and WebSphere Extended Deployment Compute Grid offerings. The batch container runs a batch job under the control of an asynchronous bean, which can be thought of as a container-managed thread. The batch container ultimately processes a job definition and carries out the life cycle of a job.

Figure 19-1 shows a typical batch container.

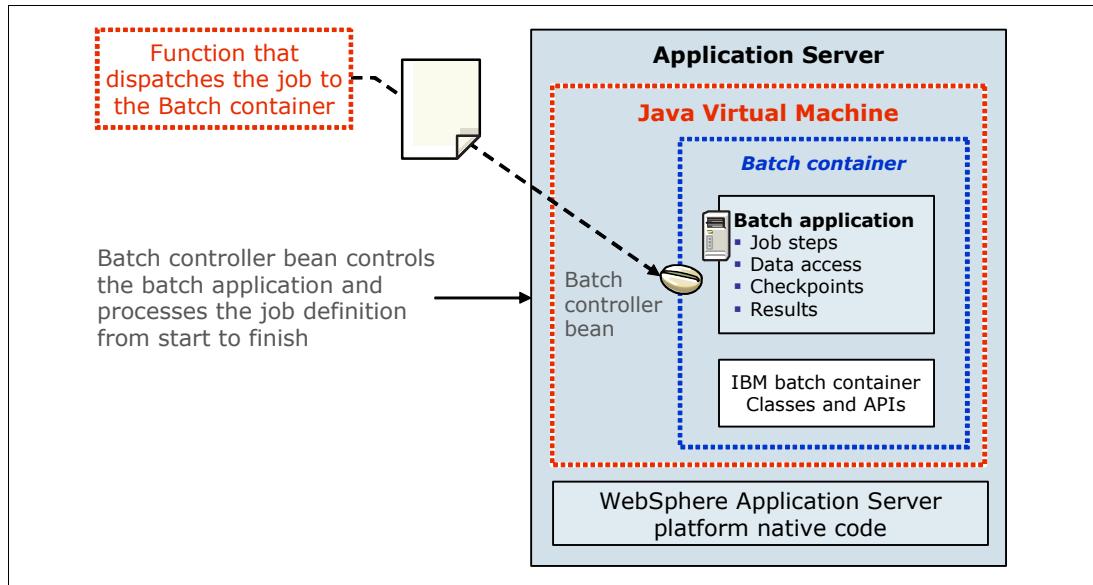


Figure 19-1 Batch container

The batch container provides these services:

- Checkpointing, which involves resuming batch work from a selected position.
- Result processing, which involves intercepting and processing step and job return codes.
- Batch data stream management, which involves reading, positioning, and repositioning data streams to files, relational databases, native z/OS data sets, and many other different types of input and output resources.

Figure 19-2 depicts a typical batch workflow.

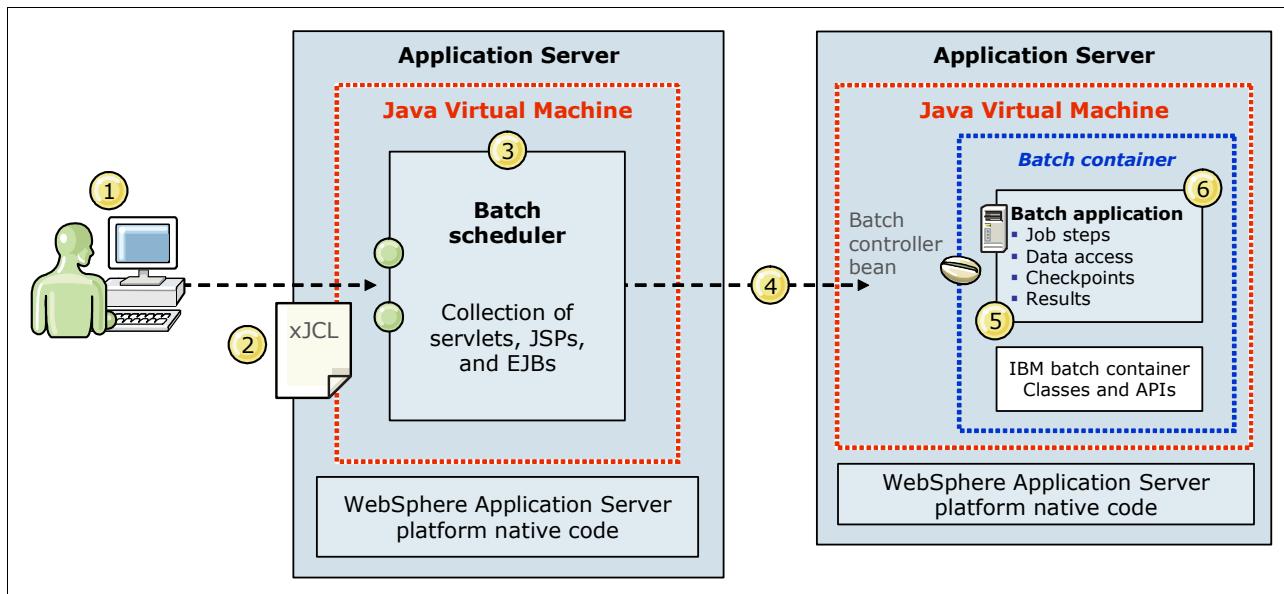


Figure 19-2 Batch workflow

This workload proceeds as follows:

1. Batch jobs are submitted to the system using the Job Management Console or programmatically by way of Enterprise Java Beans (EJB), Java Message Service (JMS), or web services.
2. Each job is submitted in the form of an XML Job Control Language (xJCL) document.
3. The batch scheduler analyzes the request.
4. The job is dispatched to the best endpoint for job execution based on several different metrics.
5. The endpoint sets up the jobs in the batch container and begins executing the batch steps based on the definitions in the xJCL.
6. The batch application is invoked.

The Job Dispatcher aggregates job logs and provides life cycle management functions such as start, stop, cancel, and so on.

19.2.3 OSGi applications programming model

OSGi applications are modular applications that use both Java EE and OSGi technologies. You can design and build applications and suites of applications from coherent, versioned, and reusable OSGi modules that are accessed only through well-defined interfaces. This enables the same, or different, applications to use different versions of the same third-party libraries without interference.

OSGi applications allow the composition of isolated enterprise applications using multiple, multi-version bundles that have dynamic life cycles. Application maintenance and upgrades can be simplified using standard OSGi mechanisms to simultaneously load multiple versions of classes in the same application. Existing web archives (WAR files) can be reused and deployed as OSGi web application bundles. WebSphere Application Server V8.0 support for OSGi applications also includes the capability of deploying web applications that use the Java Servlet 3.0 Specification.

The OSGi Applications support in WebSphere Application Server includes the following major capabilities:

- ▶ Use the OSGi Service Platform Release 4 Version 4.2 Enterprise Specification Blueprint Container for declarative assembly of components. This simplifies unit tests outside of the application server.
- ▶ Use extensions to the Blueprint component model for declarative transactions and container-managed Java Persistence API (JPA).
- ▶ Develop OSGi application projects using IBM Rational Application Developer, which enforces OSGi visibility rules so that projects can only access packages from other projects that explicitly declare them as part of the project externals. This provides environmental support to development best practices.
- ▶ Compose isolated enterprise applications using multiple, versioned bundles with a dynamic life cycle.
- ▶ Deploy applications in archive files that contain only application-specific content and metadata that points to shared bundles. This means that application archive files can be smaller. It also means that, when a library is shared by several OSGi applications, only one copy of the library is loaded into memory.
- ▶ Extend and scale running applications as business needs change, without changing the underlying application.

- ▶ Update a running application, only impacting those bundles affected by the update.
- ▶ Use an integrated bundle repository, and configure the locations of external repositories, to support the provisioning of bundles to applications.
- ▶ Deploy existing web application archive (WAR) files as web application bundles (WABs). This allows web applications to use the OSGi module system.
- ▶ Deploy web applications that use Version 3.0 of the Java Servlet Specification.
- ▶ Simultaneously load multiple versions of classes in the same application, using standard OSGi mechanisms.
- ▶ Administratively update deployed applications in a modular fashion, at the bundle-level.
- ▶ Deploy applications that use their own versions of common utility classes, distinct from the versions that are used by the server runtime environment. This can be done without configuring application Java EE class loader policies, such as PARENT_LAST mode.
- ▶ Use federated lookup mechanisms between the local Java Naming and Directory Interface (JNDI) and the OSGi service registry.

For more information about OSGi applications, go to the following website:

<http://www.osgi.org/About/WhatIsOSGI>

19.2.4 Communications enabled applications

The Communications Enabled Applications (CEA) support in WebSphere Application Server V8 allows you to add dynamic web communications to any application or business process. CEA provides Representational State Transfer (REST) and web service interfaces to enable existing applications to take advantage of communication features involving phone calls and web collaboration.

With the CEA capability, enterprise solution architects and developers can use a single core application to enable multiple modes of communication. CEA applications do not require developers to have extensive knowledge of telephony or SIP. CEA capabilities deliver call control, notifications, and interactivity, providing the platform for more complex communications.

Using this simplified programming model for adding web-based communications, enterprise developers can perform the following tasks:

- ▶ Quickly add communications support to any application; for example, click-to-call integration.
- ▶ Enable shared sessions between users and company representatives.
- ▶ Push relevant session data for application use; for example, customer phone numbers.
- ▶ Deliver automated notifications and instant messaging support.
- ▶ Provide enterprise-grade security, scalability, and high availability.
- ▶ Integrate with customer private branch exchange (PBX) systems.

19.2.5 Service Component Architecture programming model

Service Component Architecture (SCA) offers a way to construct applications based on service-oriented architecture (SOA). SCA is defined in a set of open specifications produced by IBM and other industry leaders through the Open SOA Collaboration (OSOA). The SCA support in WebSphere Application Server V8 support uses the Apache Tuscany open source technology to provide an implementation of the published SCA specifications.

You can use SCA to assemble and compose existing services in your enterprise, and to use your existing services to create new ones. A highlight of SCA is the ease-of-use characteristics of service development in Java. This is accomplished with annotated Plain-Old Java-Object (POJO) components deployed using simple JAR packaging schemes, an easy to use assembly model, and wiring abstractions that enable service definition over different transports and protocols.

The SCA support in WebSphere Application Server V8.0 enables compositions as services, Java components, and integration of key qualities of service-like transactions and security. Mediations, business rules, and business process execution language are treated as any other service. WebSphere Application Server provides the mechanisms to wire services that are implemented in those languages and environments; the product does not provide native support to host those kinds of service implementations.

The SCA implementation in WebSphere Application Server V8 uses the following specifications as guiding principles:

- ▶ SCA assembly model
- ▶ SCA EJB session bean binding
- ▶ SCA Java component implementation
- ▶ SCA Java common annotations and APIs
- ▶ SCA Java EE integration
- ▶ SCA JMS binding
- ▶ SCA policy framework
- ▶ SCA Spring implementation
- ▶ SCA transaction policy
- ▶ SCA web service binding

The features of SCA support include the following:

- ▶ POJO (Java Object) service-component implementations, including support for annotations
- ▶ Asynchronous capability
- ▶ Recursive composition model support
- ▶ Support for SCA services developed from existing WSDL files or Java code
- ▶ Deployment of SCA composites in business-level applications
- ▶ SCA authorization and security identity policies
- ▶ PassByReference optimization for SCA applications
- ▶ Several binding types, including web services binding, SCA default binding, Enterprise JavaBeans (EJB), Java Message Service (JMS), Atom, and HTTP bindings
- ▶ Support for Java Architecture for XML Binding (JAXB) data bindings in SCA applications
- ▶ SCA annotations for Java EE web modules, session beans, and message-driven beans
- ▶ Preview of native SCA deployment
- ▶ Spring 2.5.5 containers in SCA applications
- ▶ OSGi applications as SCA implementations
- ▶ Service Data Objects 2.1.1
- ▶ Sample SCA composites compiled specifically for use with the product

19.2.6 Extensible Markup Language programming model

Extensible Markup Language (XML) structured data has become the predominant format for data interchange. XML data is navigated, queried, or transformed in almost every existing application that runs on WebSphere Application Server V8. This release delivers critical technology that provides application developers with support for the following key World Wide Web Consortium (W3C) XML standards:

- ▶ Extensible Stylesheet Language Transformations (XSLT) 2.0
- ▶ XML Path Language (XPath) 2.0
- ▶ XML Query Language (XQuery) 1.0

These W3C XML standards offer application developers numerous advanced capabilities for building XML applications. WebSphere Application Server V8 support for XML has the following key features and capabilities:

- ▶ An XML application development environment tuned for developer productivity
- ▶ An XML runtime API that offers consistent execution and data navigation API while allowing access to existing Java logic
- ▶ The ability to query large amounts of data stored in XML outside of a database with XQuery 1.0
- ▶ Optimum XML-application performance
- ▶ XML-application reliability with support for XML schema-aware processing and validation
- ▶ 40+ preconfigured samples including four end-to-end scenarios

19.2.7 Integrated Web Services support

Java Architecture for XML Binding (JAXB) 2.2 provides an easy and convenient way to map Java classes and XML schema for simplified development of web services. Version 2.2 provides minor enhancements to its annotations for improved schema generation and better integration with Java API for XML-based web services.

JAX-WS 2.2 simplifies the development of web services with more platform independence for Java applications by the use of proxies and Java annotations. JAX-WS 2.2 requires JAXB 2.2.

19.2.8 Simplified development of server-side REST applications using Java API for RESTful Web Services

Java API for RESTful Web Services (JAX-RS) offers a simpler way to develop, consume, and scale REST applications. It is composed of a collection of interfaces and Java annotations that simplifies the development process. With the annotations, you can declare resource classes and the data types they support. It also allows developers to gain access to the runtime context. Through its extensible framework it is also possible to integrate custom content handlers.

19.3 Monitored directory support

Simply by dragging and dropping applications into a defined and monitored directory, you can speed the process of editing, compiling, deploying, debugging, updating, and uninstalling applications. When an application is moved into the directory, after a defined interval, it is automatically installed and started. Likewise, if the application is removed from the directory, it is stopped and uninstalled. If the application or module is moved into the directory again, it is updated. The supported file types are:

- ▶ Enterprise Archive (EAR)
- ▶ Web Application Archive (WAR)
- ▶ Java Archive (JAR)
- ▶ SIP Application Resource (SAR)

19.4 Development and deployment tools

This section includes development and deployment tools available for WebSphere Application Server V8.

19.4.1 IBM Assembly and Deploy Tools for WebSphere Administration

The new application assembly and deployment tool shipped with WebSphere Application Server V8.0 is called the IBM Assembly and Deploy Tools for WebSphere Administration. This tool replaces the IBM Rational Application Developer Assembly and Deploy shipped with previous releases. The IBM Assembly and Deploy Tools for WebSphere Administration allows the developer to accomplish key assembly and deployment tasks, including editing of deployment artifacts, script development and testing, and application deployment and debugging. It is not intended for general application development.

The key activities you can perform with this tool are:

- ▶ Import and validate applications.
- ▶ Edit deployment descriptors and binding files.
- ▶ Edit EAR-level configuration (Enhanced EAR).
- ▶ Create and debug Jython and `wsadmin` scripts.
- ▶ Deploy EJB and web services.
- ▶ Deploy applications to local or remote WebSphere Application Server V8 servers.
- ▶ Debug applications on WebSphere Application Server V8.

19.4.2 IBM Rational Application Developer Standard Edition for WebSphere Software Version 8

WebSphere Application Server V8 now offers integration with Rational Application Developer Standard Edition V8. The primary objective of this tool is to provide developers the ability to design, develop, deploy, and maintain Java EE applications on the WebSphere Application Server V8 environment. It has support for all Java EE artifacts supported by WebSphere Application Server V8, such as servlets, JSPs, JSFs, EJBs, XML, SIP, Portlet, web services and the new integration with Open Services Gateway initiative (OSGi) programming model. It also supports older versions of Java specifications (J2EE 1.2, 1.3, 1.4, and Java EE 5) and previous versions of WebSphere Application Server (V6 Remote, V6, and V7).

In addition to the features in IBM Assembly and Deploy Tools for WebSphere Administration, Rational Application Developer for WebSphere Software Standard Edition V8 provides the following features:

- ▶ Full support for Java EE 6, including EJB 3.1.
- ▶ Full support for Java Development Kit (JDK) 6.
- ▶ Portal application development with support for WebSphere Portal V6.1 and V7 beta test environments.
- ▶ EJB universal test client and generated EJB test client.
- ▶ Ant scripting and JUnit testing framework.
- ▶ Web 2.0, OSGi, JPA 2.0, SCA, XML, CEA and Web Services development features.
- ▶ WebSphere performance profiling and logging.
- ▶ Agile development support with tools for refactoring code and unit testing.
- ▶ Automated tools to manage server instances and server configurations, including automated creation and submission of wsadmin scripts.
- ▶ Find and deploy to WebSphere or Portal instances in the IBM Smart Business
- ▶ Development and Test Cloud - Smart Business Development and Test on the IBM Cloud (SBDT)

19.4.3 IBM Rational Application Developer for WebSphere Software Version 8

Rational Application Developer for WebSphere Software V8 offers the same development capabilities as the Standard Edition plus additional enhancements, such as team productivity, problem determination, and enterprise connectivity. It takes advantage of collaboration during code analysis and debugging, optimized unit testing and static analysis, and connectivity to enterprise information systems through WebSphere Adapter Support.

The following is a list of features Rational Application Developer for WebSphere Application Server V8 adds in addition to those available in the standard edition:

- ▶ WebSphere Adapter Support for third-party products such as SAP, PeopleSoft Enterprise, Siebel, Oracle E-Business Suite, and JD Edwards.
- ▶ Integration with IBM Rational Team Concert™ and IBM Rational ClearCase®, which makes it possible to perform management operations within the development environment and allows you to have an integrated view of projects and increase collaboration and team productivity.
- ▶ Code quality, testing, and deployment tools, such as the enhanced runtime analysis to detect memory leaks or thread locks.
- ▶ Unified Modeling Language (UML) modeling function.



Understanding class loaders

Understanding how class loaders work is critical to packaging and deploying applications. Failure to configure class loaders properly results in a cascade of class loading exceptions (such as `ClassNotFoundException`) when trying to start your application, or even at run time.

In this chapter, we explain class loaders and how to customize the behavior of the WebSphere class loaders to suit your particular application's requirements. The chapter concludes with an example designed to illustrate these concepts.

We cover the following topics:

- ▶ JVM class loaders
- ▶ WebSphere Application Server and Java EE application class loaders
- ▶ Configuring class loaders for Java EE applications
- ▶ Learning class loaders for Java EE by example
- ▶ OSGi class loaders

20.1 JVM class loaders

Class loaders enable the Java virtual machine (JVM) to load classes. Given the name of a class, the class loader locates the definition of this class. Each Java class must be loaded by a class loader.

When you start a JVM, use one of the following class loaders:

- ▶ The *bootstrap* class loader loads only the core Java libraries in the `Java_home/jre/lib` directory. This class loader, which is part of the core JVM, is written in native code.
- ▶ The *extensions* class loader loads the code in the extensions directories (`Java_home/jre/lib/ext` or any other directory that is specified by the `java.ext.dirs` system property). This class loader is implemented by the `sun.misc.Launcher$ExtClassLoader` class.
- ▶ The *application class loader* loads code that is found on `java.class.path`, which ultimately maps to the system `CLASSPATH` variable. This class loader is implemented by the `sun.misc.Launcher$AppClassLoader` class.

Note: Keep in mind that each JVM has its own set of class loaders. In an environment that is hosting multiple application servers (JVMs), the class loaders for the JVMs are completely separate even if they are running on the same physical machine.

Also note that the JVM uses class loaders called the extensions and application class loaders. The WebSphere run time also uses class loaders, called *extensions*, and application class loaders. However, despite their names, these class loaders are not the same as the JVM class loaders.

The *parent-delegation model* is a key concept to understand when dealing with class loaders. It states that a class loader delegates class loading to its parent before trying to load the class itself. The parent class loader can be either another custom class loader or the bootstrap class loader. However, a class loader can delegate requests only to its parent class loader and never to its child class loaders. (A class loader can go up the hierarchy but never down.)

The extensions class loader is the parent for the application class loader. The bootstrap class loader is the parent for the extensions class loader. Figure 20-1 shows the class loaders hierarchy.

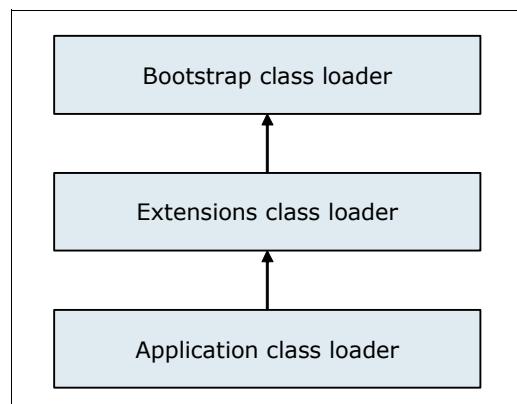


Figure 20-1 Java class loaders hierarchy

If the application class loader needs to load a class, it first delegates to the extensions class loader, which in turn delegates to the bootstrap class loader. If the parent class loader cannot load the class, the child class loader tries to find the class in its own repository. In this manner, a class loader loads only classes that its ancestors cannot load.

This behavior can lead to some interesting problems if a class is loaded from a class loader that is not on a leaf node in the class loader tree. Consider Example 20-1, where a class called WhichClassLoader1 loads a class called WhichClassLoader2, in turn invoking a class called WhichClassLoader3.

Example 20-1 WhichClassLoader1 and WhichClassLoader2 source code

```
public class WhichClassLoader1 {  
  
    public static void main(String[] args) throws javax.naming.NamingException {  
        // Get classpath values  
        String bootClassPath = System.getProperty("sun.boot.class.path");  
        String extClassPath = System.getProperty("java.ext.dirs");  
        String appClassPath = System.getProperty("java.class.path");  
  
        // Print them out  
        System.out.println("Bootstrap classpath =" + bootClassPath + "\n");  
        System.out.println("Extensions classpath =" + extClassPath + "\n");  
        System.out.println("Application classpath=" + appClassPath + "\n");  
  
        // Load classes  
        Object obj = new Object();  
        WhichClassLoader1 wcl1 = new WhichClassLoader1();  
        WhichClassLoader2 wcl2 = new WhichClassLoader2();  
  
        // Who loaded what?  
        System.out.println("Object was loaded by "  
            + obj.getClass().getClassLoader());  
        System.out.println("WCL1 was loaded by "  
            + wcl1.getClass().getClassLoader());  
        System.out.println("WCL2 was loaded by "  
            + wcl2.getClass().getClassLoader());  
  
        wcl2.getTheClass();  
    }  
}  
=====  
public class WhichClassLoader2 {  
  
    // This method is invoked from WhichClassLoader1  
    public void getTheClass() {  
        WhichClassLoader3 wcl3 = new WhichClassLoader3();  
        System.out.println("WCL3 was loaded by "  
            + wcl3.getClass().getClassLoader());  
    }  
}
```

If all WhichClassLoaderX classes are put on the application class path, the three classes are loaded by the application class loader, and this sample runs as expected. However, suppose that you package the WhichClassLoader2.class file in a JAR file that you store in the `Java_home/jre/lib/ext` directory. Example 20-2 shows the output.

Example 20-2 NoClassDefFoundError exception trace

```
Bootstrap classpath
=/opt/IBM/WebSphere/AppServer/java/jre/lib/i386/default/jclSC160/vm.jar:/opt/IBM/WebSphere/AppServer/java/jre/lib/annotation.jar:/opt/IBM/WebSphere/AppServer/java/jre/lib/beans.jar:/opt/IBM/WebSphere/AppServer/java/jre/lib/java.util.jar:/opt/IBM/WebSphere/AppServer/java/jre/lib/jndi.jar:/opt/IBM/WebSphere/AppServer/java/jre/lib/logging.jar:/opt/IBM/WebSphere/AppServer/java/jre/lib/security.jar:/opt/IBM/WebSphere/AppServer/java/jre/lib/sql.jar:/opt/IBM/WebSphere/AppServer/java/jre/lib/ibmorb.jar:/opt/IBM/WebSphere/AppServer/java/jre/lib/ibmorbapi.jar:/opt/IBM/WebSphere/AppServer/java/jre/lib/ibmcfw.jar:/opt/IBM/WebSphere/AppServer/java/jre/lib/rt.jar:/opt/IBM/WebSphere/AppServer/java/jre/lib/charsets.jar:/opt/IBM/WebSphere/AppServer/java/jre/lib/resources.jar:/opt/IBM/WebSphere/AppServer/java/jre/lib/ibmpkcs.jar:/opt/IBM/WebSphere/AppServer/java/jre/lib/ibmcertpathfw.jar:/opt/IBM/WebSphere/AppServer/java/jre/lib/ibmjgssfw.jar:/opt/IBM/WebSphere/AppServer/java/jre/lib/ibmsaslfw.jar:/opt/IBM/WebSphere/AppServer/java/jre/lib/ibmjcefw.jar:/opt/IBM/WebSphere/AppServer/java/jre/lib/ibmjgssprovider.jar:/opt/IBM/WebSphere/AppServer/java/jre/lib/ibmjssprovider2.jar:/opt/IBM/WebSphere/AppServer/java/jre/lib/ibmcertpathprovider.jar:/opt/IBM/WebSphere/AppServer/java/jre/lib/ibmxmlcrypto.jar:/opt/IBM/WebSphere/AppServer/java/jre/lib/management-agent.jar:/opt/IBM/WebSphere/AppServer/java/jre/lib/xml.jar:/opt/IBM/WebSphere/AppServer/java/jre/lib/jlm.jar:/opt/IBM/WebSphere/AppServer/java/jre/lib/javascript.jar
Extensions classpath =/opt/IBM/WebSphere/AppServer/java/jre/lib/ext
Application classpath=

Exception in thread "main" java.lang.NoClassDefFoundError: WhichClassLoader3
  at java.lang.J9VMInternals.verifyImpl(Native Method)
  at java.lang.J9VMInternals.verify(J9VMInternals.java:77)
  at java.lang.J9VMInternals.initialize(J9VMInternals.java:139)
  at WhichClassLoader1.main(WhichClassLoader1.java:18)
```

As the example shows, the program fails with a `NoClassDefFoundError` exception, which might sound strange because `WhichClassLoader3` is on the application class path. The problem is that it is *now* on the wrong class path.

In this example, the `WhichClassLoader2` class was loaded by the extensions class loader. In fact, the application class loader delegated the load of the `WhichClassLoader2` class to the extensions class loader, which in turn delegated the request to the bootstrap class loader. Because the bootstrap class loader could not find the class, the class loading control was returned to the extensions class loader. The extensions class loader found the class on its class path and loaded it.

Now, when a class is loaded by a class loader, any new classes that the class needs reuse the same class loader to load them (or go up the hierarchy according to the parent-delegation model). So, when the `WhichClassLoader2` class needed to access the `WhichClassLoader3` class in this example, it is the extensions class loader that first gets the request to load it. The extensions class loader delegates the request to the bootstrap class path, which cannot find the class, and then tries to load it itself but does not find it either, because `WhichClassLoader3` is not on the extensions class path but on the application class path.

In addition, because the extensions class loader cannot delegate the request to the application class loader (a delegate request can only go up the hierarchy, never down), a `NoClassDefFoundError` exception is thrown.

Note: Remember that developers often also load property files through the class loader mechanism using the following syntax:

```
Properties p = new Properties();
p.load(MyClass.class.getClassLoader().getResourceAsStream("myApp.properties"));
```

Thus, if the `MyClass` class is loaded by the extensions class loader and the `myApp.properties` file is seen only by the application class loader, the loading of the property file fails.

20.2 WebSphere Application Server and Java EE application class loaders

When working with Java Platform, Enterprise Edition (Java EE) applications, two additional types of class loaders are involved:

- ▶ The application server class loaders, which loads all the classes that are needed for the application server in which the enterprise applications are running.
- ▶ The application class loaders, which loads the application classes as defined in the `web.xml` file.

Figure 20-2 shows a typical Java EE class loader.

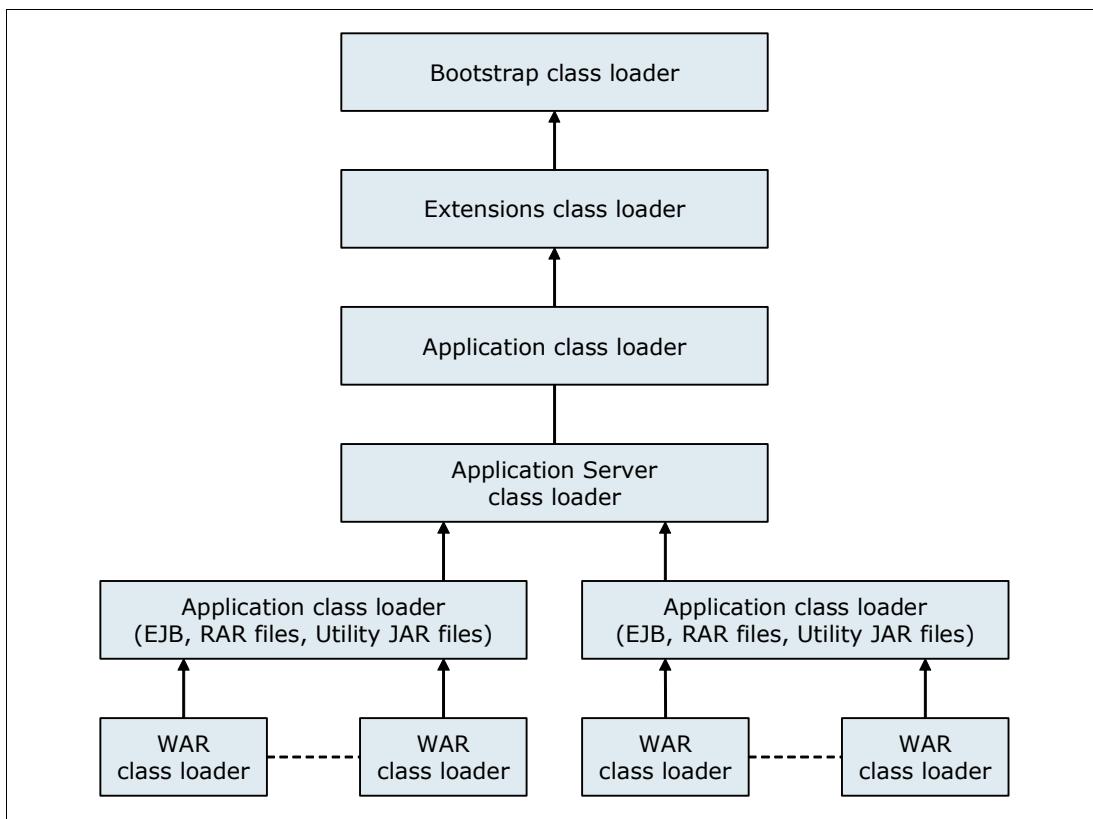


Figure 20-2 Java EE class loading

WebSphere Application Server provides several custom delegated class loaders, similar to those class loaders shown in 20.1, “JVM class loaders” on page 748, but it implements the extensions as OSGi packages, as shown in Figure 20-3.

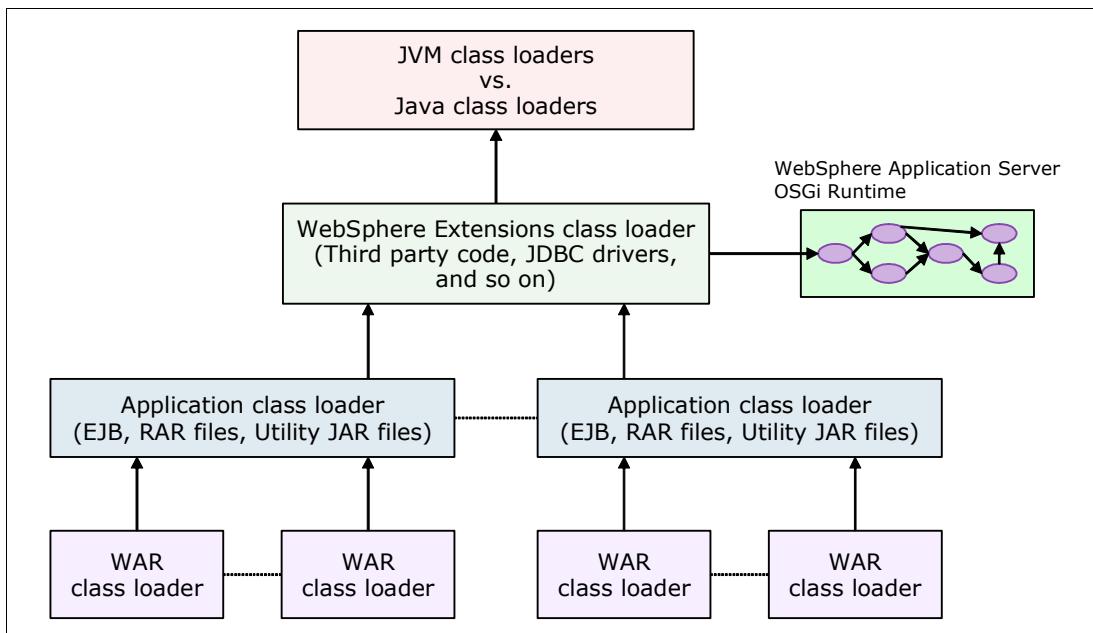


Figure 20-3 WebSphere class loaders hierarchy

The top box represents the Java class loaders (bootstrap, extensions, and application). WebSphere loads just enough here to get itself bootstrapped and to initialize the WebSphere extensions class loader.

We provide information about the other components of this hierarchy in the sections that follow.

20.2.1 WebSphere extensions class loader

The WebSphere extensions class loader is where WebSphere Application Server itself is loaded. WebSphere is packaged as a set of OSGi bundles with each OSGi bundle loaded separately by its own class loader. This network of OSGi class loaders is then connected to the extensions class loader and the remainder of the class loader hierarchy through an OSGi gateway class loader.

Beginning with WebSphere V6.1, extensions began using OSGi packaging and the runtime classes are stored in the *install_root/plugins* directory.

The class path that is used by the extensions class loader is retrieved from the *ws.ext.dirs* system property, which is initially derived from the *WAS_EXT_DIRS* environment variable set in the *setupCmdLine* script file. Example 20-3 shows the default value of the *ws.ext.dirs* property.

Example 20-3 Default value of the ws.ext.dirs property

```
SET
WAS_EXT_DIRS="$JAVA_HOME/lib:$WAS_HOME/classes:$WAS_HOME/lib:$WAS_HOME/installedChannels:$WAS_HOME/lib/ext:$WAS_HOME/web/help:$ITP_LOC/plugins/com.ibm.etools.ejbdeploy/runtime
```

Each directory listed in the *ws.ext.dirs* environment variable is added to the WebSphere extensions class loaders class path and every .jar file and .zip file in the directory is added to the class path.

Although the *classes* and *installedChannels* directories no longer exist in the *install_root* directory, the *setupCmdLine* script still adds them to the extensions class path. Thus, if you have added your own JAR files to one of these directories in previous releases, you can create this directory and add your JAR files to it, and the JAR files are still loaded by the extensions class loader. However, you should avoid this situation by migrating away from such a setup.

Alternatively, if you have developed Java applications that rely on the WebSphere JAR files that were in the *install_root/lib* directory prior to V6.1, you need to modify your application to retain compatibility. WebSphere Application Server provides the following thin client libraries that are designed specifically for such applications:

- ▶ The administrative client library
- ▶ The web services client library

You can find these thin client libraries in the *install_root/runtimes* directory:

- ▶ com.ibm.ws.admin.client_8.0.0.jar
- ▶ com.ibm.ws.webservices.thinclient_8.0.0.jar

These libraries provide everything your application might need to connect to and work with WebSphere. WebSphere Application Server V8 provides the ability to restrict access to internal WebSphere classes so that your applications do not make unsupported calls to WebSphere classes that are not published in the official WebSphere Application Server API. To restrict access to internal WebSphere classes, select the **Access to internal server classes** option.

The default setting for this option is *Allow*, meaning that your applications can make unrestricted calls to non-public internal WebSphere classes. This function might be prohibited in future releases. Therefore, as an administrator, consider switching this setting to *Restrict* to see whether applications still work as expected. If applications depend on non-public WebSphere internal classes, you will receive a `ClassNotFoundException`. In this case, you can switch back to the *Allow* setting. To retain compatibility with future WebSphere Application Server releases, developers can migrate applications so that the applications do not make unsupported calls to the WebSphere internal classes.

20.2.2 Application and web module class loaders

Java EE 6 applications consist of the following primary elements:

- ▶ Web modules
- ▶ EJB modules
- ▶ Application client modules
- ▶ Resource adapter archives (RAR files)
- ▶ Utility JAR files

Utility JAR files contain code that is used by both EJB and servlets. Utility frameworks, such as log4j, are good examples of a utility JAR file.

EJB modules, utility JAR files, resource adapter files, and shared libraries that are associated with an application are always grouped together into the same class loader. This class loader is called the *application class loader*. Depending on the class loader policy, this class loader can be shared by multiple applications (EAR files) or can be unique for each application, which is the default.

By default, web modules receive their own class loader, a WAR class loader, to load the contents of the WEB-INF/classes and WEB-INF/lib directories. You can modify the default behavior by changing the application's WAR class loader policy. You can find this policy setting in the administrative console by clicking **Applications** → **WebSphere enterprise applications** → **application_name** → **Class loading and update detection** → **WAR class loader policy**, which opens the window shown in Figure 20-4.

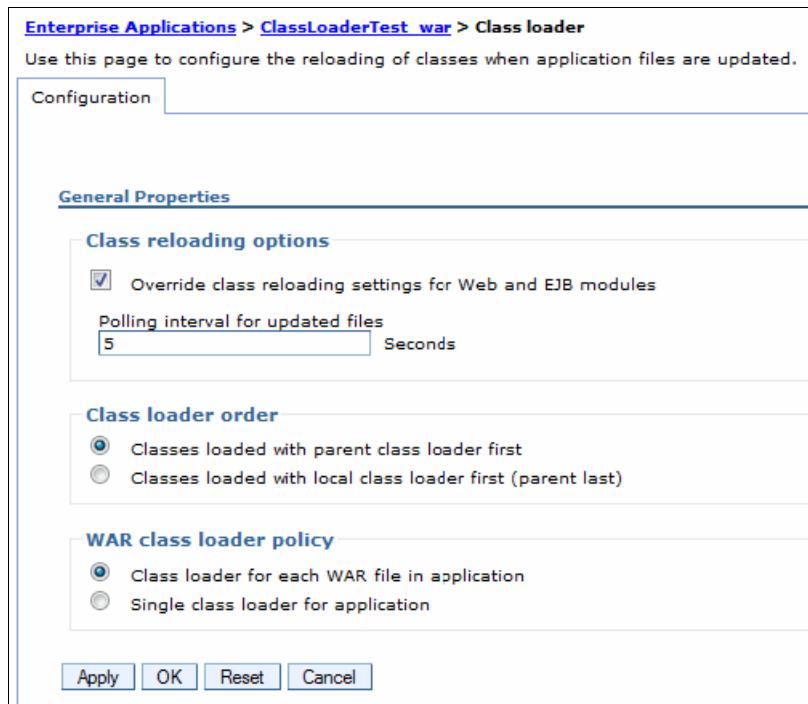


Figure 20-4 WAR class loader policy settings

The default WAR class loader policy setting is the “Class loader for each WAR file in the application” option. This setting is called *Module* in previous releases and in the application deployment descriptor, as viewed in Rational Application Developer.

If the WAR class loader policy is set to the “Single class loader for application” option, the web module contents are loaded by the application class loader in addition to the EJB, RAR files, utility JAR files, and shared libraries. The application class loader is the parent of the WAR class loader.

The application and the WAR class loaders are reloadable class loaders. They monitor changes in the application code to reload modified classes automatically. You can modify this behavior at deployment time.

20.2.3 Handling Java Native Interface code

Because a JVM has only a single address space and because native code can be loaded only once per address space, the JVM specification states that native code can be loaded only by one class loader in a JVM. This design might cause a problem if, for example, you have an application (EAR file) with two web modules that both need to load the same native code through a Java Native Interface (JNI). Only the web module that first loads the library will succeed.

To solve this problem, you can break out just the few lines of Java code that load the native code into a class on its own and place this file on WebSphere's application class loader (in a utility JAR file). However, if you deploy multiple such applications (EAR files) to the same application server, you have to place the class file on the WebSphere extensions class loader instead to ensure that the native code is loaded only once per JVM.

If the native code is placed on a reloadable class loader (such as the application class loader or the WAR class loader), it is important that the native code can unload itself properly if the Java code needs to reload. WebSphere has no control over the native code, and if the native code does not unload and load properly, the application might fail.

If one native library depends on another library, the handling of JNI code can become even more complicated. For more details and troubleshooting, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=/com.ibm.websphere.nd.multiplatform.doc/info/ae/ae/rtrb_classload_viewer.html

20.3 Configuring class loaders for Java EE applications

We provided an overview of WebSphere class loaders and how they work together to load classes. WebSphere Application Server includes settings that allow you to influence the class loader behavior. We provide information about these options in this section.

20.3.1 Application server class loader policies

For each application server in the system, you can set the class loader policy to either *Single* or *Multiple*. From the administrative console, click **Servers** → **Server Types** → **WebSphere application servers** → **server_name**. Then, on the Configuration tab under the “Server-specific Application Settings” section, select the appropriate class loader policy, as shown in Figure 20-5.

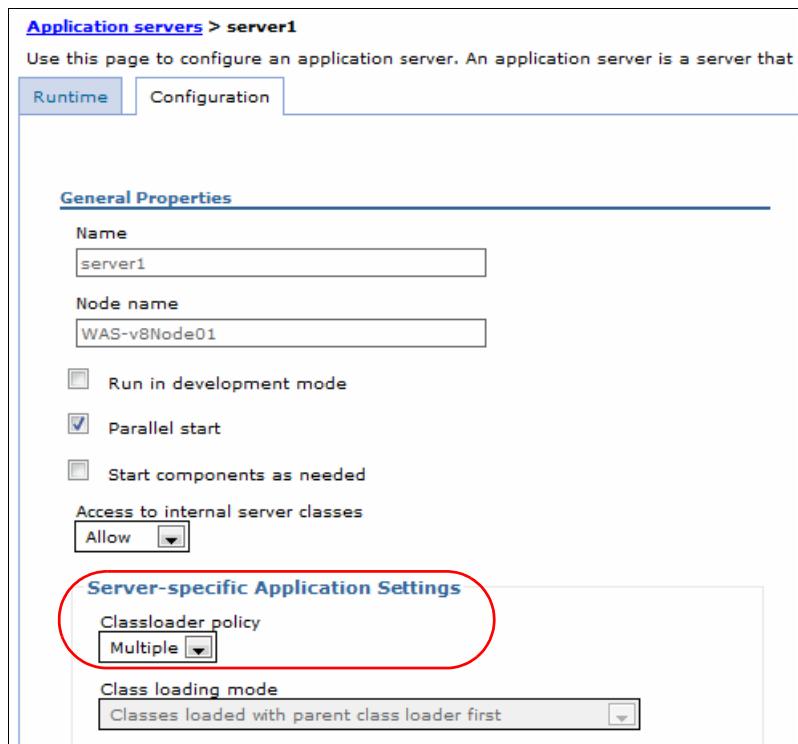


Figure 20-5 Application server class loader settings

When the application server class loader policy is set to Single, a single application class loader is used to load all EJB, utility JAR files, and shared libraries within the application server (JVM). If the WAR class loader policy is set to use the “Single class loader for application” option, the web module contents for this particular application are also loaded by this single class loader.

When the application server class loader policy is set to Multiple, which is the default, each application receives its own class loader for loading EJB, utility JAR files, and shared libraries. Depending on whether the WAR class loader policy is set to use the “Class loader for each WAR file in application” option or the “Single class loader for application” option, the web module might or might not receive its own class loader.

Here is an example to illustrate. Suppose that you have two applications, *Application1* and *Application2*, running in the same application server. Each application has one EJB module, one utility JAR file, and two web modules. If the application server has its class loader policy set to Multiple and the class loader policy for all the web modules are set to use the “Class loader for each WAR file in application” option, the result is as shown in Figure 20-6.

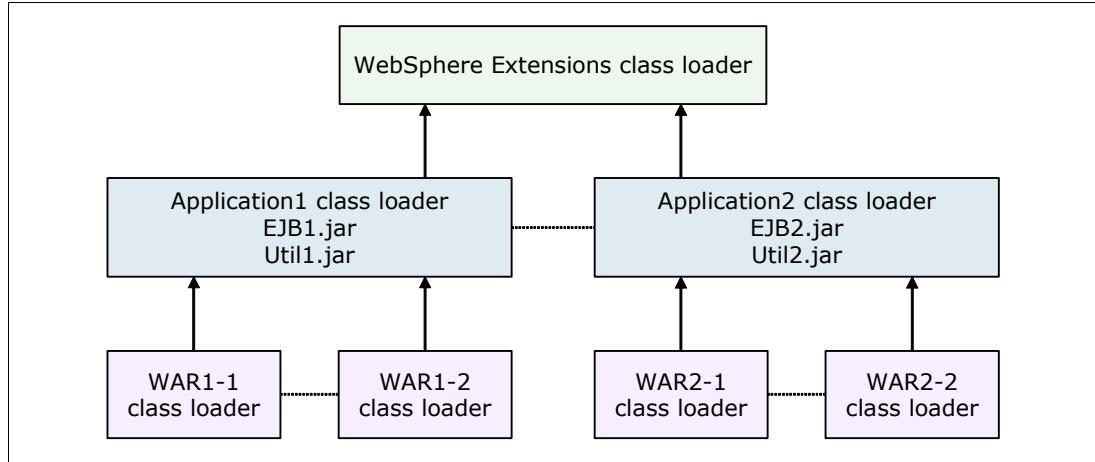


Figure 20-6 Class loader policies - Example 1

Each application is completely separated from the other application, and each web module is completely separated from the other web module in the same application. WebSphere's default class loader policies results in total isolation between the applications and the modules.

Now, if we change the class loader policy for the WAR2-2 module to use the “Single class loader for application” option, the result is shown in Figure 20-7.

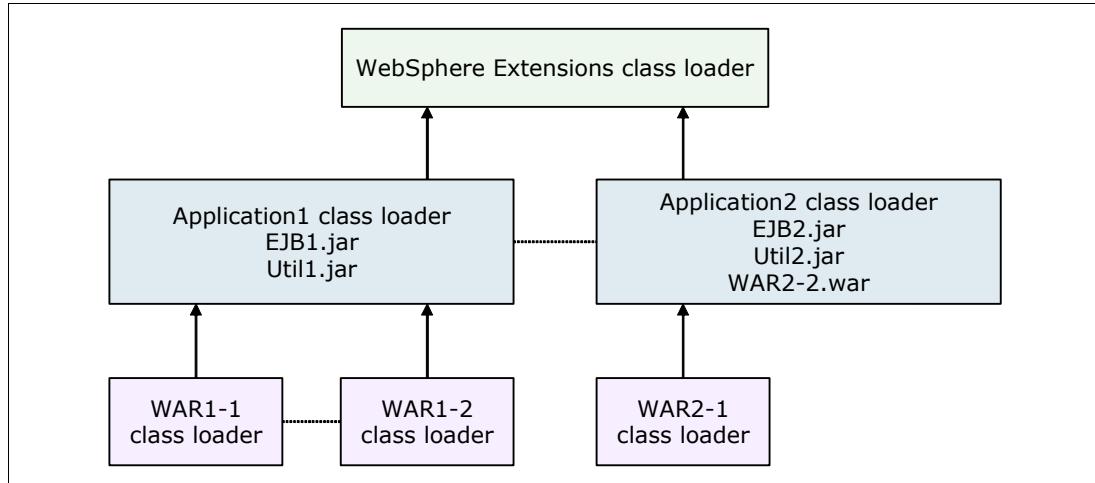


Figure 20-7 Class loader policies - Example 2

Web module WAR2-2 is loaded by Application2's class loader and classes, and for example, classes in the Util2.jar file can see classes in WAR2-2's /WEB-INF/classes and /WEB-INF/lib directories.

As a last example, if we change the class loader policy for the application server to Single and also change the class loader policy for WAR2-1 to use the “Single class loader for application” option, the result is as shown in Figure 20-8.

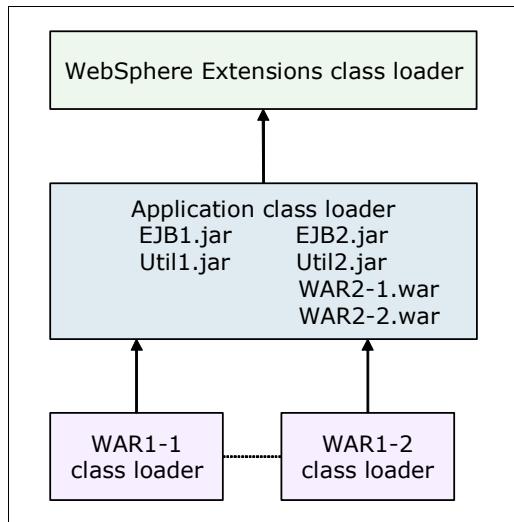


Figure 20-8 Class loader policies - Example 3

There is now only a single application class loader loading classes for both Application1 and Application2. Classes in the Util1.jar file can see classes in the EJB2.jar, Util2.jar, WAR2-1.war, and WAR2-2.war files. However, the classes that are loaded by the application class loader still cannot see the classes in the WAR1-1 and WAR1-2 modules, because a class loader can find only classes by going up the hierarchy, never down.

20.3.2 Class loading and delegation mode

The WebSphere application class loader and WAR class loader both have a setting called the *class loader order* (see Figure 20-5 on page 757). This setting determines whether the class loader order follows the normal Java class loader delegation mechanism, as described in 20.1, “JVM class loaders” on page 748, or whether the class loader overrides it.

The class loading mode uses one of the following options:

- ▶ Classes loaded with parent class loader first
- ▶ Classes loaded with local class loader first (parent last)

In previous WebSphere releases, these settings were called PARENT_FIRST and PARENT_LAST, respectively.

The default value for class loading mode is classes loaded with parent class loader first. This mode causes the class loader to first delegate the loading of classes to its parent class loader before attempting to load the class from its local class path. This policy is the default policy for standard Java class loaders.

If the class loading policy is set to classes loaded with local class loader first (parent last), the class loader attempts to load classes from its local class path before delegating the class loading to its parent. This policy allows an application class loader to override and provide its own version of a class that exists in the parent class loader.

Terminology note: The settings page for a web module includes the following options for class loader order:

- ▶ Classes loaded with parent class loader first
- ▶ Classes loaded with local class loader first (parent last)

However, in this context, the *local class loader* really refers to the WAR class loader, so the “Classes loaded with local class loader first” option actually refers to classes loaded with *WAR class loader* first.

Assume that you have an application, similar to Application1 in the previous examples, and it uses the popular log4j package to perform logging from both the EJB module and the two web modules. Also, assume that each module has its own, unique log4j.properties file that is packaged into the module. You can configure log4j as a utility JAR file so that you have only a single copy of it in the EAR file.

However, if you use this configuration, you might be surprised to see that all modules, including the web modules, load the log4j.properties file from the EJB module. The reason is that when a web module initializes the log4j package, the log4j classes are loaded by the application class loader. The log4j package is configured as a utility JAR file. Thus, it looks for a log4j.properties file on its class path and finds it in the EJB module.

Even if you do not use log4j for logging from the EJB module and the EJB module does not, therefore, contain a log4j.properties file, log4j does not find the log4j.properties file in any of the web modules. Remember that a class loader can find classes only by going up the hierarchy, never down.

To solve this problem, use one of the following approaches:

- ▶ Create a separate file, for example, a Resource.jar file. Configure the file as a utility JAR file, move all the log4j.properties files from the modules into this file, and make their names unique (such as war1-1_log4j.properties, war1-2_log4j.properties, and ejb1_log4j.properties). When initializing log4j from each module, tell it to load the proper configuration file for the module instead of the default file (log4j.properties).
- ▶ Keep the log4j.properties file for the web modules in its original directory (/WEB-INF/classes), add the log4j.jar file to both web modules (in the /WEB-INF/lib directory), and set the class loading mode for the web modules to use the “Classes loaded with local class loader first (parent last)” option. When initializing log4j from a web module, it loads the log4j.jar file from the module itself, and log4j finds the log4j.properties file on its local class path, which is the web module itself. When the EJB module initializes log4j, it loads from the application class loader, and it finds the log4j.properties file on the same class path, which is the one in the EJB1.jar file.
- ▶ If possible, merge all log4j.properties files into one file and place this file on the application class loader (for example, in a Resource.jar file).

Singletons: The singleton pattern is used to ensure that a class is instantiated only once. However, *once only* means *once for each class loader*. If you have a singleton instantiated in two separate web modules, two separate instances of this class are created, that is, one for each WAR class loader. Thus, in a multi-class loader environment, take care when implementing singletons.

20.3.3 Shared libraries

Shared libraries are files used by multiple applications. Examples of shared libraries are commonly used frameworks, such as Apache Struts or log4j. You use shared libraries typically to point to a set of JAR files and associate those JAR files to an application, a web module, or the class loader of an application server. Shared libraries are especially useful when you have different versions of the same framework that you want to associate to different applications.

Shared libraries are defined using the administration tools. They consist of a symbolic name, a Java class path, and a native path for loading JNI libraries. They can be defined at the cell, node, server, or cluster level. However, simply defining a library does not cause the library to be loaded. You must associate the library to an application, a web module, or the class loader of an application server for the classes represented by the shared library to be loaded. Associating the library to the class loader of an application server makes the library available to all applications on the server.

Note: If you associate a shared library to an application, do not associate the same library to the class loader of an application server.

You can associate the shared library to an application using one of the following methods:

- ▶ You can use the administrative console. The library is added by using the **Shared libraries references** link under the References section for the enterprise application.
- ▶ You can use the manifest file of the application and the shared library. The shared library contains a manifest file that identifies it as an extension. The dependency to the library is declared in the application's manifest file by listing the library extension name in an extension list.

For more information about this method, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=/com.ibm.websphere.base.doc/info/aes/ae/ccws_installoptpkg.html

Shared files are associated with the class loader of an application server using the administrative tools in the Server Infrastructure section. To define a new class loader, complete the following steps:

1. Expand **Java and Process Management** and select **Class loader**.
2. Click **New**.
3. After you have defined a new class loader, you can modify it. You can also use the Shared library references link to associate it to the shared libraries that you need.

Refer to 20.4.4, “Example 4: Sharing utility JAR files using shared libraries” on page 768 for more details.

Tip: When designing your application, consider how every class will be loaded to avoid having to add classes in different directories to make the application work.

20.3.4 Class loader viewer

If the class loader viewer service is not enabled, the class loader viewer displays only the hierarchy of class loaders and their class paths, but not the classes that are actually loaded by each of the class loaders. Thus, in this case, the search capability of the class loader viewer is lost.

To enable the class loader viewer service, click **Servers** → **Server Types** → **WebSphere application server** → **server_name**, and click **Class Loader Viewer Service** under the **Additional Properties** link. Then, select the **Enable service at server startup** option. You need to restart the application server for this setting to take effect.

In the next section, we provide examples of how to work with the different class loader settings, and then we use the class loader viewer to illustrate the results.

20.4 Learning class loaders for Java EE by example

We have now described all the different options for influencing class loader behavior. In this section, we take an example and use the options that we discussed so that you can better evaluate the best solution for your applications.

We created a simple application with one servlet and one EJB. Both call a class, VersionChecker, shown in Example 20-4. This class can print the class loader that was used to load the class. The VersionChecker class also has an internal value that can be printed to check the version of the class that we are using. This information is used later to demonstrate the use of multiple versions of the same utility JAR file.

Example 20-4 VersionChecker class source code

```
package com.itso.classloaders;

public class VersionChecker {
    static final public String classVersion = "v1.0";

    public String getInfo() {
        return ("VersionChecker is " + classVersion +
            ". Loaded by " + this.getClass().getClassLoader());
    }
}
```

After being installed, the application can be invoked through <http://localhost:9080/ClassLoaderTestWeb/ExampleServlet>. This invokes the ExampleServlet, which calls VersionChecker and then displays the classloader.

The VersionCheckerV1.jar file contains the VersionChecker class file that returns Version number 1.0. For all the following tests, we have, unless otherwise noted, left the class loader policies and loading modes to their defaults. In other words, we have one class loader for the application and one for the WAR file. Both have their delegation modes set to use the “Classes loaded with parent class loader first” option.

20.4.1 Example 1: Simple web module packaging

For this example, we start with the assumption that our utility class is used only by a servlet. We placed the VersionCheckerV1.jar file under the WEB-INF/lib directory of the web module.

Tip: Place JAR files that are used by a single web module or a JAR file that *only* this web module should see in the WEB-INF/lib directory.

Example 20-5 shows the results of running the application with such a configuration.

Example 20-5 Class loader - Example 1

VersionChecker called from Servlet

VersionChecker is v1.0.

Loaded by

com.ibm.ws.classloader.CompoundClassLoader@18721872[war:ClassLoaderTest/ClassLoaderTestWeb.war]

Local ClassPath:

/opt/IBM/WebSphere/AppServer/profiles/WebSphereV8/installedApps/CeilWSV8/ClassLoaderTest.ear/**ClassLoaderTestWeb.war**/WEB-INF/classes;
/opt/IBM/WebSphere/AppServer/profiles/WebSphereV8/installedApps/CeilWSV8/ClassLoaderTest.ear/**ClassLoaderTestWeb.war**/WEB-INF/lib/**VersionCheckerV1.jar**;
/opt/IBM/WebSphere/AppServer/profiles/WebSphereV8/installedApps/CeilWSV8/**ClassLoaderTest.ear**/**ClassLoaderTestWeb.war**

Parent: com.ibm.ws.classloader.CompoundClassLoader@7f277f27[app:ClassLoaderTest]
Delegation Mode: PARENT_FIRST

We can learn the following information from this trace:

- ▶ The type of the WAR class loader is as follows:
com.ibm.ws.classloader.CompoundClassLoader
- ▶ It searches classes in the following order:
ClassLoaderTestWeb.war/WEB-INF/classes
ClassLoaderTestWeb.war/WEB-INF/lib/VersionCheckerV1.jar
ClassLoaderTestWeb.war

The WEB-INF/classes folder holds unpacked resources (such as servlet classes, plain Java classes, and property files), and the WEB-INF/lib folder holds resources that are packaged as JAR files. You can choose to package your Java code in JAR files and place them in the WEB-INF/lib directory, or you can put them unpacked in the WEB-INF/classes directory. Both directories are on the same class path. Because we developed and exported our sample application from Rational Application Developer, our servlet goes into the WEB-INF/classes folder, because the Java classes are not packaged in a JAR file when exporting an application.

You can also put code or properties in the root of the WAR file. Note, however, that this folder is also the document root for the web server if the File Serving Servlet capabilities are enabled. In this case, any files in this folder are then accessible from a browser. According to the Java EE 6 specification, the WEB-INF folder is protected, which is why the classes and lib folders are under the WEB-INF folder.

The class loader class path is built dynamically at application start.

You can also use the class loader viewer to display the class loader. In the administrative console, click **Troubleshooting** → **Class Loader Viewer**. Then, expand **server1** → **Applications** → **ClassLoaderTest** → **Web modules** → **ClassLoaderTestWeb.war**, as shown in Figure 20-9.

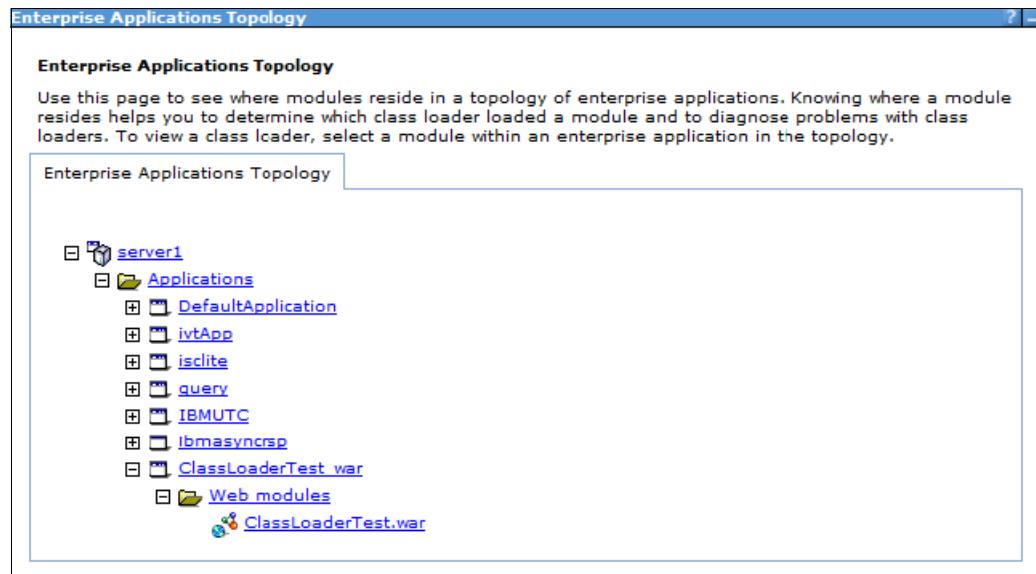


Figure 20-9 Class loader viewer showing applications tree

When the web module is expanded, the class loader viewer shows the hierarchy of class loaders, from the JDK extensions and application class loaders at the top to the WAR class loader at the bottom, called the *compound class loader* (see Figure 20-10).

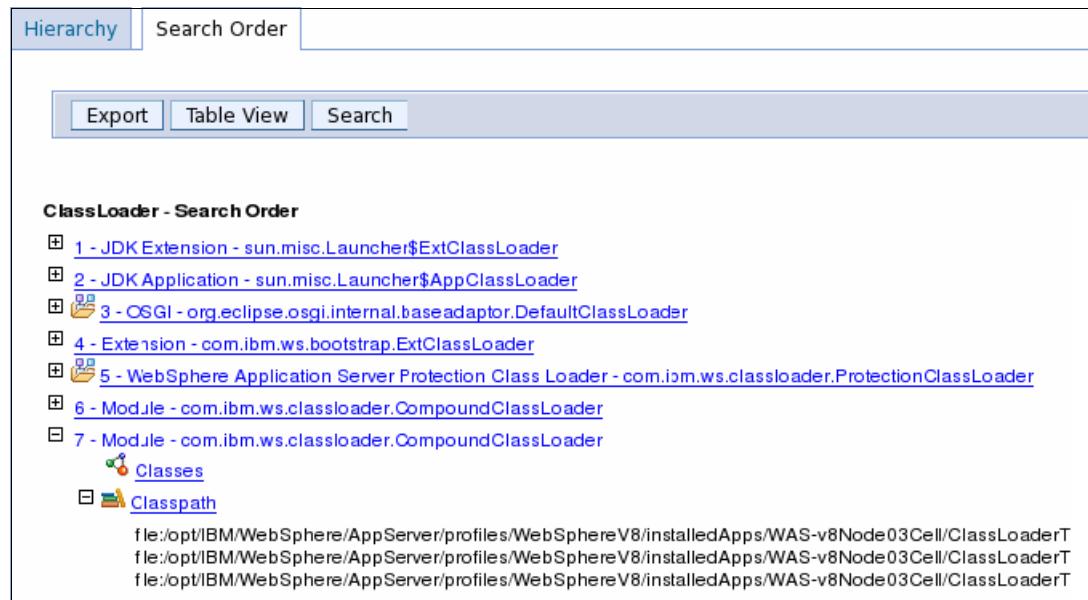


Figure 20-10 Class loader viewer showing class loader hierarchy

If you expand the class path for `com.ibm.ws.classloader.CompoundClassLoader`, you see the same information as the `VersionChecker` class prints (see Figure 20-6 on page 758).

Note: For the class loader viewer to display the classes that are loaded, it must be enabled, as described in 20.3.4, “Class loader viewer” on page 762.

The class loader viewer also has a table view that displays all the class loaders and the classes that are loaded by each of them on a single page. The table view also displays the Delegation mode. For the Delegation mode, *true* means that classes are loaded with parent class loader first, and *false* means that classes are loaded with local class loader first (parent last) or the WAR class loader in the case of a web module (see Figure 20-11). The WAR class loader loaded our example servlet and the VersionChecker class, as expected.

Module - com.ibm.ws.classloader.CompoundClassLoader	
Delegation	<i>true</i>
Classpath	file:/opt/IBM/WebSphere/AppServer/profiles/WebSphereV8/installedApps /WAS-v8Node03Cell/ClassLoaderTest_war.ear/ClassLoaderTest.war/WEB-INF/classes
Classpath	file:/opt/IBM/WebSphere/AppServer/profiles/WebSphereV8/installedApps /WAS-v8Node03Cell/ClassLoaderTest_war.ear/ClassLoaderTest.war/WEB-INF/lib
Classpath	file:/opt/IBM/WebSphere/AppServer/profiles/WebSphereV8/installedApps /WAS-v8Node03Cell/ClassLoaderTest_war.ear/ClassLoaderTest.war

Figure 20-11 Class loader viewer table view

The class loader viewer also has a search feature where you can search for classes, JAR files, folders, and so on. This search capability can be particularly useful if you do not know which of the class loaders loaded a class in which you are interested. The search feature is case sensitive but allows wild cards. So, a search for **VersionChecker** finds our VersionChecker class.

20.4.2 Example 2: Adding an EJB module and utility jar

Next, we add an EJB to our application that also depends on the VersionChecker.jar file. For this task, we added a VersionCheckerV2.jar file to the root of our EAR file. The VersionChecker class in this JAR file returns Version 2.0. To make it available as a utility JAR on the extensions class loader, we add a reference to it in the EJB module’s manifest file, as shown in Example 20-6.

Example 20-6 Updated MANIFEST.MF for EJB module

Manifest-Version: 1.0
Class-Path: VersionCheckerV2.jar

The result is that we now have a web module with a servlet in the WEB-INF/classes folder and the VersionCheckerV1.jar file in the WEB-INF/lib folder. We also have an EJB module that references the VersionCheckerV2.jar utility JAR file in the root of the EAR file.

Example 20-7 shows the test results.

Example 20-7 Class loader - Example 2

VersionChecker called from Servlet
VersionChecker is v2.0.
Loaded by
com.ibm.ws.classloader.CompoundClassLoader@404a404a [app:ClassLoaderTestV2]

```
Local ClassPath:  
/opt/IBM/WebSphere/AppServer/profiles/WebsphereV8/installedApps/WAS-v8Node03Cell1/ClassloaderTestV2.ear/ClassLoaderTestEJB.jar;/opt/IBM/WebSphere/AppServer/profiles/WebsphereV8/installedApps/WAS-v8Node03Cell1/ClassLoaderTestV2.ear/VersionCheckerV2.jar
```

```
Parent: com.ibm.ws.classloader.ProtectionClassLoader@a540a54
```

```
Delegation Mode: PARENT_FIRST
```

VersionChecker called from EJB

```
VersionChecker is v2.0.
```

```
Loaded by
```

```
com.ibm.ws.classloader.CompoundClassLoader@404a404a [app:ClassLoaderTestV2]
```

```
Local ClassPath:
```

```
/opt/IBM/WebSphere/AppServer/profiles/WebsphereV8/installedApps/WAS-v8Node03Cell1/ClassLoaderTestV2.ear/ClassLoaderTestEJB.jar;/opt/IBM/WebSphere/AppServer/profiles/WebsphereV8/installedApps/WAS-v8Node03Cell1/ClassLoaderTestV2.ear/VersionCheckerV2.jar
```

```
Parent: com.ibm.ws.classloader.ProtectionClassLoader@a540a54
```

```
Delegation Mode: PARENT_FIRST
```

As the results show, the VersionChecker is Version 2.0 when called both from the EJB module and the web module. The reason is that the WAR class loader delegates the request to its parent class loader instead of loading it itself. Thus, the utility JAR file is loaded by the same class loader regardless of whether it was called from the servlet or the EJB.

20.4.3 Example 3: Changing the WAR class loader delegation mode

For this example, we consider that we now want the web module to use the VersionCheckerV1.jar file from the WEB-INF/lib folder. For that task, we have to change the class loader delegation from parent first to parent last.

Set the delegation mode to PARENT_LAST by completing the following steps:

1. Select the **WebSphere Enterprise Applications** entry in the navigation area.
2. Select the **ClassLoaderTest** application.
3. Select **Manage modules** under the Modules section.
4. Select the **ClassLoaderTestWeb** module.
5. Change the Class loader order to the **Classes loaded with local class loader first (parent last)** option. Remember, this entry should really be called classes loaded with *WAR class loader* first, as noted in 20.3.2, “Class loading and delegation mode” on page 759.
6. Click **OK**.
7. Save the configuration.
8. Restart the application.

The VersionCheckerV1 in the WEB-INF/lib folder returns a class version of 1.0. Example 20-8 shows that this is the version now used by the WAR file.

Example 20-8 Class loader - Example 3

VersionChecker called from Servlet

VersionChecker is v1.0.

Loaded by

com.ibm.ws.classloader.CompoundClassLoader@1c421c42 [war:ClassLoaderTestV2/ClassLoaderTestWeb.war]

Local ClassPath:

/opt/IBM/WebSphere/AppServer/profiles/WebsphereV8/installedApps/WAS-v8Node03Cell1/ClassLoaderTestV2.ear/ClassLoaderTestWeb.war/WEB-INF/classes;/opt/IBM/WebSphere/AppServer/profiles/WebsphereV8/installedApps/WAS-v8Node03Cell1/ClassLoaderTestV2.ear/**ClassLoaderTestWeb.war/WEB-INF/lib/VersionCheckerV1.jar**; /opt/IBM/WebSphere/AppServer/profiles/WebsphereV8/installedApps/WAS-v8Node03Cell1/ClassLoaderTestV2.ear/ClassLoaderTestWeb.war

Parent: com.ibm.ws.classloader.CompoundClassLoader@1a5e1a5e [app:ClassLoaderTestV2]

Delegation Mode: PARENT_LAST

VersionChecker called from EJB

VersionChecker is v2.0.

Loaded by

com.ibm.ws.classloader.CompoundClassLoader@1a5e1a5e [app:ClassLoaderTestV2]

Local ClassPath:

/opt/IBM/WebSphere/AppServer/profiles/WebsphereV8/installedApps/WAS-v8Node03Cell1/ClassLoaderTestV2.ear/ClassLoaderTestEJB.jar; /opt/IBM/WebSphere/AppServer/profiles/WebsphereV8/installedApps/WAS-v8Node03Cell1/**ClassLoaderTestV2.ear/VersionCheckerV2.jar**

Parent: com.ibm.ws.classloader.ProtectionClassLoader@a540a54

Delegation Mode: PARENT_FIRST

Tip: Use this technique to specify that a web module should use a specific version of a library, such as Struts, or to override classes coming with the WebSphere runtime. Put the common version at the top of the hierarchy and the specialized version in the WEB-INF/lib folder.

The Java EE 6 specification does not provide a standard option to specify the delegation mode in the EAR file. However, by using a WebSphere Extended EAR file, you can specify this setting so that you do not have to change it every time you redeploy your application.

If you use the search feature of the class loader viewer to search for **VersionChecker**, you see the two entries shown in Example 20-9.

Example 20-9 Class Loader Viewer search feature

WAS Module Compound Class Loader (WAR class loader):

file: / opt / IBM / WebSphere / AppServer / profiles / WebsphereV8 / installedApps / WAS-v8Node03Cell1 / ClassLoaderTestV2.ear / ClassLoaderTestWeb.war / WEB-INF / lib / VersionCheckerV1.jar

WAS Module Jar Class Loader (Application class loader):

```
file: / opt / IBM / WebSphere / AppServer / profiles / WebsphereV8 / installedApps  
/ WAS-v8Node03Cell / ClassLoaderTestV2.ear / VersionCheckerV2.jar
```

20.4.4 Example 4: Sharing utility JAR files using shared libraries

In this example, the `VersionCheckerV2.jar` file is used by a single application. If you want to share the JAR file among multiple applications, you could package it within each EAR file. However, changes to this utility JAR file would require that you redeploy all applications. To avoid this situation, you can externalize global utility JAR files using a *shared library*.

Shared libraries can be defined at the cell, node, application server, and cluster levels. After you define a shared library, you must associate it to the class loader of an application server, application, or individual web module. Depending on the target to which the shared library is assigned, WebSphere uses the appropriate class loader to load the shared library.

You can define as many shared libraries as you want. You can also associate multiple shared libraries with an application, web module, or application server.

Using shared libraries at the application level

To define a shared library named `VersionCheckerV2_SharedLib` and associate it to our `ClassLoaderTest` application, complete the following steps:

1. In the administrative console, click **Environment** → **Shared Libraries**.
2. Select the scope at which you want this shared library to be defined, such as Cell, and click **New**.

3. Specify the following properties, as shown in Figure 20-12:

Name Enter VersionCheckerV2_SharedLib.

Class path Enter the list of entries on the class path. Press Enter between each entry. Note that if you need to provide an absolute path, you can use WebSphere variables, such as %FRAMEWORK_JARS%/VersionCheckerV2.jar. Make sure that you declare this variable at the same scope as the shared library for cell, node, server, or cluster.

Native library path Enter a list of DLLs and .so files for use by the JNI code.

If you want to have only one instance of a version of a class shared among applications, select the **Use an isolated class loader for this shared library** option.

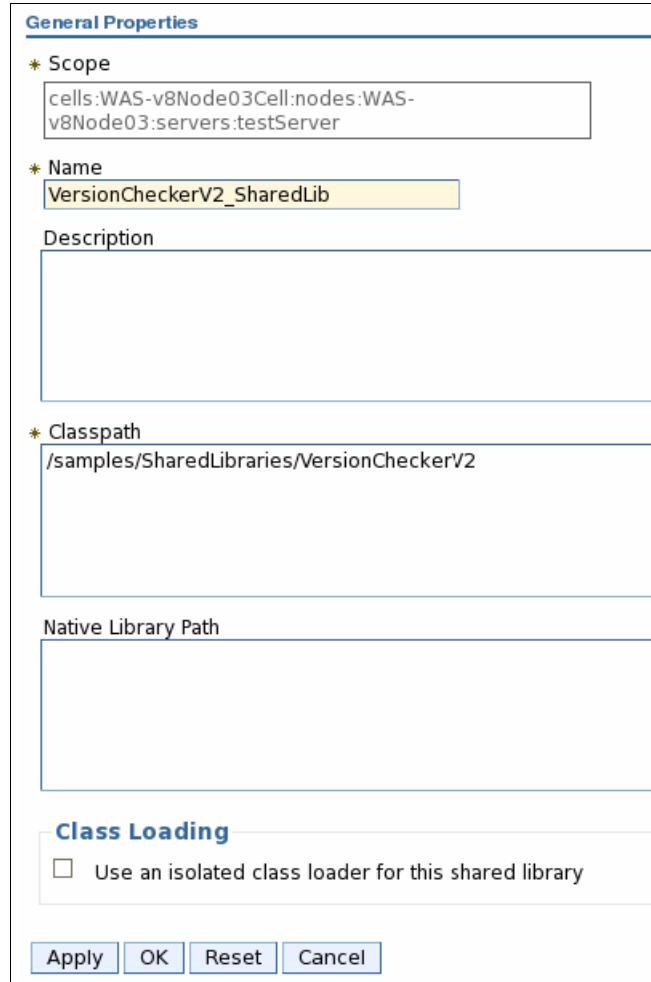


Figure 20-12 Defining a shared library

4. Click **OK**.
5. Click **Applications** → **Application Types** → **WebSphere enterprise applications**.
6. Select the **ClassLoadersTest** application.
7. In References, select **Shared library references**. Then, select **ClassLoadersTest** in the Application row.

- Click **Reference shared libraries**. Then, select **VersionCheckerV2_SharedLib**, and click the arrow button to move it to the Selected column, as shown in Figure 20-13.

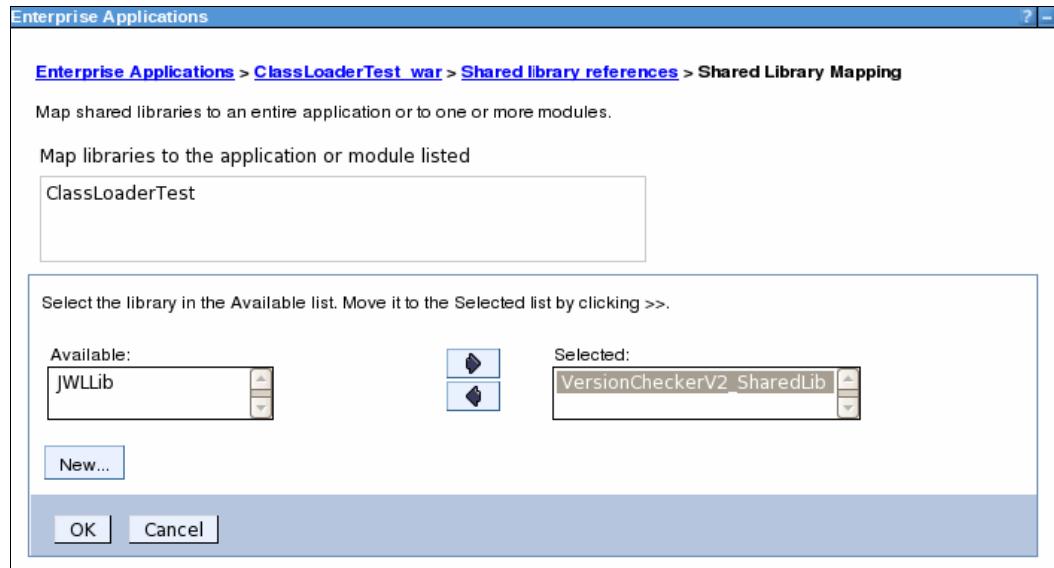


Figure 20-13 Assigning a shared library

- Click **OK**.

The shared library configuration window for the ClassLoaderTest application now looks as shown in Figure 20-14.

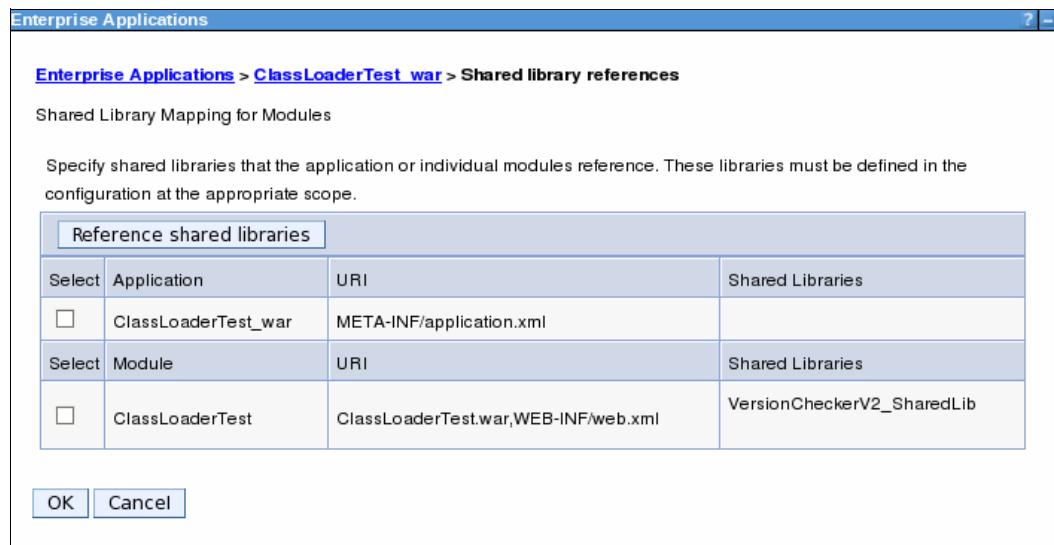


Figure 20-14 Shared library assigned to ClassLoaderTest application

- Click **OK** and save the configuration.

If we remove the VersionCheckerV2.jar file from the root of the EAR file, remove the reference to it from the EJB module's manifest file, and restart the application server, we see the results shown in Example 20-10. Remember that the class loader order for the web module still uses the "Classes loaded with local class loader first (parent last)" option.

Example 20-10 Class loader - Example 5

VersionChecker called from Servlet

VersionChecker is v1.0.

Loaded by

com.ibm.ws.classloader.CompoundClassLoader@405d405d[war:ClassLoaderTestV3/ClassLoaderTestWeb.war]

Local ClassPath:

/opt/IBM/WebSphere/AppServer/profiles/WebsphereV8/installedApps/WAS-v8Node03Cell1/ClassLoaderTestV3.ear/ClassLoaderTestWeb.war/WEB-INF/classes;/opt/IBM/WebSphere/AppServer/profiles/WebsphereV8/installedApps/WAS-v8Node03Cell1/ClassLoaderTestV3.ear/ClassLoaderTestWeb.war/WEB-INF/lib/VersionCheckerV1.jar;/opt/IBM/WebSphere/AppServer/profiles/WebsphereV8/installedApps/WAS-v8Node03Cell1/ClassLoaderTestV3.ear/ClassLoaderTestWeb.war Parent:

com.ibm.ws.classloader.CompoundClassLoader@3e793e79[app:ClassLoaderTestV3]

Delegation Mode: PARENT_LAST

VersionChecker called from EJB

VersionChecker is v2.0.

Loaded by

com.ibm.ws.classloader.CompoundClassLoader@3e793e79[app:ClassLoaderTestV3]

Local ClassPath:

/opt/IBM/WebSphere/AppServer/profiles/WebsphereV8/installedApps/WAS-v8Node03Cell1/ClassLoaderTestV3.ear/ClassLoaderTestEJB.jar;/opt/IBM/examples/VersionCheckerV2.jar Parent: com.ibm.ws.classloader.ProtectionClassLoader@7d677d67

Delegation Mode: PARENT_FIRST

As expected, because of the delegation mode for the web module, the VersionCheckerV1.jar file was loaded when the servlet that is needed the VersionChecker class. When the EJB needed the VersionChecker class, it was loaded from the shared library, which points to the /opt/export/VersionCheckerV2.jar file.

If we want the web module to also use the shared library, we restore the class loader order to the default "Classes loaded with parent class loader first" option for the web module.

Using shared libraries at the application server level

A shared library can also be associated with an application server. All applications deployed on this server see the code listed on that shared library.

To associate a shared library to an application server, you must first create an additional class loader for the application server, as follows:

1. Select an application server.
2. In the Server Infrastructure section, expand **Java and Process Management**. Select **Class loader**.

3. Choose **New**, and select a class loader order for this class loader, either the **Classes loaded with parent class loader first** option or the **Classes loaded with local class loader first (parent last)** option. Click **OK**.
4. Click the class loader that is created.
5. Click **Shared library references**.
6. Click **Add**, and select the library that you want to associate to this application server. Repeat this operation to associate multiple libraries to this class loader. For our example, we selected the **VersionCheckerV2_SharedLib** entry.
7. Click **OK**.
8. Save the configuration.
9. Restart the application server for the changes to take effect.

Because we have now attached the VersionCheckerV2 shared library to the class loader of the application server, we obtain the results shown in Example 20-11.

Example 20-11 Class loader - Example 6

VersionChecker called from Servlet
VersionChecker is v1.0.

Loaded by
com.ibm.ws.classloader.CompoundClassLoader@26a426a4[war:ClassLoaderTestV3/ClassLoaderTestWeb.war]

Local ClassPath:
/opt/IBM/WebSphere/AppServer/profiles/WebsphereV8/installedApps/WAS-v8Node03Cell1/ClassLoaderTestV3.ear/ClassLoaderTestWeb.war/WEB-INF/classes;/opt/IBM/WebSphere/AppServer/profiles/WebsphereV8/installedApps/WAS-v8Node03Cell1/ClassLoaderTestV3.ear/ClassLoaderTestWeb.war/WEB-INF/lib/VersionCheckerV1.jar;/opt/IBM/WebSphere/AppServer/profiles/WebsphereV8/installedApps/WAS-v8Node03Cell1/ClassLoaderTestV3.ear/ClassLoaderTestWeb.war Parent:
com.ibm.ws.classloader.CompoundClassLoader@1f381f38[app:ClassLoaderTestV3]

Delegation Mode: **PARENT_LAST**

VersionChecker called from EJB
VersionChecker is v2.0.

Loaded by **com.ibm.ws.classloader.ExtJarClassLoader@48964896[server:0]**

Local ClassPath: **/opt/IBM/examples/VersionCheckerV2.jar**
Parent: **com.ibm.ws.classloader.ProtectionClassLoader@7d677d67**
Delegation Mode: **PARENT_FIRST**

The new class loader that we defined is called *ExtJarClassLoader*, and it loads the VersionCheckerV2.jar file when requested by the EJB module.

The WAR class loader continues to load its own version due to the delegation mode.

20.5 OSGi class loaders

In the previous sections, we described how Java EE class loaders work. OSGi takes a completely different approach when it comes to class loaders, defining that each bundle has its own class path, as illustrated in Figure 20-15. This approach eliminates the problem with the class path parent delegation approach that we described in 20.1, “JVM class loaders” on page 748.

A series of imports and exports are defined when packaging an OSGi bundle. The imports are then used by the class loader to identify which classes should be loaded for the bundle. For more information about packaging OSGi, see 24.3, “Packaging OSGi applications” on page 879.

The tree organization of the Java EE class loading allows only going up in the tree for loading necessary classes. Thus, to make a library available for different branches of the tree, we must put it in a common ancestor of all the branches that will be using it, forcing all descendent children to use that same version of the library.

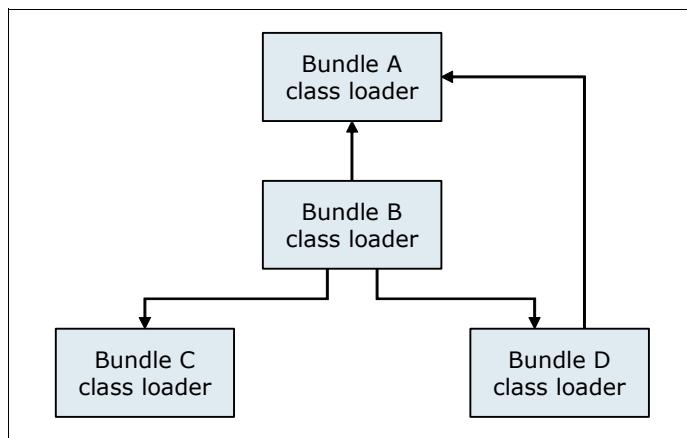


Figure 20-15 OSGi class loading

OSGi bundles allow you to change the standard Java hierarchical and strict class loading structure. Using the information provided in additional metadata properties, an OSGi environment wires a bundle only to declared packages and allows only those packages that are exported explicitly.

Figure 20-16 illustrates the difference between the traditional Java class loading process and the OSGi model. The OSGi model is similar to a network topology, while the standard Java is a static, one-way process.

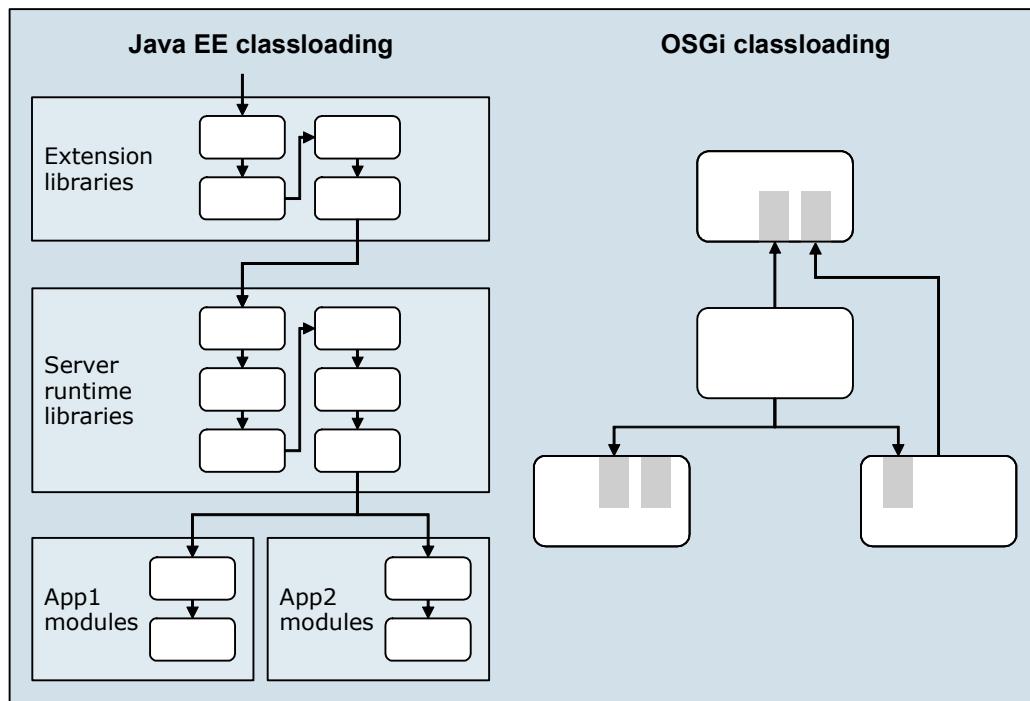


Figure 20-16 Java EE and OSGi class loading

Instead of a hierarchical class loading structure that is traditional in Java, particularly Java EE, OSGi has a network class loading structure, as shown in Figure 20-16. In the hierarchical model, classes are searched from the bottom through various layers of system classes, extension libraries, runtime libraries, and application modules. All classes are visible in the same layer and any lower layers. So, a new library in the extension libraries affects the entire stack as displayed.

In contrast to that approach, OSGi bundles are wired in a network. Furthermore, a bundle has visibility only of the individual packages for which it declares dependencies and no more.

Because by default everything is hidden, the same module can exist many times in such a network, and a single module can depend indirectly on the same module in many different versions. Furthermore, with the modularity metadata, which also allows package dependencies to be marked as optional, it is possible to check whether a bundle will work at run time or whether there are missing dependencies. So, with correctly written metadata, a bundle will never throw a `NoClassDefFoundError` exception at run time.

Note that this change of class loading structure is invasive. Certain assumptions that hold in Java EE no longer hold for OSGi bundles. For example, the thread context class loader is a commonly used mechanism in Java EE to access both application and server runtime classes. However, in OSGi, the thread context class loader contains only the classes that are visible to a single bundle. Consequently, some commonly used libraries are not fully OSGi-compatible even when repackaged.



Packaging and deploying Java EE applications

In this chapter, we provide information about how to work with Java Platform, Enterprise Edition 6 (Java EE 6) applications using Enterprise Java Beans 3.1 (EJB 3.1). We also provide a section about working with business-level applications. First, we cover the packaging of the following Java EE artifacts:

- ▶ Enterprise archives (EAR files)
- ▶ EJB 3.1 modules
- ▶ Web modules
- ▶ Java Persistence API (JPA) persistence units
- ▶ Working with Enhanced EAR files

We also describe some of the IBM enhancements WebSphere Application Server V8 supports in addition to the Java EE specification.

Then we provide information about how to deploy a Java EE file, both as an enterprise application and then as an asset in a business-level application. We explain how to set up the environment for the application and then deploy the application. Finally, we explain how to deploy the client part of the application. Note that you can automate the deployment tasks in this chapter using command-line tools, as explained in Chapter 8, “Administration with scripting” on page 343.

This chapter includes the following topics:

- ▶ Java EE applications introduction
- ▶ Preparing to use the sample application
- ▶ Configuring web module extensions
- ▶ Packaging recommendations
- ▶ Creating WebSphere Enhanced EAR files
- ▶ Exporting an application project to an EAR file
- ▶ Preparing the runtime environment for the application
- ▶ Deploying the application
- ▶ Deploying business-level applications
- ▶ Deploying application clients

21.1 Java EE applications introduction

WebSphere Application Server V8 supports new specifications of Java EE, EJB, and Java Persistence API. In this section, we describe the components of a Java EE application and how to work with each one of them.

21.1.1 Java EE 6 EAR files

WebSphere Application Server V8 supports the Java EE 6 and the EJB 3.1 specifications, which allow developers to use annotations in their source code. These annotations contain information about how the application should be deployed, making much of the packaging and deployment tasks that were necessary in previous versions of WebSphere Application Server no longer relevant. Annotations can also reduce the number of classes and interfaces that the developer needs to manage within the project.

As with previous versions of the Java EE specification, Java EE 6 applications are packaged in enterprise archive (EAR) files. An EAR file can contain web archive (WAR) files, EJB modules (packaged as EJB JAR files), resource adapter archive (RAR) files, Java utility projects (packaged as JAR files), and application client modules. Figure 21-1 shows a schematic overview of a Java EE EAR file.

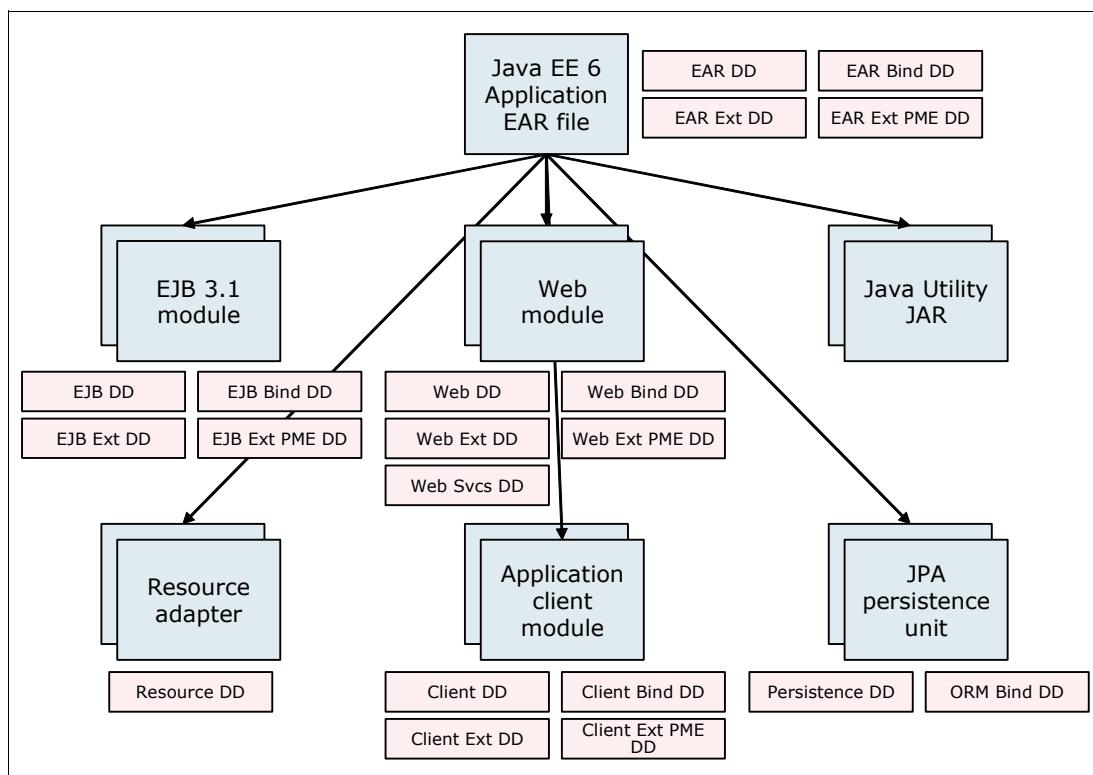


Figure 21-1 Java EE 6 EAR file structure

In previous WebSphere Application Server versions, deployment descriptors were necessary to tell the application server how to deploy the modules and how clients should locate EJB interfaces. In Java EE 6 applications, the use of deployment descriptors is optional. If the code is correctly annotated, the EAR file or modules do not need to contain any deployment descriptors.

Java EE 6 relies on default values for the settings traditionally found in the deployment descriptors, and as long as the default values are acceptable, you do not need to include a deployment descriptor. However, if you include an optional deployment descriptor, the settings in the descriptor override the defaults and the settings given by the annotations in the source code. This method gives the deployer the flexibility to deploy the application as preferred for the target environment.

21.1.2 Development tools

IBM Rational Application Developer for WebSphere Software V8 and IBM Assembly and Deploy Tools for WebSphere Administration for WebSphere 8 provide editors for the deployment descriptors.

IBM Assembly and Deploy Tools for WebSphere Administration ship with the WebSphere Application Server license and are a subset of Rational Application Developer. These tools contain only the Eclipse plug-ins that are needed to develop, test, package, and deploy J2EE and Java EE applications for WebSphere Application Server V8. The productivity enhancements features found in Rational Application Developer are not included.

IBM Assembly and Deploy Tools for WebSphere Administration for WebSphere 8 support testing only on WebSphere Application Server V8. Rational Application Developer for WebSphere Software 8 supports testing on previous versions of WebSphere Application Server.

21.1.3 Enterprise applications

Table 21-1 lists the deployment descriptors that are valid for a WebSphere Application Server V8.0 EAR file.

Table 21-1 Enterprise archive deployment descriptors

File name	Required	Content
application.xml	No	Defines modules and security roles that are used by the enterprise application.
ibm-application-bnd.xml	No	Includes mappings for security roles.
ibm-application-ext.xml	No	Defines WebSphere-specific application extensions.
ibm-application-ext-pme.xml	No	Includes configuration for WebSphere programming model extensions to the Java EE specification.

Because the EAR deployment descriptors are no longer required, the current development tools do not generate them automatically. If you choose to include deployment descriptors, start Rational Application Developer or IBM Assembly and Deploy Tools for WebSphere Administration. Create a new Java EE application project or import an existing project:

- ▶ To create a new project, click **File** → **New** → **Enterprise Application Project**, and then follow the wizard instructions.
- ▶ To import an existing project, click **File** → **Import**, and then follow the wizard instructions.

Then, in the Enterprise Explorer view, right-click the Java EE project, and click **Java EE** → **Generate Deployment Descriptor Stub**.

Expanding the Java EE projects META-INF folder reveals the created application.xml deployment descriptor file. To edit it, either double-click the file or double-click the EAR file's deployment descriptor icon.

On the right side of the deployment descriptor editor is a pane with fields for the information that can be entered into the deployment descriptor. Refer to Figure 21-2. Complete the fields, and then press Ctrl+s to save the deployment descriptor.

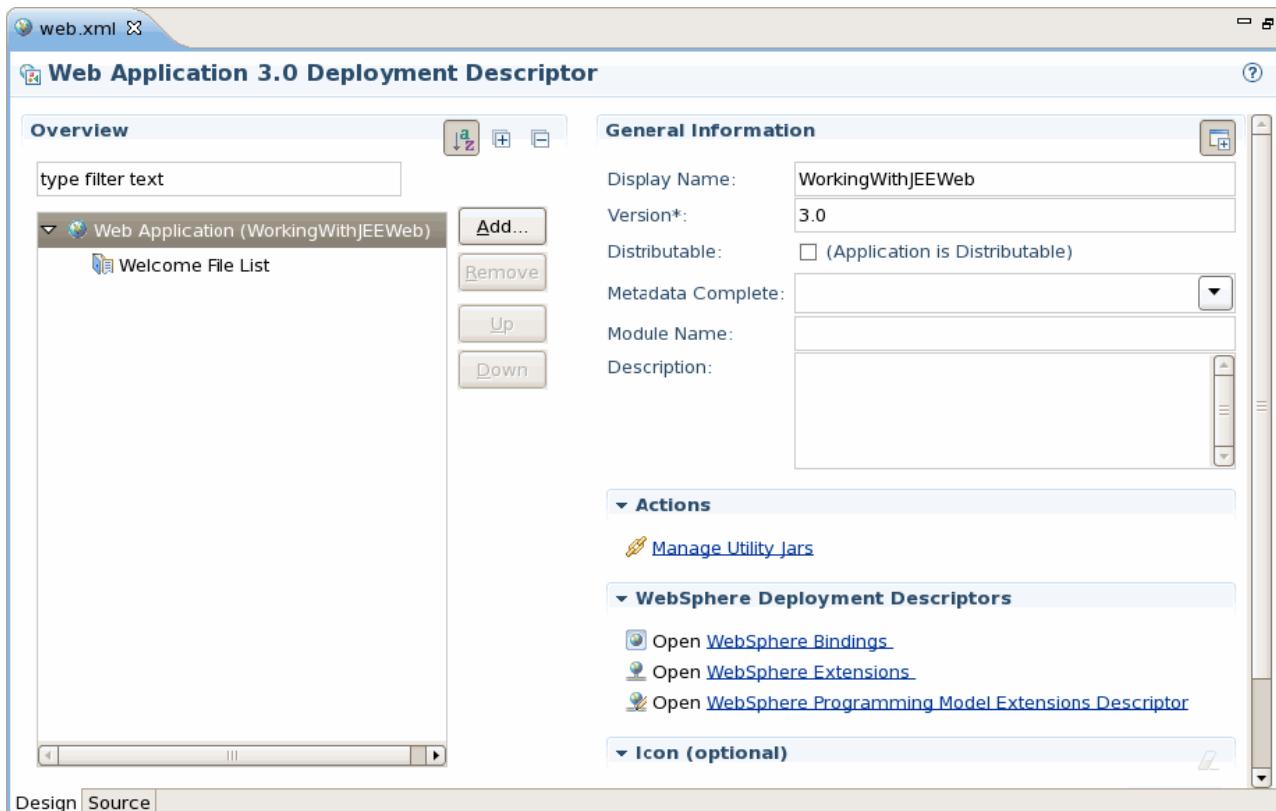


Figure 21-2 Java EE EAR deployment descriptor editor

To create other optional Java EE EAR-file level deployment descriptors, right-click the Java EE EAR project and select **Java EE**. Then select one of the following deployment descriptor options:

- ▶ **Bindings**
- ▶ **Extensions**
- ▶ **Programming Model Extensions Deployment Descriptor**

Selecting any of these options creates the corresponding deployment descriptor file in the Java EE EAR file's META-INF folder. The file can then be accessed either by double-clicking it in the META-INF folder or by clicking the corresponding link under the Actions heading on the main Java EE deployment descriptor editor.

21.1.4 EJB 3.1 modules

The EJB 3.1 and EJB 3.0 specifications are similar; however, EJB 3.1 content can be packaged and deployed as a part of a WAR file. In 21.1.10, “EJB 3.1 content in WAR modules” on page 787, we show an example of how to package and export a WAR file that has EJB 3.1 content.

EJB 3.1 modules can also be packaged without a deployment descriptor. To package a module without a deployment descriptor, you must create a JAR file or WAR file with metadata in an annotation that is located in the EJB component.

The EJB 3.0 or later specification does not support the use of container-managed persistence (CMP) or bean-managed persistence (BMP). JPA is used for data persistence instead. The EJB 2 or later specification and earlier modules continue to support CMP and BMP as per the J2EE specifications. WebSphere Application Server V8 also supports CMP and BMP. Table 21-2 lists the deployment descriptors for an EJB 3 or later module.

Table 21-2 EJB 3.1 deployment descriptors

File name	Required	Content
ejb-jar.xml	No	EJB and EJB method definitions, transaction attributes, resource references, and so on
ibm-ejb-jar-bnd.xml	No	Explicit binding names for EJB and resource references
ibm-ejb-jar-ext.xml	No	Configuration of WebSphere extensions to the Java EE EJB module specification
ibm-web-ext-pme.xml	No	Configuration for WebSphere programming model extensions to the Java EE specification

EJB interface bindings

WebSphere Application Server V8.0 binds EJB 3.1 interfaces and homes into two distinct JNDI namespaces, one JVM-local and one global namespace. Local interfaces and homes are bound to the JVM-local namespace, and remote interfaces and homes are bound to the global namespace.

Unless overridden by explicitly assigned bindings, the interfaces are bound using default names that are generated automatically by the EJB container. Each default name has a short version and a long version. The short name consists of only the Java package name and class name of the interface. The long name prefixes the short name with a component ID, which is composed of the enterprise application name, the module name, and the component name.

Consider an enterprise application called *RAD8EJBWebEAR* that has an EJB module called RAD8EJB.jar with the following bean and interfaces:

- ▶ A session bean with an implementation class called EJBBankBean
- ▶ A local interface called itso.bank.service.EJBBankService
- ▶ A remote interface called itso.bank.service.EJBBankRemote

The local names are bound into the JVM-local namespace called ejblocal. The remote names are bound to the global namespace, and to avoid cluttering the root of the namespace, the long name is prefixed with ejb/.

The auto-generated short and long names for the bean's interfaces are shown in Table 21-3:

Table 21-3 Auto-generated long and short names

Short name	Long name
ejblocal:itso.bank.service.EJBBankService	ejblocal:RAD8EJBWebEAR/RAD8EJB.jar/EJBBankBean#itso.bank.service.EJBBankService
itso.bank.service.EJBBankRemote	ejb/RAD8EJBWebEAR/RAD8EJB.jar/EJBBankBean#itso.bank.service.EJBBankRemote

The auto-generated default names can be overridden by placing a file named `ibm-ejb-jar-bnd.xml` in the EJB JAR module's META-INF directory with the preferred names. By overriding the default names, you can define your own naming convention independently from how the beans are packaged into the application or module hierarchy.

EJB reference resolution using the AutoLink feature

When an EJB client (typically a servlet, or another EJB) wants to call an EJB, it first needs to locate the EJB home in the JNDI namespace. In EJB 2.1 and earlier, this call was completed with a few lines of code written explicitly by the EJB client developer. However, with the EJB 3 or later support and source code annotations, WebSphere Application Server V8 uses a feature called AutoLink that automates this task in many cases, making the lookup code superfluous.

When the EJB container encounters an annotation for an EJB reference, it tries to look up the referenced EJB automatically. The AutoLink algorithm first looks to see if the EJB interface was explicitly given a name in the module's bindings file. If not found, AutoLink searches within the referring module for an EJB that implements the interface. If it does not find exactly one EJB that implements the interface within the same module, AutoLink expands the search scope and searches within other modules that are defined in the application. If it finds exactly one EJB that implements the interface, it uses that EJB as the reference target.

The scope of AutoLink is limited to the enterprise application in which the EJB reference appears and within the application server on which the referring module is assigned. If the target EJB resides in an application other than the client's or if it is deployed on an application server other than the client's, AutoLink does not work. In this case, target bindings must be defined explicitly in the client's bindings file. For an EJB module, this bindings file is the `ibm-ejb-jar.bnd.xml` file, and for a web module, it is the `ibm-web-bnd.xmi` file.

The AutoLink feature handles only EJB references and is available for clients running in the EJB container, web container, or application client container. For more information about AutoLink, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=%2Fcom.ibm.websphere.express.doc%2Finfo%2Fexp%2Fae%2Fcejb_bindingsejbfp.html

Just-in-time generation of EJB deployed code

EJB modules must contain EJB deployed code in order for an application server to be able to run the EJB. EJB deployed code contains application server specific code that bridges the EJB interface and implementation code to the application server's EJB implementation.

In previous versions of WebSphere Application Server and with previous versions of Java EE, the EJB deployed code was generated using one of the following methods:

- ▶ During development, using the Prepare for Deploy action in Rational Application Developer or the WebSphere Application Server Toolkit
- ▶ Before installing an EAR file to WebSphere Application Server using the EJBDeploy tool from a command line
- ▶ During installation of an EAR file to WebSphere Application Server using the installation windows in the administrative console

WebSphere Application Server V8 and EJB 3.1 support a feature called *just-in-time* (JIT) deployment. This feature removes the need to process the EJB modules to generate the deployed code. Instead, the EJB container dynamically generates the necessary code in-memory as needed when the application is running. This feature simplifies and speeds up the development, packaging, and deployment of EJB.

For EJB 3 or later, clients that are not running inside a web container, EJB container, or client container that was upgraded to the EJB 3 or later level, the JIT development does not generate the necessary classes. In this case, use the `createEJBStubs` tool to make the generated classes available on the client's class path. An example of this situation is a servlet running in WebSphere Application Server V6.1 calling an EJB 3 or later bean running in WebSphere Application Server V8. In this case, the EJB stubs are created manually, and the generated classes are added to the servlet's web module.

For details about the `createEJBStubs` tool and the syntax that it uses, refer to the WebSphere Application Server V8 Information Center at the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic/com.ibm.websphere.nd.multiplatform.doc/info/ae/ae/rejb_3stubsmd2.html

Mixing different EJB versions in an EAR file

WebSphere Application Server V8 allows you to mix EJB 1.1, 2 or later, and 3 or later beans in the same Java EE 6 enterprise application. EJB 1.1 and 2 or later beans can be directly carried forward in EJB 1.1 and EJB 2 or later modules, respectively. WebSphere can also handle EJB 1.1, 2 or later, and 3.0 session beans, and EJB 2 or later message driven beans within the same EJB 3 or later module.

Note that the EJB 3.1 specification does not support the use of BMP and CMP entity beans in EJB 3 or later modules. EJB entity beans can be used on Version 8, but they must be packaged in an EJB 2.1 or earlier module. This method will give you time to migrate to the newest specification while you use your existing modules.

21.1.5 JPA persistence units

Persistence units using JPA can be packaged either into the module that uses the persistence unit or in a separate utility JAR file (packaged as a standard .jar file). If packaged as a separate utility JAR file, it must be referenced from the module that uses the persistence unit using the module's META-INF.MF class path directive.

Persistence units require a `persistence.xml` file, which defines a JPA entity manager's configuration. Among other information, the `persistence.xml` file lists the entity classes and the data source to use.

A persistence unit can also include an optional `orm.xml` file that specifies the object-relational mapping configuration. The `orm.xml` file is an alternative to using annotations and can be used to override annotations in the source code to specify how the objects are persisted to the database.

Table 21-4 lists the deployment descriptors that are valid for a JPA persistence unit.

Table 21-4 JPA persistence units deployment descriptors

File name	Required	Content
<code>persistence.xml</code>	Yes	Entity manager's configuration, entity classes, data sources, and so on
<code>orm.xml</code>	No	Object-to-relational mapping annotation overrides

21.1.6 JPA access intents

WebSphere Application Server provides an optimization enhancement for EJB 2 or later entity beans called *access intents*. However, because the EJB 3 or later specifications do not support entity beans, access intents support is not available for EJB 3 or later beans. Instead, WebSphere Application Server V8 provides JPA access intents that can be used to improve performance and scalability for JPA applications.

JPA access intents specify the isolation and lock levels that are used when reading data from a data source. JPA access intents can be used, providing that the following restrictions are honored:

- ▶ Access intent is available for the application in the Java EE server environment.
- ▶ Access intent is applicable to non-query entity manager interface methods. Query uses a query hint interface to set its isolation and read lock values.
- ▶ Access intent is only available for DB2 databases.
- ▶ Access intent is in effect only when pessimistic lock manager is used. To specify a pessimistic lock manager, add the following statement to the persistence unit's property list:

```
<property name="openjpa.LockManager" value="pessimistic"/>.
```

Table 21-5 compares EJB 2 or later access intents to JPA access intents.

Table 21-5 JPA access intents properties

EJB 2 entity bean access intent	JPA access intent	Description
optimistic	isolation: Read Committed	Data is read but no lock is held. Version ID is used on update to ensure data integrity. Other transactions can read and update data.
	lockManager: Optimistic	
	query Hint: ReadLockMode: READ	
pessimistic read	isolation: Repeatable Read	Data is read with shared locks. Other transactions attempting to update data are blocked.
	lockManager: Optimistic	
	query Hint: ReadLockMode: READ	
pessimistic update	isolation: Repeatable Read	Data is retrieved with update or exclusive lock. Other writes are blocked until commit. This access intent can be used to serialize update access to data when there are multiple writers.
	lockManager: Pessimistic	
	query Hint: ReadLockMode: WRITE	
pessimistic exclusive	isolation: Serializable	Data is retrieved with update or exclusive lock. Other writes are blocked until commit. This access intent can be used to serialize update access to data when there are multiple writers.
	lockManager: Pessimistic	
	query Hint: ReadLockMode:WRITE	

JPA access intents are specified in the `persistence.xml` deployment descriptor.

For more information about access intents, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.express.iseries.doc/info/iseriesexp/ae/tejb_accessintentjpa.html

For information about EJB 2 or later access intents, refer to *WebSphere Application Server V6.1: System Management and Configuration*, SG24-7304.

21.1.7 Resource adapters

A resource adapter archive (RAR) module, also called a *connector module*, contains code that implements a library for connecting with a back-end enterprise information system (EIS), such as CICS, SAP, and PeopleSoft. RAR files (called *connectors*) are packaged as a standard Java archive with a .rar file extension. A resource adapter can be installed as a stand-alone adapter or as part of an enterprise application, in which case the resource adapter is referred to as an *embedded* adapter.

A connector module contains a mandatory deployment descriptor file named ra.xml that resides in the module's META-INF directory.

21.1.8 Web modules

Java EE 6 web modules are packaged just like web modules in earlier Java EE versions. A web module can contain servlet code, JSPs, static HTML pages, images, JavaScript, style sheets, and so on.

A common challenge when working with web modules is to make sure that the correct version of a required Java library is loaded. Often web application developers need to include specific third-party libraries, such as log4j or Xalan/Xerces, and must make sure that the correct version of a library is loaded for an application. This requires knowledge about how the EAR and web module's class loaders work. Refer to Chapter 20, "Understanding class loaders" on page 747 for detailed information about this topic.

A web module supports several deployment descriptors, as shown in Table 21-6.

Table 21-6 Web module deployment descriptors

File name	Required	Purpose
web.xml	No	Servlet definitions, URL mappings, and init parameters, servlet listeners, and so on
ibm-web-bnd.xml	No	Mapping of logical resources that are used by the web module to their runtime managed resources
ibm-web-ext.xml	No	Configuration of WebSphere extensions to the Java EE web module specification
ibm-web-ext-pme.xml	No	Configuration for WebSphere programming model extensions to the Java EE specification
webservices.xml	No	Configuration of web services and implementation code

Deployment descriptor note: If an application.xml deployment descriptor is not included in the EAR file, the context root for a web module defaults to the web module's name without the .war extension.

Until Java EE 5, the web.xml deployment descriptor was required. From Java EE 6 on, the deployment descriptor can be replaced by annotations.

21.1.9 WebSphere extensions to web modules

WebSphere Application Server provides multiple extensions for web modules. These extensions are configured in the ibm-web-ext.xml deployment descriptor in the web module. To create this file in Rational Application Developer or IBM Assembly and Deploy Tools for WebSphere Administration, right-click the web module in the Enterprise Explorer view and click **Java EE → Generate WebSphere Extensions Deployment Descriptor**. To edit the file, use either of the following methods:

- ▶ Expand the web module's WebContent/WEB-INF folder, and then double-click the ibm-web-ext.xml file.
- ▶ Click the **Open WebSphere Extensions** link on the web module's deployment descriptor editor for the web.xml file, as shown in Figure 21-3.

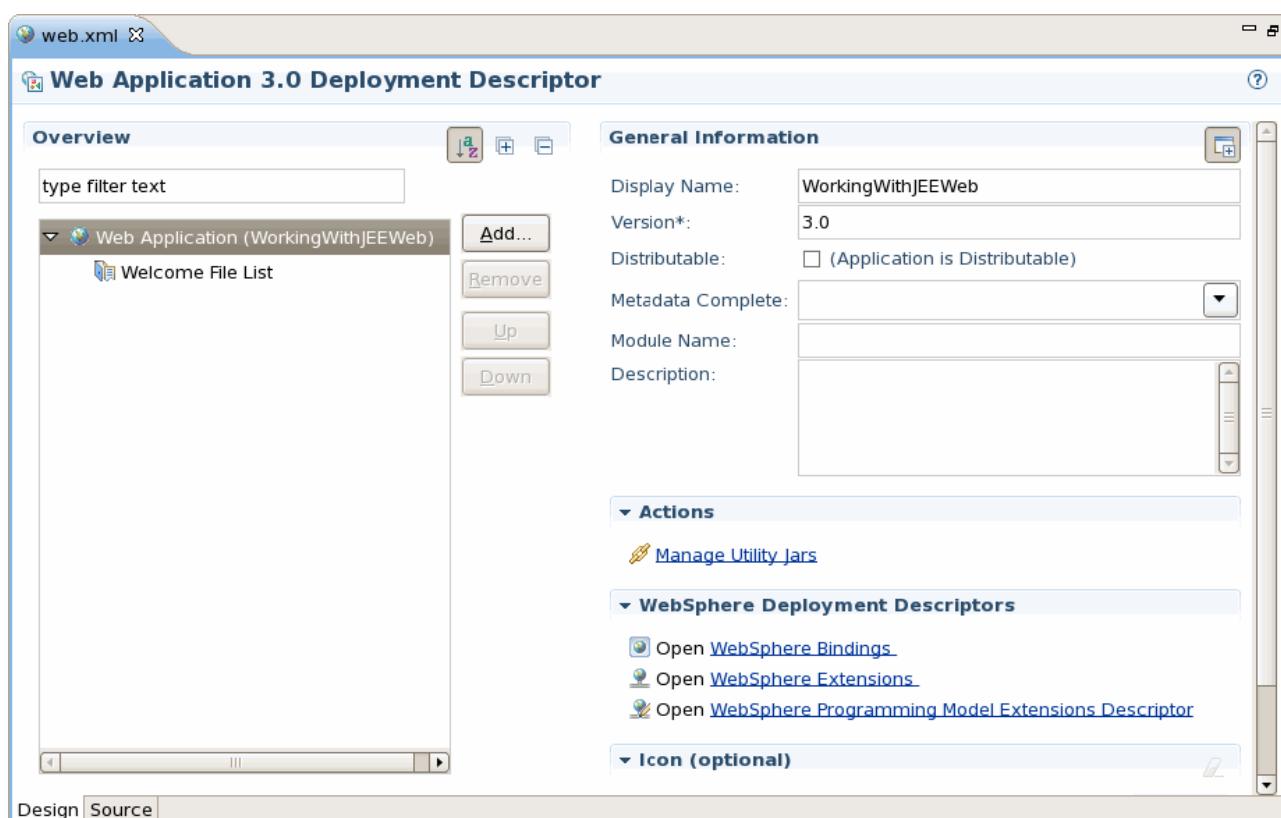


Figure 21-3 Web module's deployment descriptor editor

Figure 21-4 shows the web module extensions editor.

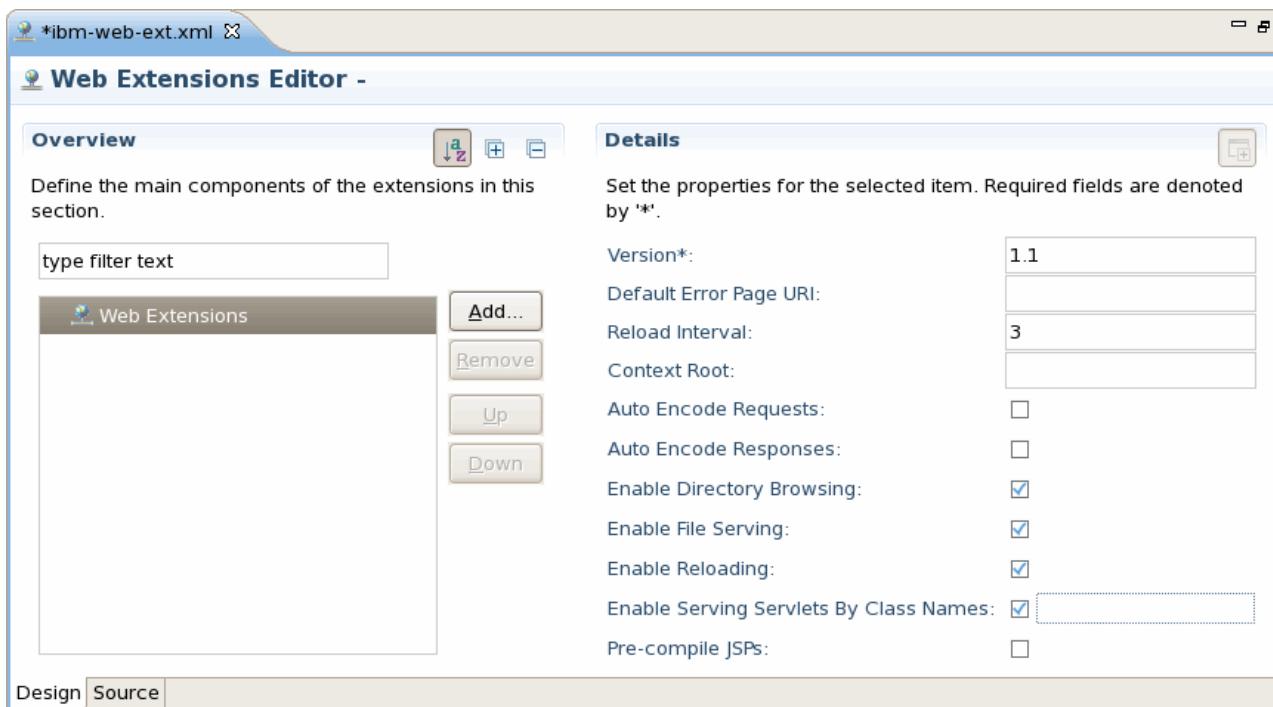


Figure 21-4 Editing WebSphere web module extensions

The following sections provides information about the options found in the web module extensions.

File serving

When dealing with static content (HTML pages, images, style sheets, and so on), you can choose to have these resources served by WebSphere or have them served by the HTTP server itself.

If you want WebSphere to serve the static content of your application, you must enable file serving in the web module extensions deployment descriptor (Figure 21-4). Enabling this feature activates a servlet that serves up any resource file that is packaged in the WAR file. The file serving enabled attribute is set to true by default. By changing it to false, the web server plug-in does not send requests for static content to WebSphere but leaves it up to the HTTP server to serve them.

To enable this option, select the **Enable File Serving** option.

Web application auto reload

Tip: You can experience better performance serving static content from the web server than using WebSphere to serve the static content of your application, because the web server is serving the content directly. Moreover, a web server has more customization options than the file servlet can offer.

However, using the WebSphere file serving servlet has the advantage of keeping the static content organized in a single, deployable unit with the rest of the application. Additionally, it allows you to protect the static pages using WebSphere security.

If you select the **Enable Reloading** option in the web module extensions, the class path of the web application is monitored and all components, JAR files, or class files are reloaded when a component update is detected. The web module's class loader is shut down and restarted. The Reload Interval is the interval between reloads of the web application. It is set in seconds.

The automatic reload feature plays a critical role in hot deployment and dynamic reload of your application.

Important: You must set the Enable Reloading enabled option to **true** for JSP files to be reloaded when they are changed on the file system. Reloading a JSP does not trigger the reload of the web module, because separate class loaders are used for servlets and JSP.

This option is enabled by default, with the reload interval set to 3 seconds. Thus, the classloader checks the classes on the class path for updates every 3 seconds. If any changes are found, those classes are reloaded. If no changes are detected, nothing happens to the classloader or the classes that are loaded. In production mode, you might consider turning this feature off or making the reload interval much higher.

Serving servlets by class name

You can use the invoker servlet to invoke servlets by class name. Note that there is a potential security risk with leaving this option set in production. Use it more as a development-time feature for testing servlets quickly.

A better alternative than this option is to define servlet mappings in the web deployment descriptor for the servlets that should be available.

You configure the invoker servlet by using the “Enable Serving Servlets By Class Names” option.

Default error page

This page is invoked to handle errors if no error page is defined or if none of the defined error pages matches the current error.

Directory browsing

This Boolean defines whether it is possible to browse the directory if no default page has been found.

Turn off this option for improved security.

Pre-compile JSPs

When a JSP is hit for the first time, it is compiled automatically into a servlet and then executed. To avoid this performance penalty the first time a JSP is hit, WebSphere allows JSPs to be pre-compiled during application installation instead of at first invocation. Selecting this option causes the installation of the application to WebSphere to take longer, but the JSPs are served faster on the first hit. You can enable this feature from the deployment descriptor, as shown on Figure 21-9 on page 792.

Automatic HTTP request and response encoding

The web container no longer automatically sets request and response encodings and response content types. The programmer is expected to set these values using the methods available in the Servlet 3.0 API. If you want the application server to attempt to set these values automatically, select the **Auto Encode Requests** option to have the request encoding value set. Similarly, you can select the **Auto Encode Responses** option to have the response encoding and content type set.

The default value of the autoRequestEncoding and autoResponseEncoding extensions is false, which means that both the request and response character encoding is set to the Servlet 3.0 specification default of ISO-8859-1. Different character encodings are possible if the client defines character encoding in the request header or if the code uses the setCharacterEncoding(String encoding) method.

If the autoRequestEncoding value is set to **true**, if the client did not specify character encoding in the request header, and if the code does not include the setCharacterEncoding(String encoding) method, the web container tries to determine the correct character encoding for the request parameters and data.

The web container performs each step in the following list until a match is found:

1. Looks at the character set (charset) in the Content-Type header.
2. Attempts to map the server's locale to a character set using defined properties.
3. Attempts to use the DEFAULT_CLIENT_ENCODING system property, if one is set.
4. Uses the ISO-8859-1 character encoding as the default.

If you set the autoResponseEncoding value to **true** and the following conditions are also true:

- ▶ The client did not specify character encoding in the request header.
- ▶ The code does not include the setCharacterEncoding(String encoding) method.

Then, the web container performs the following actions:

- ▶ Attempts to determine the response content type and character encoding from information in the request header.
- ▶ Uses the ISO-8859-1 character encoding as the default.

21.1.10 EJB 3.1 content in WAR modules

In Java EE 6 applications, it is possible to package EJB content in WAR modules. A bean that is packaged inside a WAR module has the same behavior as a bean that is packaged inside an EJB JAR module, but the rules for packaging vary, depending on the type of module that is being used.

The bean class files must be placed in one of two locations within the WAR file:

- ▶ The WEB-INF/classes directory
- ▶ Within a JAR file that is placed in the WEB-INF/lib directory

In the first approach, the Java file is developed as a part of the same WAR project. When the class is compiled, it is placed in the WEB-INF/classes directory.

The second approach imports the JAR file that contains the EJB contents. This JAR file can also be the result of the packaging of other projects.

If the same class is placed in both the WEB-INF/classes and WEB-INF/lib directories, the instance of the class placed in the WEB-INF/classes directory is loaded, and the instance placed in a JAR file in the WEB-INF/lib directory is ignored.

If the same bean class is placed in two different JAR files in the WEB-INF/lib directory, the server arbitrarily picks one class instance and loads it, ignoring the other class. This method can result in erratic behavior.

If you need to use deployment descriptors, you must place them in the WEB-INF directory.

21.2 Preparing to use the sample application

As an example of how to package and deploy a Java EE 6 application using EJB 3.1, we use the ITSO Bank application that was developed for *Rational Application Developer for WebSphere Software V8 Programming Guide*, SG24-7835. The structure of this application, as seen in the Enterprise Explorer view in Rational Application Developer, is shown in Figure 21-5.

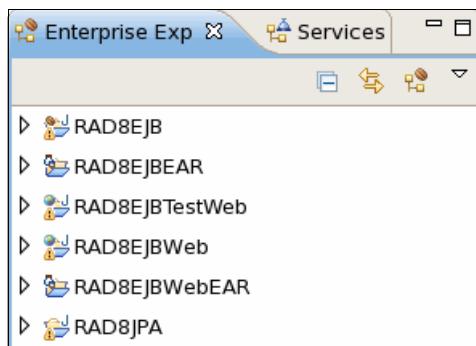


Figure 21-5 Imported ITSO Bank application

When the application has been imported, the workspace will have two enterprise archive (EAR) projects, called *RAD8EJBEAR* and *RAD8EJBWebEAR*. *RAD8EJBEAR* contains EJBs and a simple servlet for testing (in the *RAD8EJBTestWeb* project). A more sophisticated web application is available in the *RAD8EJBWeb* project. We use this application in our example.

The *RAD8EJBWeb* project uses the EJB in the *RAD8EJB* project, which in turn relies on the Persistence Unit in the *RAD8JPA* project.

This section provides the information needed to import the sample application into the IBM Assembly and Deploy Tools for WebSphere Administration. If you are not planning to download the sample application, you can skip to 21.3, “Configuring web module extensions” on page 791.

21.2.1 Downloading the application

To download the sample application, complete the following steps:

1. Go to the following website:

<http://www.redbooks.ibm.com/abstracts/sg247835.html?Open>

2. Click the **Additional Material** link.
3. Click the sg247835.zip file, and select **Save** to save the compressed file to your computer.
4. Extract the contents of the compressed file. You will have two directories called 7835code and 7835codesolution.

21.2.2 Importing the application to the development tool

If you navigate to the 7835codesolution directory, you notice a number of directories. For our discussion in this section, we use the ejb directory. This directory contains two compressed files that include Rational Application Developer project interchange files:

- ▶ RAD8EJB.zip
- ▶ RAD8EJBWeb.zip

To use the sample application for our exercise, import both files into IBM Assembly and Deploy Tools for WebSphere Administration by completing the following steps:

1. Start IBM Assembly and Deploy Tools for WebSphere Administration.
2. To import the code, click **File → Import** and then expand the **General** section. Select **Existing projects into workspace**. Click **Next**.
3. Click **Browse** next to the “Select Archive file” field, and browse to the ejb directory where you extracted the sample code. Select the RAD8EJB.zip file, and click **Open**.
4. Click **Select All** to select all projects in the file. Click **Finish**.
5. You may see a pop-up window during the import process that says the project needs to be migrated. If so, click **Next** until the migration has been completed.
6. Repeat the process for the RAD8EJBWeb.zip project.

If you see errors in the Markers view:

- ▶ Make sure you have the WebSphere Application Server V8 runtime environment installed and an application server defined when you import the application.
- ▶ If you have a server defined, but see errors that indicate a build path error, right-click each module where the error occurs and click **Properties → Java Build Path**. Click the **Libraries** tab. Click **JRE System Library (WebSphere Application Server v8.0 JRE)** and then click **Edit**. Select **Alternate JRE**, then in the drop-down menu, select **WebSphere Application Server JRE**. Click **Finish**, and then **OK**.
- ▶ If you have errors in the persistence.xml file, select **RAD8JPA/src/META-INF/persistence.xml** in the Enterprise Explorer view. Right-click and select **JPA Tools → Synchronize Class List**.
- ▶ If you see an error in the ejb-jar.xml file of the RAD8EJB project, you can ignore it. That file will be deleted in the next section.

21.2.3 Customizing the sample application

You can use the ITSO Bank application without customizing it. The team that created the application did all the necessary development for the application to work. However, to illustrate common packaging tasks and the functionality of the IBM Assembly and Deploy Tools for WebSphere Administration development tool, customize the application for this example by completing the following steps:

1. Remove the unnecessary deployment descriptors that were included in the application by the development team as follows:
 - a. Expand the **RAD8EJB** project, and expand the ejbModule folder. Then, expand the META-INF folder, and delete the ejb-jar.xml file.
 - b. Expand the **RAD8EJBWebEAR** project, and expand its META-INF folder. Delete the application.xml file.

- c. Expand the **RAD8JPA** project, and expand the `src` folder. Then, expand the `META-INF` folder, and delete the `orm.xml` file.
2. The RAD8EJB project depends on the Persistence Unit defined in the RAD8JPA project. To verify that this dependency is set up correctly, use the Deployment Assembly properties sheet (which replaces the J2EE Module Dependencies properties sheet used in previous versions of Rational Application Developer). The integrated development environment (IDE) updates the `application.xml` file automatically, if one exists.
To access the Deployment Assembly properties sheet, right-click the **RAD8EJB** project and select **Properties**. Then, click **Deployment Assembly** in the left pane.
3. In the EAR Module Assembly window, the list of projects included in the EAR should look like Figure 21-6. All three projects, including the `RAD8JPA.jar` project, should be in the list.

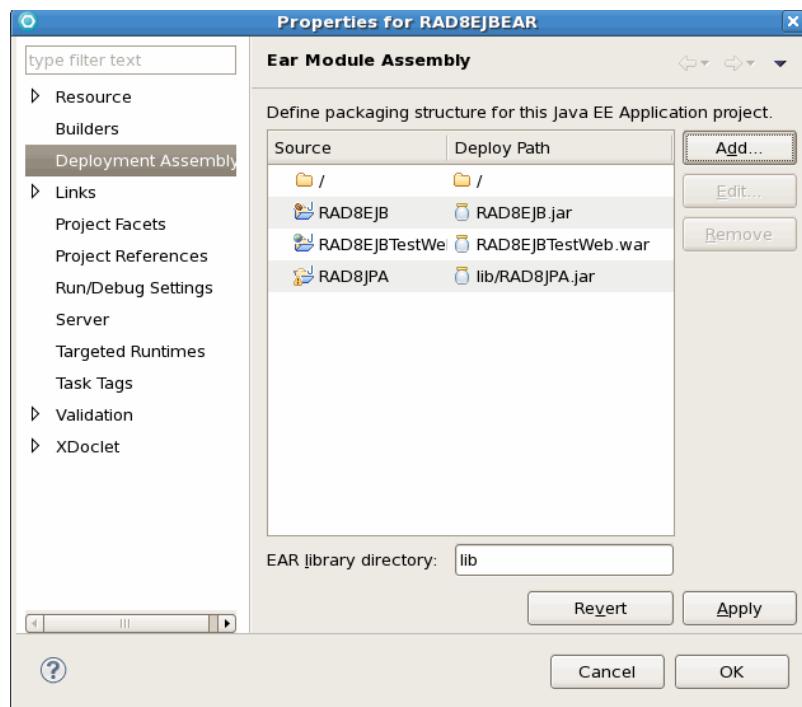


Figure 21-6 Deployment Assembly

If all three packages shown in Figure 21-6 are not listed in your properties:

- a. Click **Add**.
- b. In the New Assembly directive wizard, in the Select Directive Type window, click **Project** and click **Next**.
- c. In the New Assembly directive wizard, in the Project window, select the missing module, for example, **RAD8JPA** and click **Finish**.

Otherwise, click **Cancel**.

21.2.4 Creating the ITSO Bank DB2 database

If you plan to deploy and test this application as described in this chapter, you need to create the database on a DB2 system.

In the extracted files for the sample application, locate the 7835code folder. The database directory in this compressed file contains scripts that we use to prepare the database for the application.

If you are working on a pre-Java EE 5 application or are using EJB 2.1 or earlier modules, also refer to *WebSphere Application Server V6.1: System Management and Configuration*, SG24-7304.

To set up the DB2 database, make sure that DB2 installed and running. Then, complete the following steps:

1. Open a command prompt.
2. Change to the database/db2 folder in the folder that was created when you extracted the additional material.
3. Execute the createbank.bat file to define the database and table.
4. Execute the loadbank.bat file to delete the existing data and add records.
5. Execute the listbank.bat file to list the contents of the database.

Each command opens a new window where the DB2 script executes. Each command also leaves a connection to the database open, so you might want to execute a `db2 connect reset` command in each window opened to disconnect from the database so no unused connections are kept open.

21.3 Configuring web module extensions

To prepare an application for deployment, you want to review and possibly customize the WebSphere web module extensions. In this example, the serve servlets by class name and directory browsing options will be disabled. It is a best practice to disable these options in production environments so that only the servlets and folders that the developers intended to be accessible are accessible.

To configure the webs module extensions, complete the following steps:

1. Expand the **RAD8EJBWeb** project, and double-click the **RAD8EJBWeb** heading (Figure 21-7) to open the web module deployment descriptor.

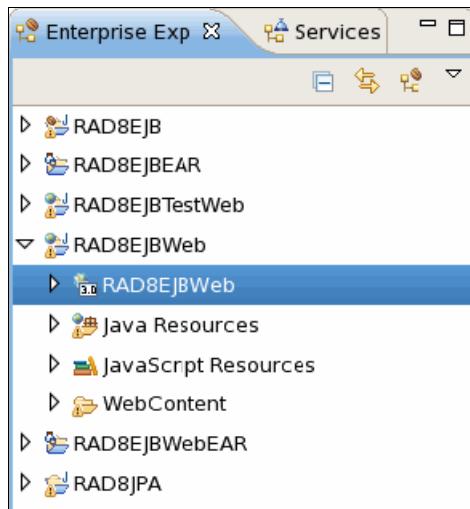


Figure 21-7 Opening the web module deployment descriptor

- When the window opens, click the **Open WebSphere Extensions Descriptor** link in the bottom right corner, as shown in Figure 21-8, to open the extensions editor.

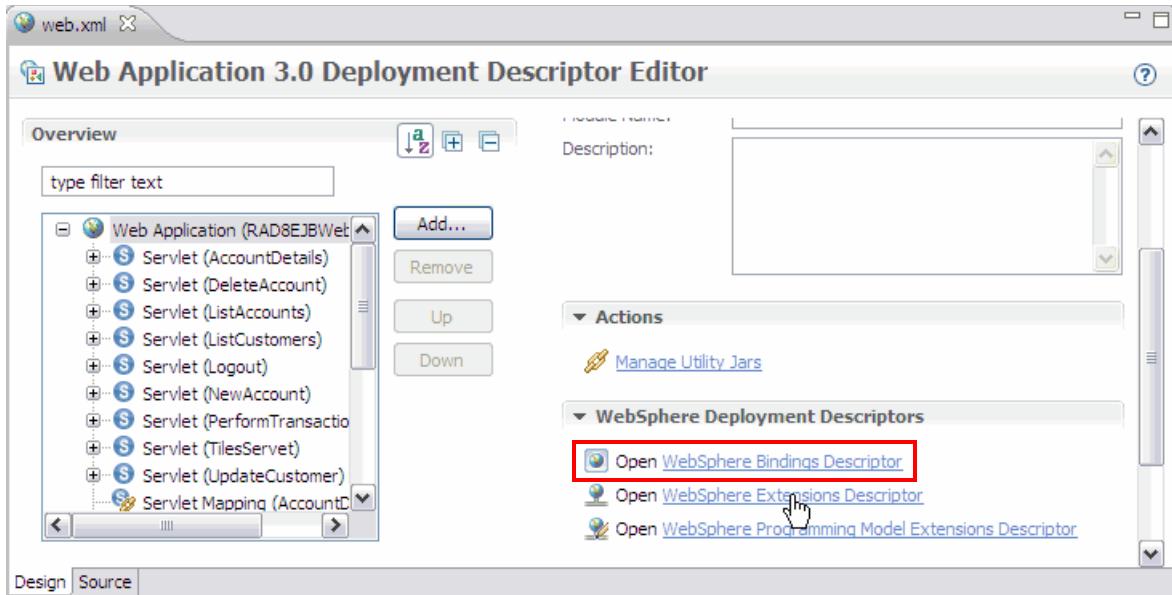


Figure 21-8 Web module deployment descriptor editor

- The web module extensions editor contains options to configure the optional WebSphere extensions to web modules. In our application, we clear the **Enable Directory Browsing** and **Enable Serving Servlets By Class Names** options, as shown in Figure 21-9.

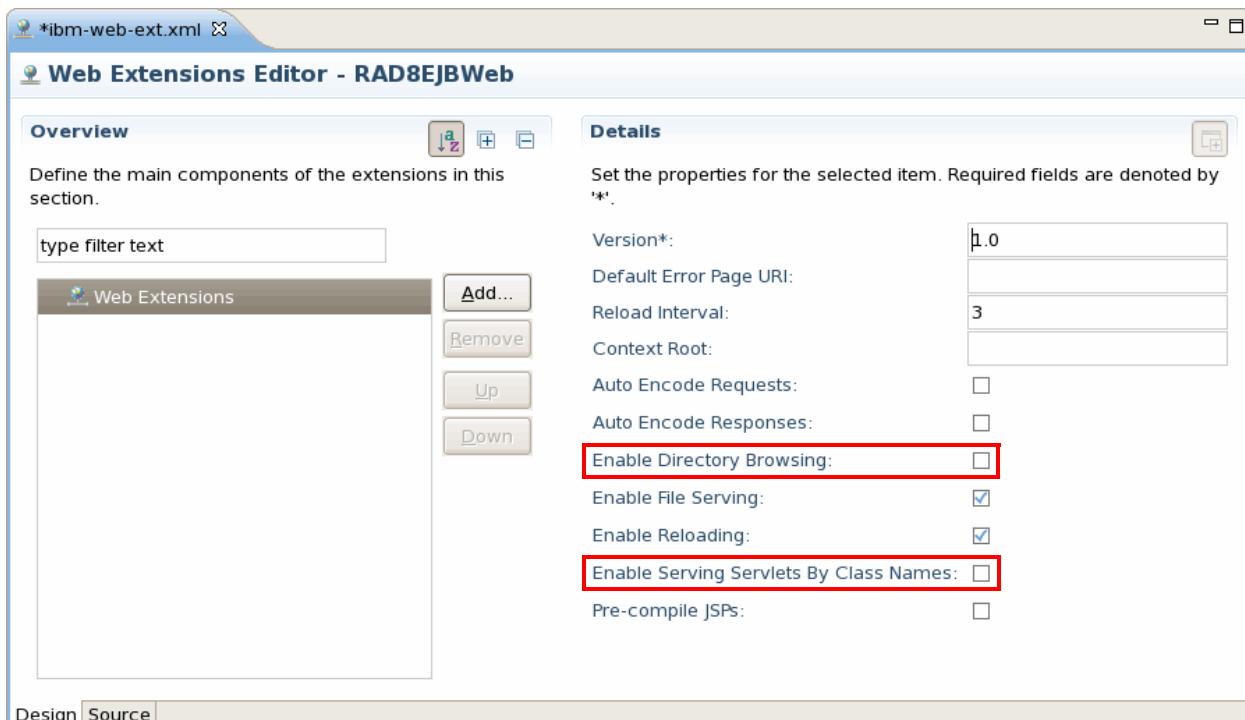


Figure 21-9 Web module extensions editor

- When finished, press Ctrl+s to save the deployment descriptor and then close both the extensions file (ibm-web-ext.xml) and the web.xml file in the editor window.

21.4 Packaging recommendations

Consider the following basic rules when packaging an enterprise application:

- ▶ Package EJB JAR modules and web WAR modules that make up an application together in the same EAR module and execute them within the same application server JVM process. This configuration avoids remote EJB calls (RMI/IOP) across application server JVM processes, which is costly from a performance perspective.
- ▶ Place utility classes that are used by a single web module only within the web module's WEB-INF/lib folder.
- ▶ Place utility classes that are used by multiple modules within an enterprise application at the root of the EAR file as Utility Projects so that they are accessible by both servlets and EJB.
- ▶ Place utility classes that are used by multiple enterprise applications outside the applications on a directory that is referenced through a shared library definition.
- ▶ Keep the class path clean and reference only required libraries for your application.

21.5 Creating WebSphere Enhanced EAR files

A WebSphere Enhanced EAR file is a regular Java EE EAR file but with additional configuration information for resources required by Java EE applications. Although adding this extra configuration information at packaging time is not mandatory, it can simplify deployment of Java EE applications to multiple run times if the environments are similar.

If you plan to provide the configuration of these resources by using the runtime environment administrative tools, you can skip this section and go directly to 21.6, “Exporting an application project to an EAR file” on page 804.

When an Enhanced EAR is deployed to WebSphere Application Server, the resources specified in the Enhanced EAR are automatically configured. When an Enhanced EAR is uninstalled, the resources that are defined at the application level scope are removed as well. However, resources that are defined at a scope other than application level are not removed because they might be in use by other applications. Resources that are created at the application level scope are limited in visibility to only that application.

Table 21-7 shows the resources that are supported by the Enhanced EAR and the scope in which they are created.

Table 21-7 Scope for resources in WebSphere Enhanced EAR file

Resource	Scope
JDBC providers	Application
Data sources	Application
Resource adapters	Application
JMS resources	Application
Substitution variables	Application
Class loader policies	Application
Shared libraries	Server

Resource	Scope
JAAS authentication aliases	Cell
Virtual hosts	Cell

J2C Resource Adapters can be configured either as embedded or external resources. An embedded RAR is packaged within an EAR file, is deployed as a part of Java EE application installation, and is removed when the application is uninstalled from the server. An external RAR file is packaged as a stand-alone RAR file, is deployed explicitly on a WebSphere node, and is not managed as a Java EE application. If an adapter is used only by a single application, configure it as an embedded RAR file. If it is to be shared between multiple applications, configure it as an external RAR file.

To view the application scoped resources using the administrative console, click **Applications** → **Application Types** → **WebSphere Enterprise Applications** → <*application*>. Select **Application scoped resources** in the References section. If there are no application scoped resources, you do not see this option.

21.5.1 Configuring a WebSphere Enhanced EAR

You can modify the supplemental information in an Enhanced EAR using the WebSphere Application Server Deployment editor of IBM Assembly and Deploy Tools for WebSphere Administration. The deployment information is contained in XML files in a folder called `ibmconfig` in the EAR file's `META-INF` folder.

In the sample application, the provider is changed to use DB2. To make this change, the following configuration items will be added to the deployment file:

- ▶ JAAS authentication alias
- ▶ JDBC provider for DB2
- ▶ Data source for DB2 database

A new virtual host for a domain called `www.itsobank.ibm.com` is also added.

To access the Enhanced EAR deployment options, right-click the **RAD8EJBWebEAR** project, and select **Java EE**. Then, click **Open WebSphere Application Server Deployment** to open the editor, as shown in Figure 21-10.



Figure 21-10 WebSphere Enhanced EAR editor

21.5.2 Configuring application options

The Application section shown in Figure 21-11 contains the class loader policies and class loader mode configured for each of the containing modules. ITSO Bank runs with the default policies and modes. You do not need to change them. The auto start feature is new with WebSphere Application Server V8. When set to Yes, the application will start at the application server start.

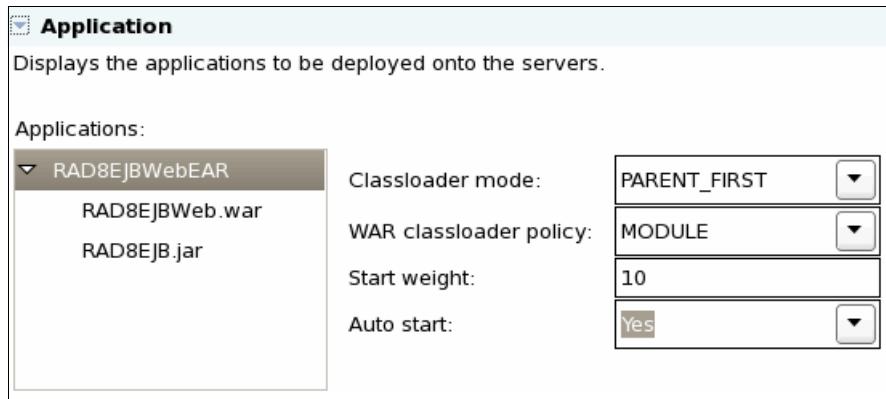


Figure 21-11 Configuring class loader mode and class loader policies

Configuring a JAAS authentication alias

To configure the JAAS authentication alias required to access the application database, complete the following steps:

1. Expand the **Authentication** section.
2. Click **Add**.
3. In the window that opens, enter the following information:
 - An alias of **itsobank**
 - A user ID with access to the **ITSOBANK** database (**db2inst1** in our case)
 - The password for the user ID
 - A description of **ITSO**
4. Click **OK**. Figure 21-12 shows the results.



Figure 21-12 Configuring JAAS authentication alias for ITSO Bank

21.5.3 Configuring the JDBC provider and data source for DB2

To configure JDBC providers, expand the **Data Sources** section. When creating an Enhanced EAR file, IBM Assembly and Deploy Tools for WebSphere Administration automatically adds a JDBC provider for the Derby database. Before you add the DB2 provider, delete the pre-configured Derby JDBC Provider (XA) provider by selecting it and by clicking **Remove**.

To configure the DB2 JDBC provider, complete the following steps:

1. Click **Add** next to the JDBC provider list.
 2. In the window that opens, select the following options (as shown in Figure 21-13):
 - **IBM DB2** as the Database type
 - **DB2 Using IBM JCC Driver (XA)** as the JDBC provider type
- Click **Next**.

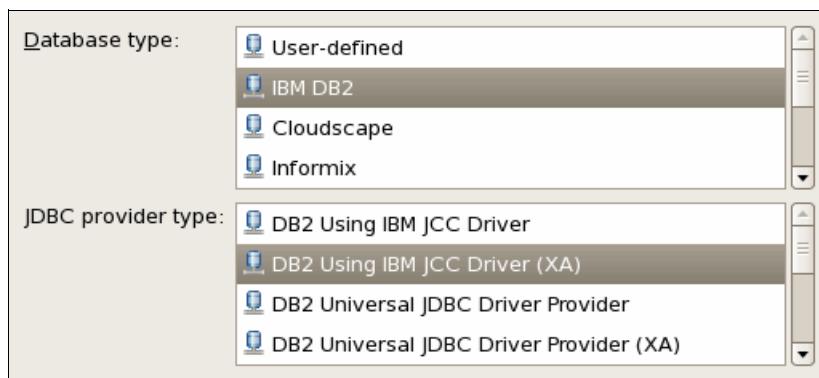


Figure 21-13 Creating a DB2 JDBC provider

3. In the next window, enter a name for the JDBC provider (for administration purposes only), and leave the other properties as the default values. See Figure 21-14. Click **Finish**.

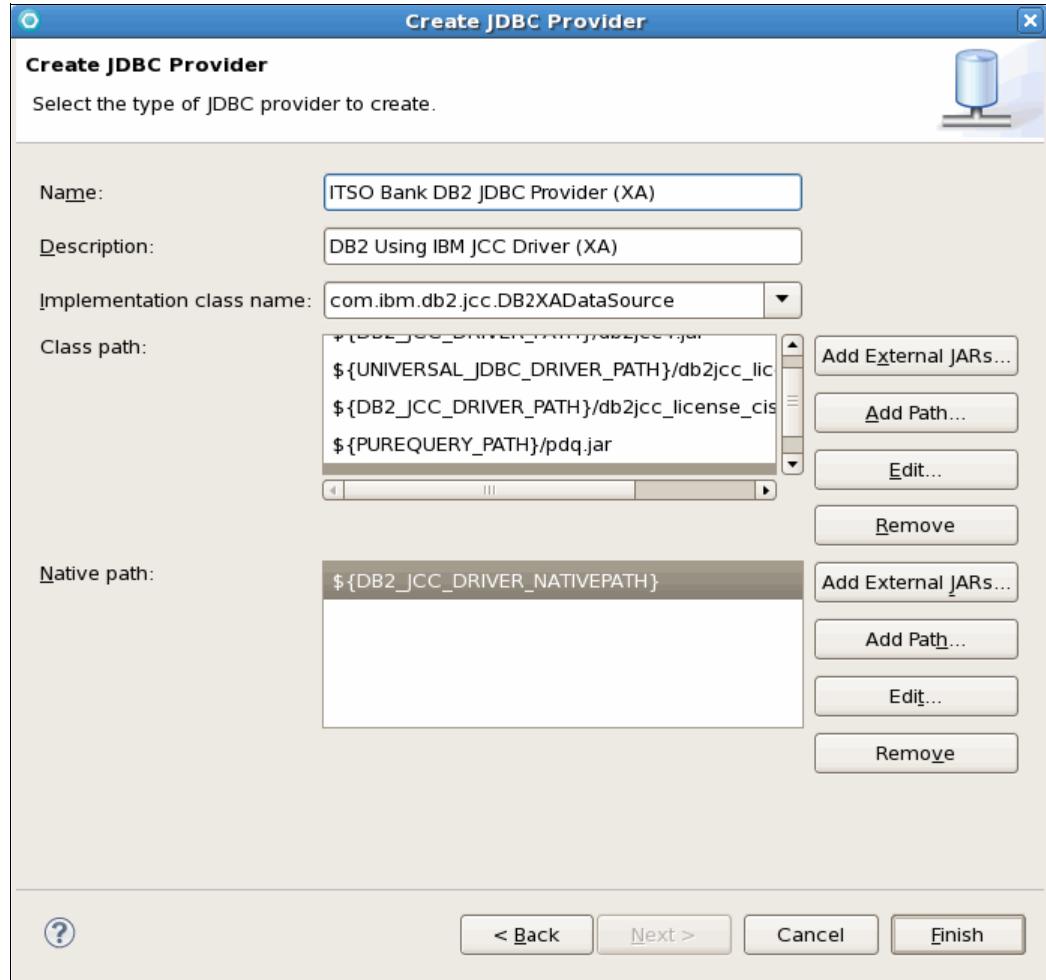


Figure 21-14 Creating a DB2 JDBC provider

- Click the **ITSO Bank DB2 JDBC Provider (XA)** that you just created, and click **Add** next to the Data source list, as shown in Figure 21-15.

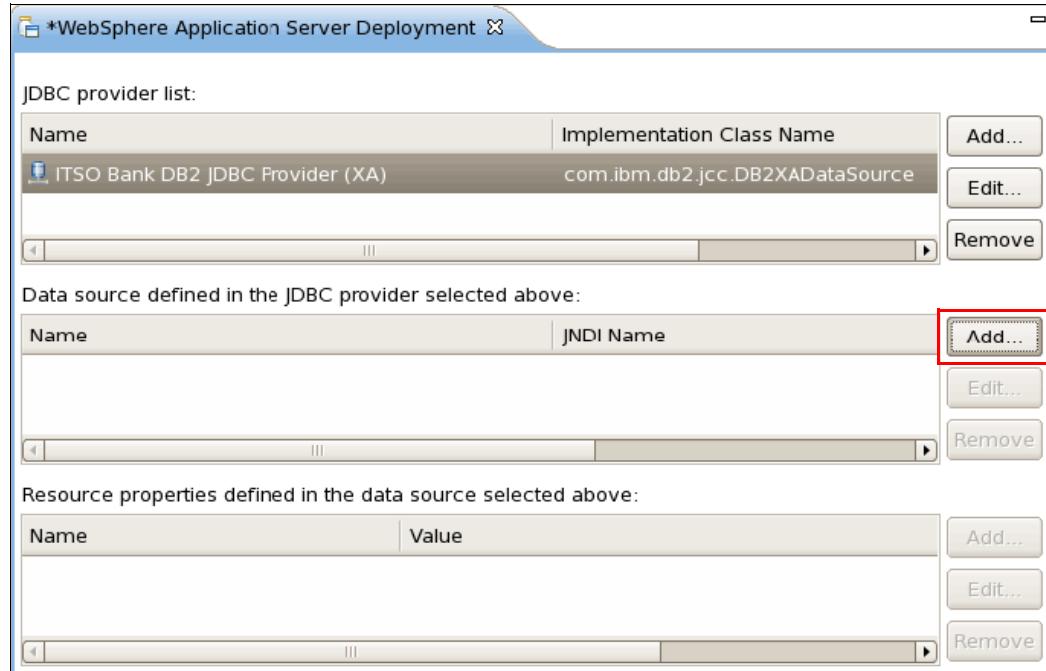


Figure 21-15 Creating a DB2 data source

- In the Create a Data Source window, select **DB2 Using IBM JCC Driver (XA)** as the JDBC provider type and **Version 5.0 data source** as the data source type, as shown in Figure 21-16. Click **Next**.

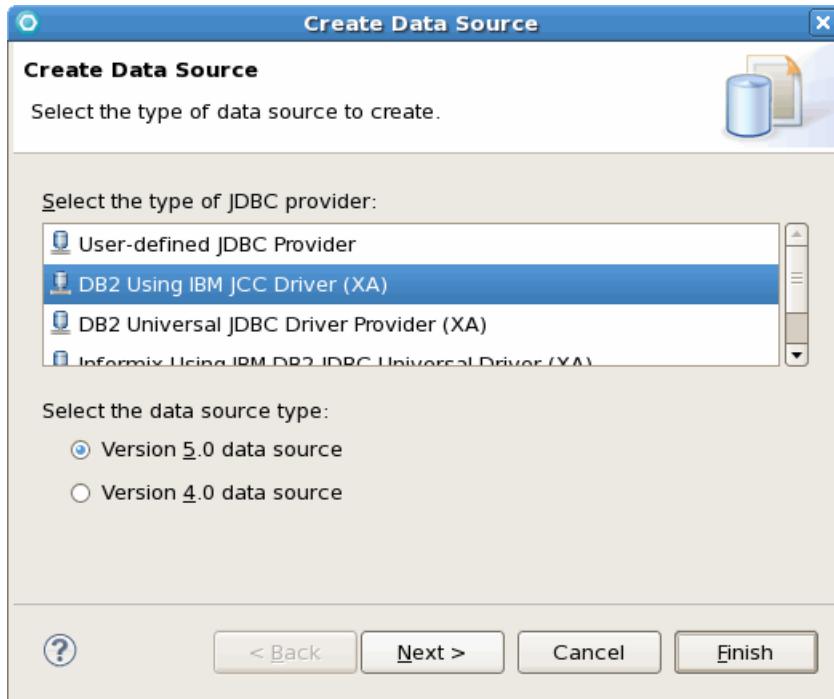


Figure 21-16 Creating a DB2 data source

- In the window that opens, enter the appropriate values for the DB2 data source, as shown in Figure 21-17.
 - Enter ITSOBankDS as the name.
 - Enter jdbc/itsobank as the JNDI name.
 - Enter DB2 Data Source for ITSO Bank as the description.
 - Select **itsobank** as the container-managed authentication alias (you might need to scroll the window to the far right to see the drop-down menu).
 - Clear the **Use this data source in container manager persistence (CMP)** option. The ITSO Bank application uses JPA for persistence, so you do not need to add support for CMP Entity beans for this data source.

Click **Next**.

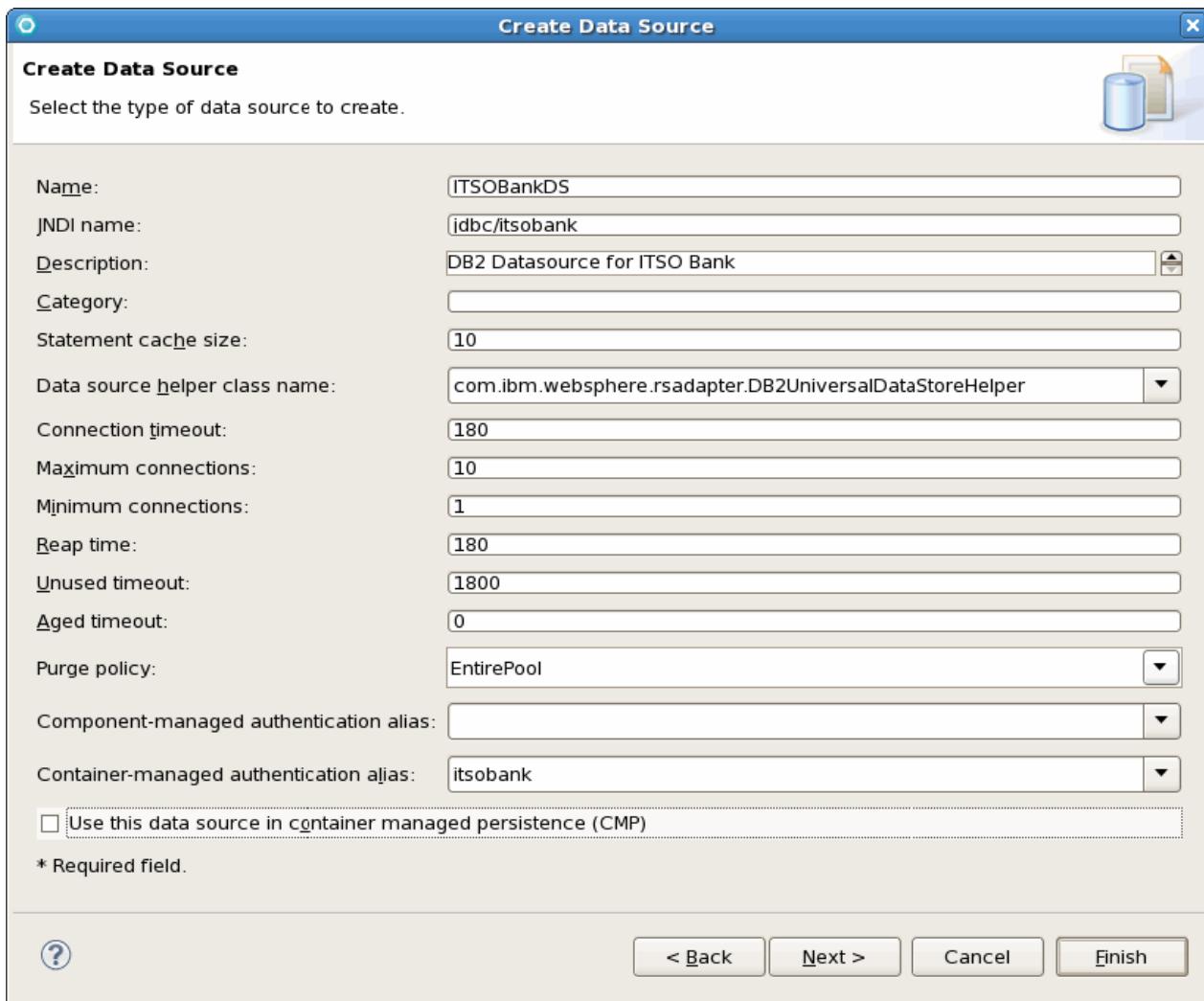


Figure 21-17 Creating a DB2 data source

- Select the **databaseName** property in the Resource properties section. Enter ITSOBANK in the Value field. Then select the **driverType** property, and change the value from type 4 to type 2.

Type 2 means that the DB2 database is installed on the same machine as WebSphere Application Server, or that the DB2 Client software is installed on the same machine and configured to handle the remote connection if the database is on a remote machine. A type 4 driver can connect directly to a remote database over TCP/IP. See Figure 21-18.

Click **Finish**.

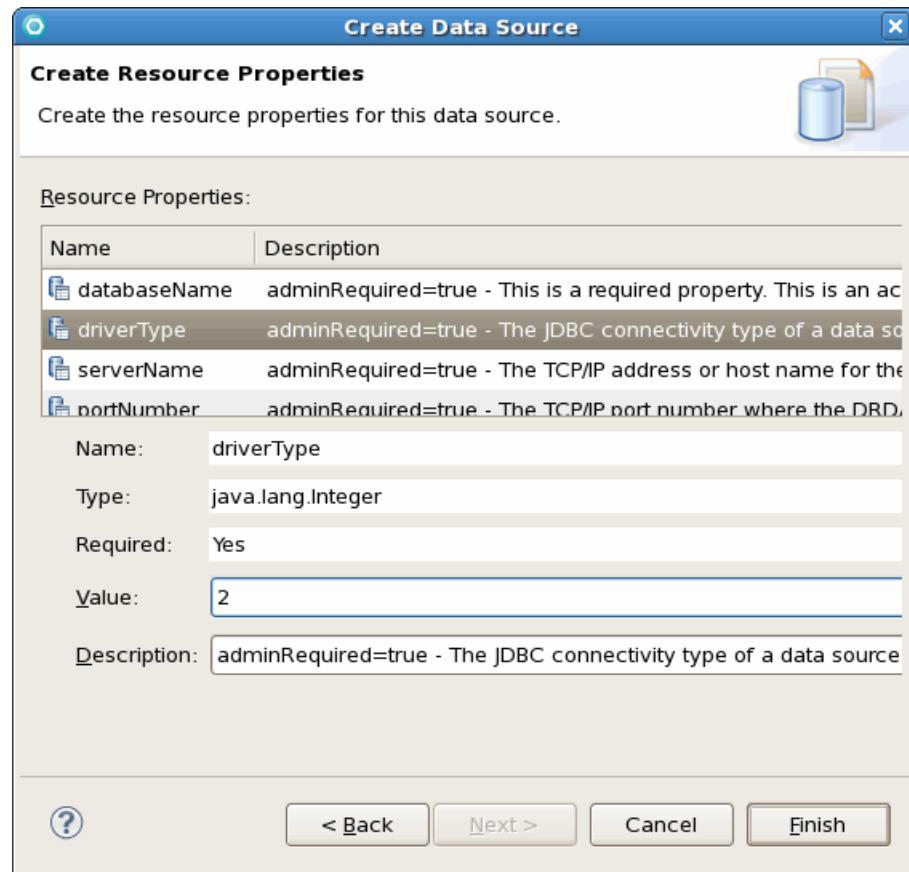


Figure 21-18 Setting database name and driver type

When you are finished, your data source configuration should look similar to the window shown in Figure 21-19.

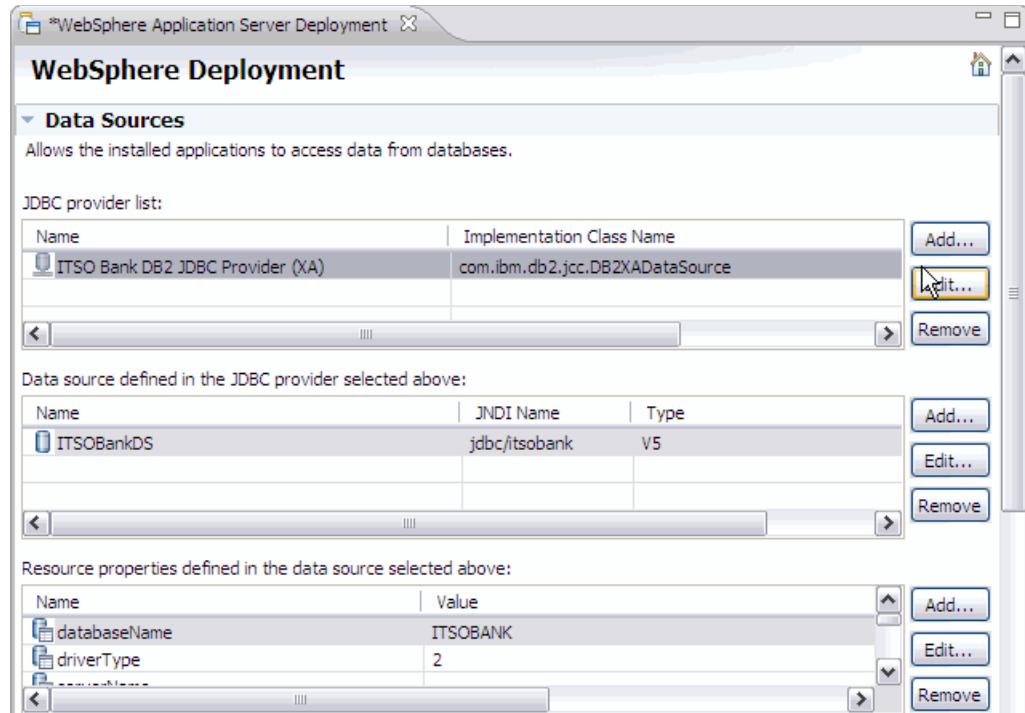


Figure 21-19 DB2 data source configured

21.5.4 Configuring a virtual host

To configure a virtual host, complete the following steps:

1. Expand the **Virtual Hosts** section of the Deployment tab, and click **Add** next to the Virtual host name list.
2. In the Add Host Name Entry dialog box, enter `itsobank_host`, and click **OK**. Your new virtual host displays in the Virtual Hosts list, as shown in Figure 21-20.



Figure 21-20 Add a new virtual host

3. Click **Add** next to the Host aliases list.
4. In the Add Host Alias Entry dialog box, enter `www.itsobank.ibm.com` for the host name and 80 for the port number. Click **OK**.

Repeat the procedure, and add number 9080 as well. You will use this port when you deploy the application later. If your server uses another port, use that port number instead. You can have as many host aliases as you want.

The host aliases appear in the list (Figure 21-21).

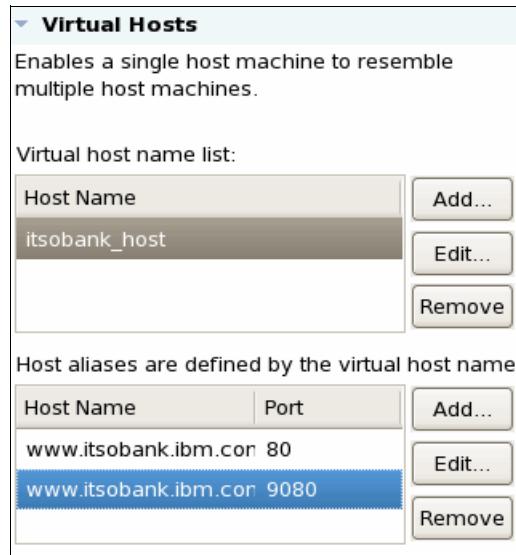


Figure 21-21 Host aliases

- When you are finished, press Ctrl+s to save the deployment descriptor editor.

21.5.5 Setting the default virtual host for web modules

Although you configured a new virtual host, itsobank_host, in the Enhanced EAR file, the web modules do not use it automatically. The default virtual host for a web module that is created in Rational Application Developer or IBM Assembly and Deploy Tools for WebSphere Administration is default_host, which is also the case for the ITSO Bank application.

To modify the web modules to use itsobank_host instead, complete the following steps:

- Expand the **RAD8EJBWeb** project, and double-click the **RAD8EJBWeb** heading, as shown in Figure 21-7 on page 791, to open the web module deployment descriptor.
- In the lower right corner of the window, click the **Open WebSphere Bindings Descriptor** link.
- Change the Virtual Host Name to itsobank_host, as shown in Figure 21-22.



Figure 21-22 Setting the default virtual host for a web module

- Save the deployment descriptor by pressing Ctrl+s, and then close it.

21.5.6 Examining the WebSphere Enhanced EAR file

The information about the configured resources is stored in the `ibmconfig` subdirectory of the EAR file's `META-INF` directory. Expanding this directory reveals the well-known directory structure for a cell configuration, as shown in Figure 21-23. You can also see the scope level where each resource is configured.

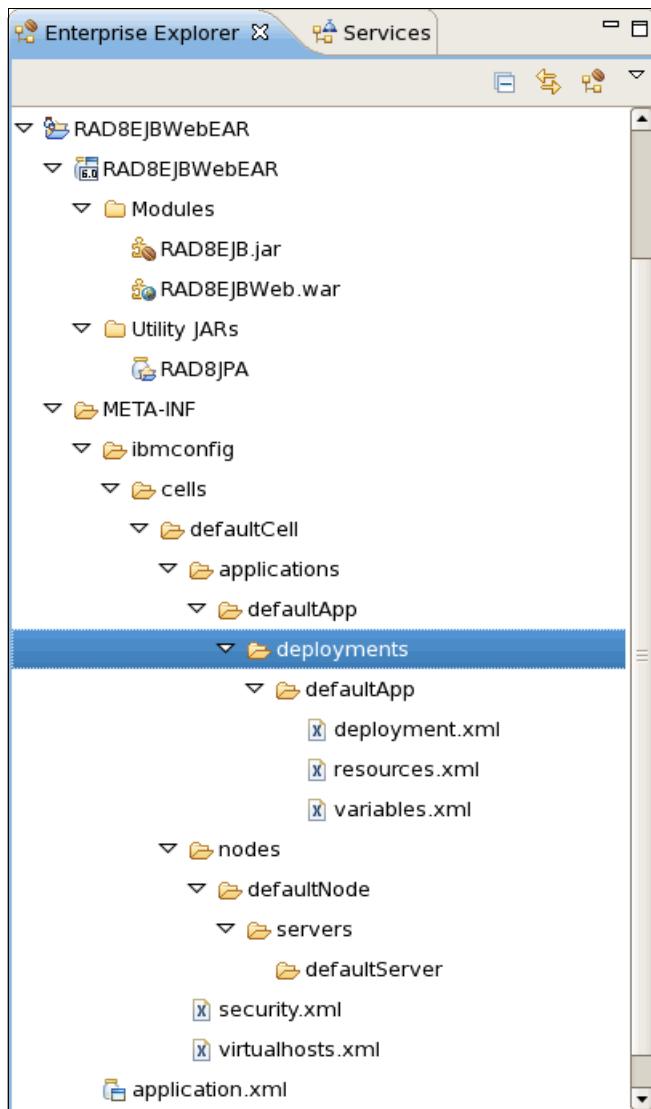


Figure 21-23 Enhanced EAR file contents

At deployment time, WebSphere Application Server uses this information to create the resources automatically.

After you add the additional configuration information to the application, export the project as an EAR file for installation in WebSphere Application Server V8.

21.6 Exporting an application project to an EAR file

To deploy an application, package it as an EAR file. To export the RAD8EJBWebEAR application as an EAR file with all its dependent modules, complete the following steps:

1. Select the **RAD8EJBWebEAR** project, and right-click **Export → EAR file** from the pop-up menu.
2. In the Export dialog box, browse to a destination, as shown in Figure 21-24.

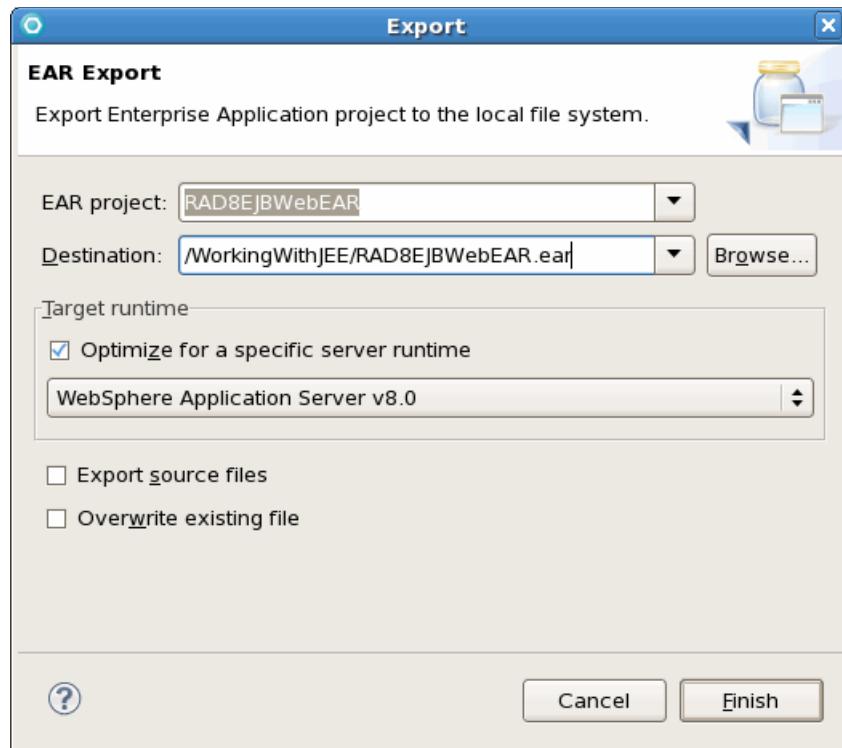


Figure 21-24 Exporting to an EAR file

3. Click **Finish**.
4. The EAR file that you export is now prepared for installation in WebSphere Application Server V8.

Version note: If you select the **Optimize for a specific server runtime** option, the application runs only on the version that you specify or on later versions. In this case, for example, the application will not run on WebSphere Application Server V7 if you select this option.

21.7 Preparing the runtime environment for the application

In this section, we show you how to set up a complete environment for the ITSO Bank application and how to deploy the EAR file. You will not always need or want to customize the environment as extensively as we do in this example. Some steps are optional. If all you want to do is deploy your application quickly, using the WebSphere defaults for directory names, log files, and so on, skip to 21.8, “Deploying the application” on page 818.

To deploy the ITSO Bank application, complete the following steps:

1. Optional: Create an environment variable for ITSO Bank server.
2. Optional: Create an application server to host the application.
3. Optional: Customize the IBM HTTP Server configuration.
4. Define a JDBC provider, data source, and authentication alias. If you are using an Enhanced EAR file, this step is optional.
5. Define virtual hosts. If you are using an Enhanced EAR file, this step is optional.

If the application to be deployed is a WebSphere Enhanced EAR file, the resources configured in the Enhanced EAR file are created automatically when the application is deployed. The resources that can be configured in the enhanced EAR are described in the following sections.

21.7.1 Creating an environment variable for the application file directory

Use WebSphere environment variables, rather than hardcoded paths, when deploying an application. In this step, a variable called `ITSOBANK_ROOT` is defined. The variable represents the root directory where application resources will be stored. You use this variable in subsequent configuration processes when specifying, for example, the JVM log location.

Complete the following steps:

1. Determine how you want to organize your application files.

There are several ways to organize WebSphere applications. You can create a directory for each application, such as `/apps/application_name`, and keep all resources and directories that are required by the application in subdirectories under this directory. This strategy works well when deploying only one application per application server because the application server’s log files can then all be changed to point to the `/apps/application_name/logs` directory.

An alternative is to organize resources by resource type and create directories such as `/apps/logs/application_name.log`, `/apps/properties/application_name.properties`, and so on.

Finally, you can keep the vendor defaults as much as possible. For WebSphere, vendor defaults mean that the applications are installed in the `profile_root/installedApps` directory and that the logs files are written to the `profile_root/logs/server_name` directory.

The option that you choose is a matter of personal preferences and corporate guidelines.

In this example, we deploy a single application to a single application server. A directory called `/apps/ITSOBANK` will be used to contain the application resources.

2. Create the target directory on the application server operating system:

`/apps/ITSOBANK`

Important: If you define and use the variable in configuration steps, but the directory is not created, the application server may not start.

3. Use the steps that are described in 6.1.10, “Using variables” on page 244 to create a ITSOBANK_ROOT variable with a value of /apps/ITSOBANK.

Be certain that you declare this variable at the correct scope. For example, if you define this variable at the application server scope, it is known only at that level. As long as you work with the WebSphere Application Server Base or Express editions, this configuration is fine. However, if you later decide to use the Network Deployment edition and you create a cluster of application servers, you need to define the ITSOBANK_ROOT variable at the cluster or cell level.

21.7.2 Creating the ITSO Bank application server

In a distributed server environment, you have the option of using a single application server or creating multiple application servers or clusters.

The advantage of deploying multiple applications to a single application server is that it consumes less resources. There is no impact for any extra application server processes. Another benefit is that applications can make in-process calls to each other. For example, servlets in one EAR file could access the local interfaces of the EJB beans in another EAR file.

One alternative to using a single application server is to deploy each application to its own server. The advantages of deploying only one application on an application server is that it gives you greater control over the environment. The JVM heap sizes and environment variables are set at the application server level. Thus, all applications running in an application server share the JVM memory given to the application server and all see the same environment variables. Running each application in its own application server can also make it easier to perform problem determination. For example, if an application consumes a lot of CPU resources, you can determine which application is consuming CPU resources by looking at the process ID of the application server.

In our example, we have a unique application server on which we run the ITSO Bank sample application. Use the instructions in 7.4.1, “Creating an application server” on page 272 to create a new application server called ITSOBankServer1.

Changing the application server working directory

This step will change the working directory for the application server process. This directory is the relative root for searching files. For example, if you perform a `File.open("filename.gif")` command, filename.gif must be present in the working directory. This directory is created by WebSphere if it does not exist. Create a specific working directory for each application server.

To change the working directory, complete the following steps:

1. Create the working directory on the operating system:

```
/apps/ITSOBANK/workingDir
```

Important: The working directory is not created automatically. Create the path before starting the application server or the start sequence fails.

2. In the administrative console, click **Servers** → **Server Types** → **WebSphere Application Servers**. Click the **ITSOBankServer1** server.
3. Expand the **Java and Process Management** in the Server Infrastructure section and select **Process Definition**.
4. Scroll down the window and change the working directory from \${USER_INSTALL_ROOT} to \${ITSOBANK_ROOT}/workingDir.
5. Click **OK**.

Changing the application server logging and tracing options

Next, customize the logging and tracing properties for the application server. The properties will be updated so the logs are stored in the *ITSOBANK_ROOT/logs* directory.

Complete the following steps:

1. Create the /apps/ITSOBANK/logs directory on the operating system.
2. Update the logging and tracing properties to use this directory.

There are several ways to access the logging and tracing properties for an application server in the administrative console:

- Click **Troubleshooting** → **Logs and Trace** in the navigation bar, and then select a server.
- Click **Servers** → **Server Types** → **WebSphere application servers**, select a server, and then select **Logging and Tracing** from the Troubleshooting section.
- Click **Servers** → **Server Types** → **WebSphere application servers**, select a server, and then select **Process definition** from the Java and Process Management section.

Because you just updated the application server process definition, you can use the third navigation path to customize the location of the JVM logs, the diagnostic trace logs, and the process logs. Then, select **Logging and Tracing** from the Additional Properties section. For this example, we use basic logging. However, you might want to use High Performance Extensible Logging (HPEL). For more information about HPEL, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.zseries.doc/info/zseries/ae/ctrb_HPELOverview.html

3. Select **JVM Logs**.

This option allows you to change the JVM standard output and error file properties. Both the JVM standard output and the error file properties are rotating files. You can choose to save the current file and create a new one, either when it reaches a certain size or at a specific moment during the day. You can also choose to disable the output of calls to `System.out.print()` or `System.err.print()`.

Specify the new file name, using an environment variable:

```
 ${ITSOBANK_ROOT}/logs/SystemOut.log  
 ${ITSOBANK_ROOT}/logs/SystemErr.log
```

On this window, you can also modify how WebSphere rotates log files.

Click **OK**.

4. Select **Diagnostic Trace**.

Each component of the WebSphere Application Server is enabled for tracing with the JRsas interface. This trace can be changed dynamically while the process is running using the Runtime tab or can be added to the application server definition from the Configuration tab. As shown in Figure 21-25, the trace output can be directed either to memory or to a rotating trace file.

Change the trace output file name so that the trace is stored in a specific location for the server using the ITSOBANK_ROOT variable:

```
 ${ITSOBANK_ROOT}/logs/trace.log
```

Then select the **Basic** format.

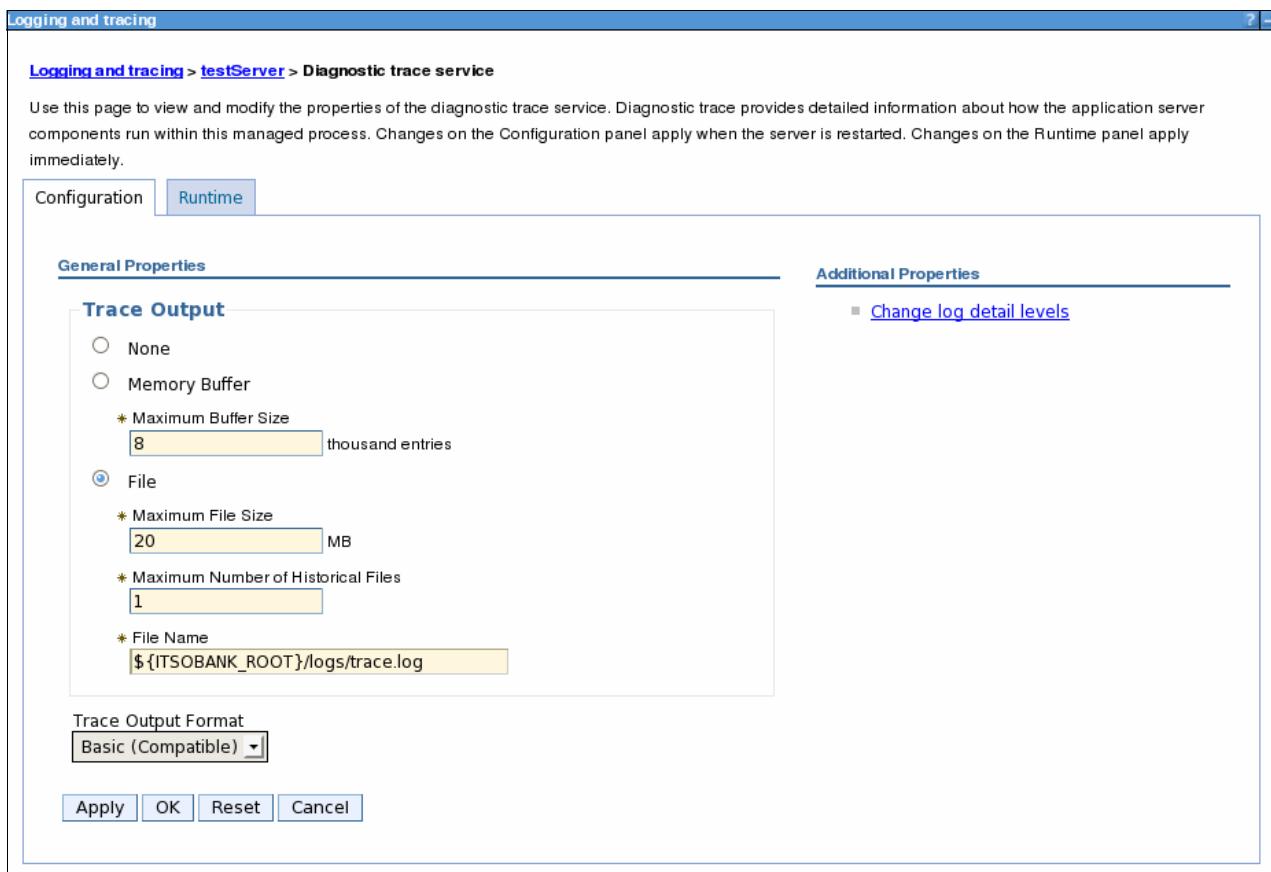


Figure 21-25 Specifying diagnostic trace service options

Click **OK**.

5. Select **Process Logs**.

Messages written by native code (JNI) to standard out and standard error streams are redirected by WebSphere to process logs, usually called native_stdout.log and native_stderr.log. Change the native process logs to use the following paths:

```
 ${ITSOBANK_ROOT}/logs/native_stdout.log  
 ${ITSOBANK_ROOT}/logs/native_stderr.log
```

Click **OK**.

6. All log files that are produced by the application server are now redirected to the \${ITSOBANK_ROOT}/logs directory. Save the configuration.

Verify the port number: The remainder of this example assumes a default HTTP port of 9080 for the web container. Before you continue, check the application server you created to determine the port that you should use by completing the following steps:

1. Click **Servers** → **Server Types** → **WebSphere application servers**.
2. Select **ITSOBankServer1**.
3. Select **Ports** in the Communications section.
4. Scroll down the window, and then note the port listed for WC_defaulthost.

21.7.3 Defining the ITSO Bank virtual host

Enhanced EAR file users: If you are using an Enhanced EAR file, you can define the virtual host at packaging time. See 21.5.4, “Configuring a virtual host” on page 801.

Web modules need to be bound to a specific virtual host. For our sample, we chose to bind the RAD8EJBWeb web module to a specific virtual host called *itsobank_host*. This virtual host has the following host aliases:

- ▶ www.itsobank.ibm.com:80
- ▶ www.itsobank.ibm.com:9080

Any request starting with *itsobank_host_alias/RAD8EJBWeb*, such as <http://www.itsobank.ibm.com:9080/RAD8EJBWeb>, is served by the RAD8EJBWeb application.

Tip: You can restrict the list of hosts that are used to access the web application by removing hosts from the virtual host definition.

For example, if you want to prevent users from accessing the ITSO Bank application directly from the WebSphere internal HTTP server when they invoke <http://www.itsobank.ibm.com:9080/RAD8EJBWeb>, you can remove www.itsobank.ibm.com:9080 from the virtual host aliases list. Then, all requests go through the web server plug-in.

To create the *itsobank_host* virtual host, complete the following steps:

1. Click **Environment** → **Virtual Hosts** in the navigation pane.
2. Click **New**.
3. Enter the virtual host name, *itsobank_host*, and then click **Apply**.
4. Select **Host Aliases** in the Additional Properties section.

- Add the two aliases shown in Figure 21-26 by clicking **New**, entering the values, and then clicking **OK**.

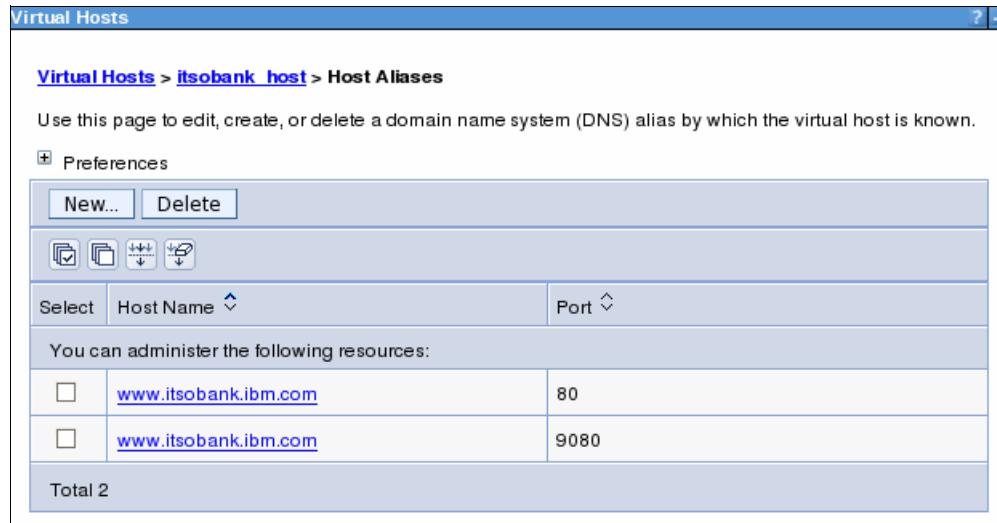


Figure 21-26 WebSphere Bank virtual host aliases

- Click **OK**.
- Save the configuration.

21.7.4 Creating the virtual host for IBM HTTP Server and Apache

Now that you have defined the itsobank_host virtual host, you need to configure the web server to serve the host aliases in the virtual host. The steps that we describe in this section are valid for both the IBM HTTP Server V8 and Apache 2 or later.

Configuring virtual hosting

Tip: You do not need to create a virtual host in the httpd.conf file. A virtual host in the httpd.conf file is required only if you want to customize the configuration, for example, by separating the logs for each virtual host.

You create virtual hosts using the `<VirtualHost>` directive, as shown in Example 21-1.

Example 21-1 Using `<VirtualHost>`

```
<VirtualHost www.itsobank.ibm.com:80>
    ServerAdmin webmaster@itsobank.ibm.com
    ServerName www.itsobank.ibm.com
    DocumentRoot "/opt/IBM/HTTPServer/htdocs/itsobank"
    ErrorLog logs/itsobank_error.log
    TransferLog logs/itsobank_access.log
</VirtualHost>
```

If you want to have multiple virtual hosts for the same IP address, you must use the `NameVirtualHost` directive, as shown in Example 21-2.

Example 21-2 Using the `NameVirtualHost` and `VirtualHost` directives

NameVirtualHost 9.42.171.97:80

```
<VirtualHost itso_server:80>
    ServerAdmin webmaster@itso_server.com
    ServerName itso_server
    DocumentRoot "/opt/IBM/HTTPServer/htdocs/itso_server"
    ErrorLog logs/itso_server_error.log
    TransferLog logs/itso_server_access.log
</VirtualHost>

<VirtualHost www.itsobank.ibm.com:80>
    ServerAdmin webmaster@itsobank.ibm.com
    ServerName www.itsobank.ibm.com
    DocumentRoot "/IBM/HTTPServer/htdocs/itsobank"
    ErrorLog logs/itsobank_error.log
    TransferLog logs/itsobank_access.log
</VirtualHost>
```

The `www.itsobank.ibm.com` and the `itso_server` hosts have the same IP address, 9.42.171.97. We set this address by inserting the following line in the machine hosts file, which is located in `%windir%\system32\drivers\etc` or in `/etc` on UNIX systems:

9.42.171.97 www.itsobank.ibm.com itso_server

In a real-life environment, name resolution is achieved by creating aliases at the DNS level. In any event, you must be able to ping the host that you have defined, using a command such as `ping www.itsobank.ibm.com`.

As shown in Example 21-2, each virtual host has a different document root. Make sure that the directory that you specify exists before you start the HTTP server. While testing the setup, you can place an `index.html` file at the document root stating which virtual host is being called, which lets you determine easily the virtual host that is being used.

You must restart the IBM HTTP Server to apply these changes. If you are running a Windows system, try to start the server by running `apache.exe` from the command line rather than from the Services window. This method allows you to spot error messages that are thrown at server start. In Linux or UNIX servers, you can run `apachectl` with the `-t` flag to check for errors in the `httpd.conf` file.

If your virtual hosts are correctly configured, invoking `http://www.itsobank.ibm.com` or `http://itso_server` returns different HTML pages.

21.7.5 Creating a DB2 JDBC provider and data source

Enhanced EAR file users: If you are using an Enhanced EAR file, you can define the JDBC provider, data source, and J2C authentication entry at packaging time. Refer to 21.5.3, “Configuring the JDBC provider and data source for DB2” on page 796 for more information.

The ITSO Bank sample application uses a relational database, with the Java Persistence API, to store information. To access this database, you need to define a data source with a JNDI name that matches the data source configuration in the JPA module persistence.xml file. Figure 21-27 shows the persistence.xml file for the RAD8JPA project open in the IBM Assembly and Deploy Tools for WebSphere Administration. As you can see, the JNDI name in this case is jdbc/itsobank.

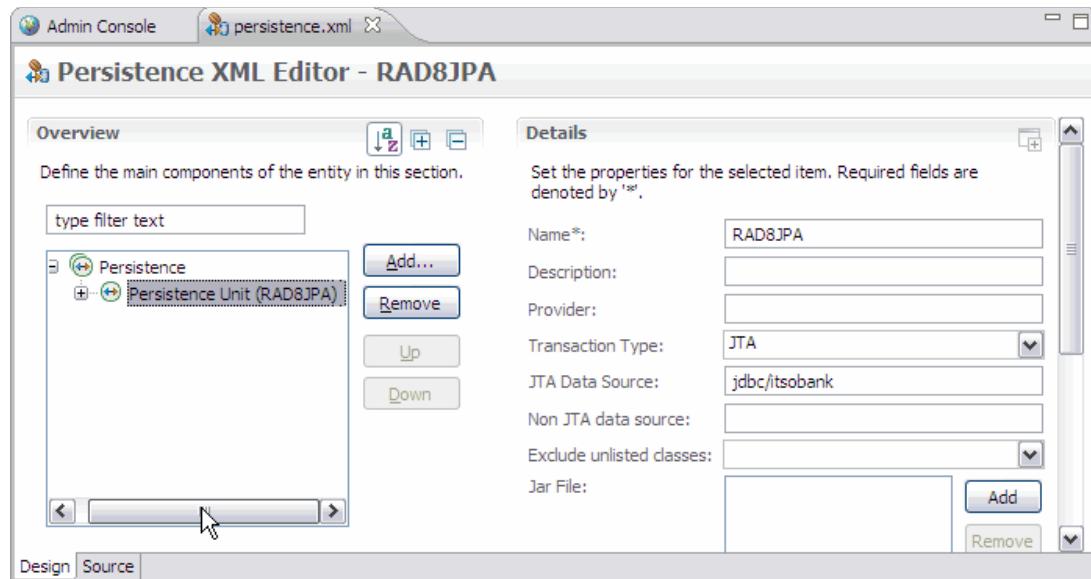


Figure 21-27 Persistence.xml file opened in an editor

The ITSO Bank sample application is configured for Derby by default, but we use a DB2 database with the application. In our example here, we create the DB2 JDBC provider, data source, and Java Authentication and Authorization Service (JAAS) authentication alias that are required to run against DB2.

For detailed information about JDBC providers and data sources, refer to Chapter 9, “Accessing databases from WebSphere” on page 379.

Configuring environment variables for DB2 JDBC driver

For the DB2 JCC JDBC Provider to find its classes, you need to set up the DB2_JCC_DRIVER_PATH and DB2_JCC_DRIVER_NATIVEPATH environment variables. These variables can be defined when you create the JDBC provider (see 9.3.1, “Creating the JDBC provider” on page 385), or you can predefined them as is done in this example.

Complete the following steps:

1. Click **Environment** → **WebSphere Variables**.
2. Locate and click the **DB2_JCC_DRIVER_PATH** entry.

3. In the value field, enter the path where the DB2 JDBC driver is located (Figure 21-28). For example, for DB2, the location is likely to be as follows:

/opt/IBM/SQLLIB/java

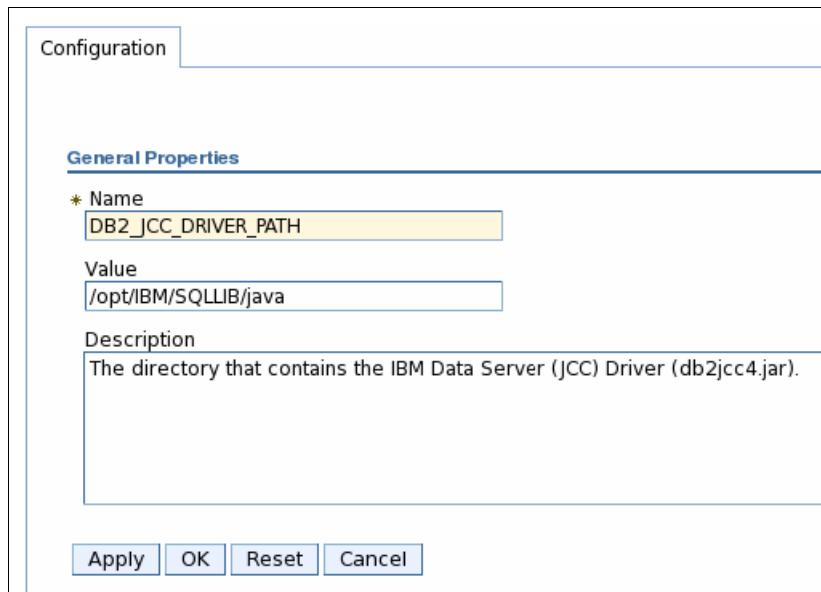


Figure 21-28 Configuring the DB2 driver path

Click **OK**.

4. Repeat the process for the DB2_JCC_DRIVER_NATIVEPATH variable. For DB2, use the same path, /opt/IBM/SQLLIB/java.

Configuring J2C authentication data

The user ID and password that are required to access the database are specified in a J2C authentication data entry.

Complete the following steps:

1. Determine the user ID and password required by DB2 to access the database.
2. In the administrative console, click **Security** → **Global Security**. Expand the Java Authentication and Authorization Service section under the Authentication section and click **J2C authentication data**.

3. Click **New**, and specify the information shown in Figure 21-29 to create the authentication data.

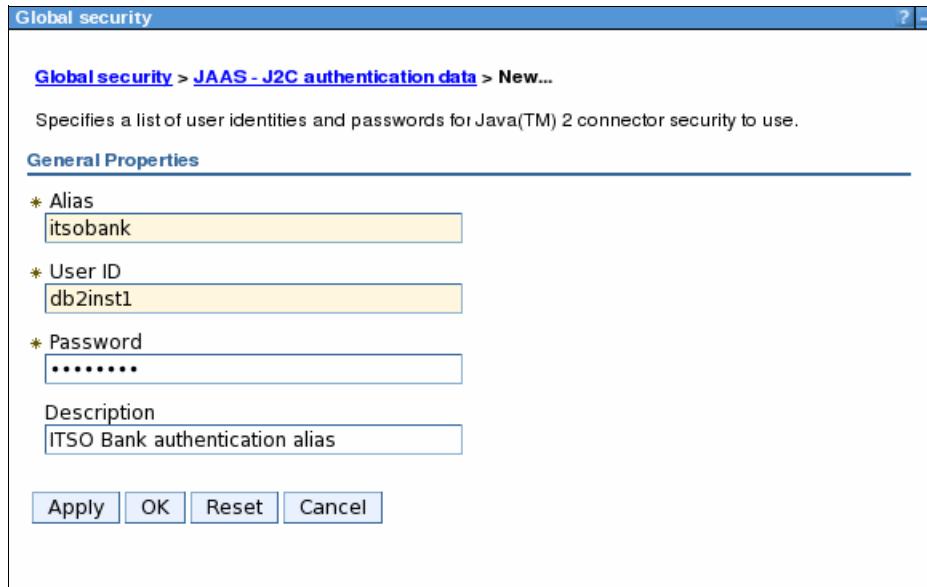


Figure 21-29 Creating the ITSO Bank JAAS authentication alias

4. Click **OK**.

Creating the ITSO Bank JDBC provider

To create a JDBC provider targeting a DB2 database from the administrative console, complete the following steps:

1. Expand the **Resources** entry, and then select the **JDBC** entry. Then, select the **JDBC Providers** entry.
2. Select the scope of this resource. The JDBC provider and data source will be defined at the same scope, so consider the scope that you want to define the data source at and select that same scope here. The more specific the scope, the more isolation you provide for the database.

Also, keep in mind that the JDBC provider will, by default, use variables to define the location of the provider's JAR files. If you plan to use predefined variables for the location of the JDBC provider JAR files, the scope you select for the JDBC provider definition must be at the same scope as the variable, or at a more specific scope. For example, if you defined the \${DB2UNIVERSAL_JDBC_DRIVER_PATH} variable at the node level, then you need to define the JDBC provider at the node or server level. However, you can choose to specify the JAR file locations during the JDBC provider creation process and the variables will be defined at the proper scope.

3. Click **New**.

4. In the Configuration dialog box, select the following general properties for the JDBC provider, as shown in Figure 21-30:

- Database type: DB2
- Provider type: DB2 Using IBM JCC Driver
- Implementation type: XA data source
- Name: DB2 Using IBM JCC Driver (XA)

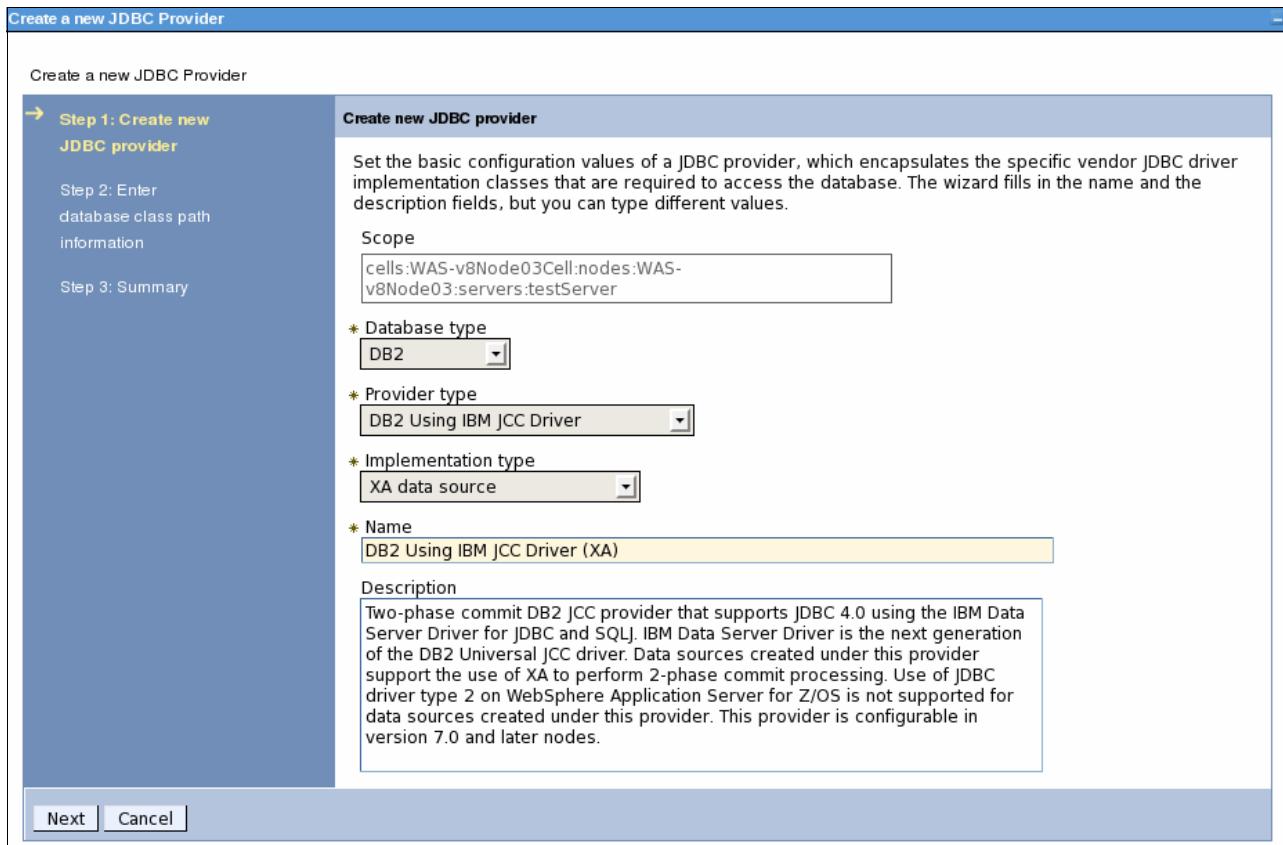


Figure 21-30 Creating a DB2 JDBC provider

Click **Next**.

Naming note: We used the DB2 XA-capable JDBC Driver for the ITSO Bank sample. If your application does not require two-phase commit capabilities, you can use a non-XA capable driver. If you are using an XA-capable driver, it is a best practice to indicate that it is an XA-capable driver by including *XA* in its name, such as *MyJDBCDriverXA*.

5. The next window allows you to change the class path for the provider, and to configure the location of the JAR files. The JAR file locations you specify here will be stored in WebSphere variables at the proper scope for the JDBC provider. You configured the paths earlier and do not need to do it again. Click **Next**.
6. On the Summary window, click **Finish**.

Creating the ITSO Bank data source

The next step is to create the data source for the ITSO Bank DB2 database. To create a data source, complete the following steps:

1. Click **Resources** → **JDBC** → **JDBC Providers**.

2. Click the **DB2 Using IBM JCC Driver (XA)** JDBC provider and then select **Data Sources** under Additional Properties.
3. Click **New** to add the new data source. Enter the following basic data source information, as shown in Figure 21-31:
 - Data source name
Enter the data source name, which must be unique in the administrative domain or cell. Use a value that indicates the name of the database that this data source is targeting, such as ITSOBankDS.
 - JNDI name
Enter the name by which applications access this data source. If not specified, the JNDI name defaults to the data source name, prefixed with jdbc/. For the ITSO Bank, set this field to jdbc/itsobank. You can change this value at any time after you create the data source.

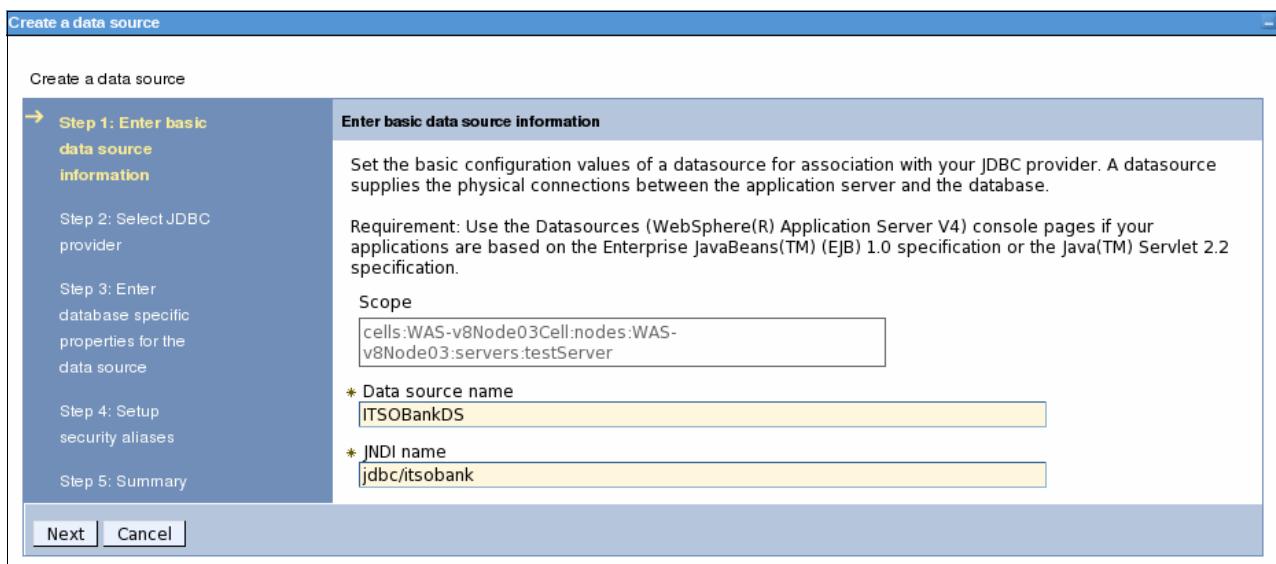


Figure 21-31 ITSO Bank basic data source properties

4. Click **Next**. Enter the following database specific properties, as shown in Figure 21-32 on page 817:
 - Driver type
Select the driver type to use.
If the database is on the same machine as the WebSphere installation, you can use a type 2 driver and do not need to enter a server name. If the database is on a remote machine, you can use either a type 4 driver, which allows WebSphere to connect remotely to the database over TCP/IP, or a type 2 driver.
To use a type 2 driver with a remote database, you need a local DB2 client installation on the WebSphere system and a catalog for the database on the DB2 client. WebSphere then sees the database as local, and the DB2 client handles the remote calls. In our setup, the database is on the same system as the WebSphere installation. So, we choose a type 2 driver and do not enter a server name.
 - Database name
Enter the name of the database (ITSOBANK in our example).

- Port number

Our DB2 installation uses port 50000. So, we keep the default value for this setting.

- Use this Data Source in container-managed persistence (CMP)

Because the ITSO Bank uses Java Persistence API for database persistence instead of CMP EJB, you do not need to set up the data source for CMP EJB. Clear this option.

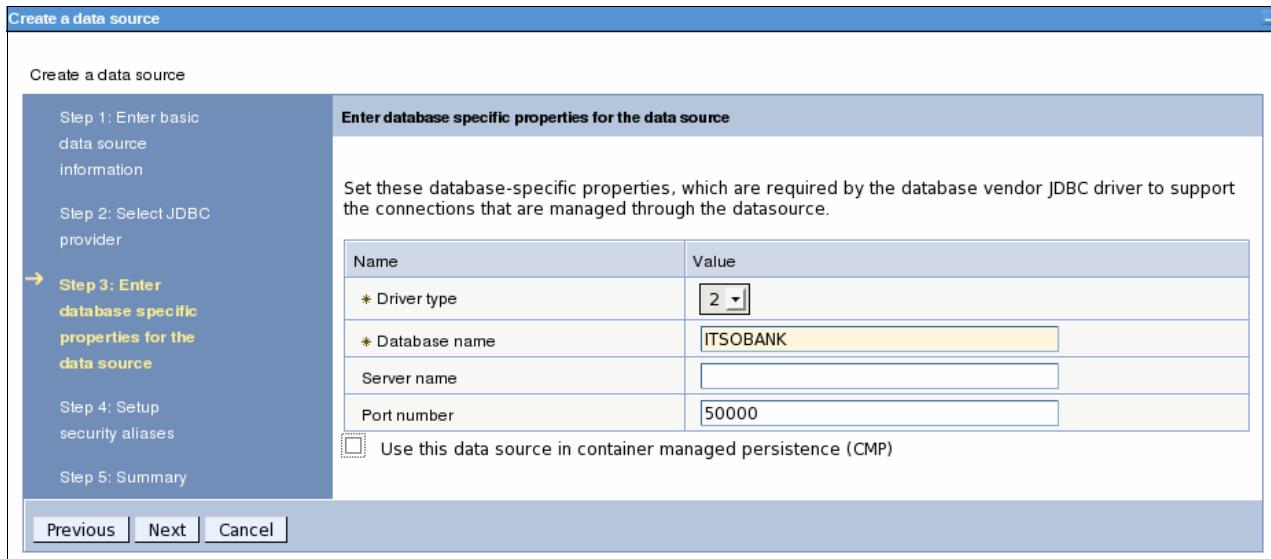


Figure 21-32 ITSO Bank data source database properties

- Click **Next**. Set the security aliases, as shown in Figure 21-32. The container-managed authentication alias is the preferred method for specifying authentication information for the database. Enter the J2C alias that is used for connecting to the data source by selecting the authentication alias that you created previously (*cell name/itsobank*).

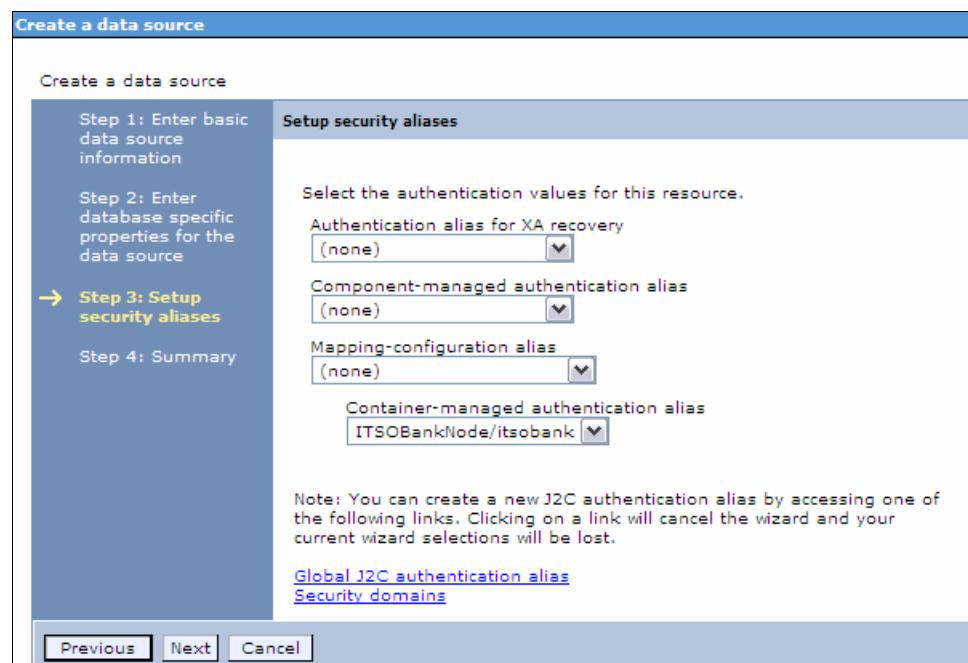


Figure 21-33 ITSO Bank database security alias properties

6. Click **Next**, and then on the summary window, click **Finish**.
7. Save the configuration.
8. Test the connection by selecting the data source and clicking **Test Connection**.

21.8 Deploying the application

In this section, we show how to deploy the application to WebSphere Application Server. The EAR file that we deploy as an example is the RAD8EJBWebEAR file.

To deploy the application, complete the following steps:

1. Click **Applications** → **New Application** from the administrative console navigation bar, and then click the **New Enterprise Application** on the window.
2. Select the **Local file system** option, and then click **Browse** to locate the RAD8EJBWebEAR.ear file. Select the file, and click **Open**.

From the installation window, you can install files that are located either on the same system as the browser that you are using to access the administrative console (the local file system option) or on any node in the cell (the remote file system option).

When you have made your selection, click **Next**. The selected file will be uploaded to the application server or to the deployment manager if this is a distributed server environment.

3. The administrative console allows you to take a shortcut when installing an application. If you select the **Fast Path - Prompt only when additional information is required** option, only the configuration windows where you actually need to complete information during installation are shown.

For this example, however, we take the detailed path to explain the options. Select the **Detailed - Show all installation options and parameters** option.

If you expand **Choose to generate default bindings and mapping**, you can alter the bindings for the application that you are deploying. If you select the **Generate Default Bindings** option, WebSphere Application Server completes any incomplete bindings in the application with default values, but it does not alter any existing bindings. Selecting the **Override existing bindings** option allows you to specify a bindings file that contains new bindings.

The contents of the application or module that you are installing determines which options are displayed on the bindings window. For our ITSO Bank application, the options documented in Table 21-8 are displayed.

Because our application uses Java EE 5 and EJB 3.0 and relies on the bindings and mappings that are generated automatically by the EJB container, you do not need to override bindings. Leave all options for bindings cleared.

Table 21-8 Application default bindings

Binding name	Detailed information
Specific bindings file	You can create a specific bindings file using your favorite editor and load it during application installation by clicking Browse next to the specific bindings file.
Unique prefix for beans	You can generate default EJB JNDI names using a common prefix. EJB beans for which you did not specify a JNDI name get a default name, built by concatenating the prefix and the EJB name. If you specify a prefix of myApp/ejb, JNDI names default to myApp/ejb/EJBName, such as myApp/ejb/Account.

Binding name	Detailed information
Virtual host bindings	You can bind all web modules to a specific virtual host, such as itsobank_host.

Click **Next**.

The remainder of the wizard is divided into steps. The number of steps depends on your application. For example, if the application contains EJB modules or web modules, you are prompted for the information necessary to deploy them.

4. Step 1: Select installation options.

Step 1 gives you a chance to review the installation options. You can specify various deployment options, such as JSP precompiling, and whether you want to generate EJB deployment code (not applicable for EJB 3 or later beans). In this step, you can set the following options:

- If you are deploying an Enhanced EAR file, you specify here whether to use the resource configuration information that is packaged in the Enhanced EAR file. The installation window pre-selects the **Process embedded configuration** option. If you do not want to use the resource configuration information that is packaged in the Enhanced EAR file, clear this option. In this example, we have configured the necessary resources using the administrative console, so make sure that this option is not selected.
- Selecting the **Pre-compile JavaServer Pages files** option makes WebSphere compile all JSP pages in the EAR file during installation instead of during run time. Thus, the first user who accesses the application does not have to wait for the JSP pages to compile.

An alternative to precompiling JSP pages is to use the **JspBatchCompiler** script found in the bin directory of the profile that you are using to compile the JSP pages after the application is installed.

- You can specify file permissions for files in your application. To use one of the predefined file permissions, select it, and then click **Set file permissions**. You can also specify your own file permissions using regular expressions.
- The administrative console displays the Application Build ID of the application that is being installed. This string is specified in the MANIFEST.MF file in the EAR file's META-INF folder and can be set using the Rational Application Developer Assembly and Deploy tool.

The following example shows a version number that is specified in the MANIFEST.MF file:

Implementation-Version: Version 1.2.3

- The dispatching and servicing of remote resources are extensions to the web container that allows frameworks, servlets, and JSP pages to include content from outside of the current executing resource's JVM as part of the response that is sent to the client.

To enable these features, select the options to allow dispatching or servicing to or from remote resources.

- The “Allow EJB reference targets to resolve automatically” option is used for EJB 2.1 or earlier or Web 2.3 or earlier modules and allows WebSphere Application Server to provide a default value or to resolve EJB references automatically for any EJB reference that does not have a binding. Because the sample application is at EJB 3.0, this option does not apply.

After you have changed the settings as appropriate, click **Next**.

5. Step 2: Map modules to servers.

Select the server on which you want each module deployed. For better performance, deploy all modules from one application in a single server. Specifically, do not separate the EJB clients, usually servlets in web modules, from the EJB beans themselves.

Click the icon to select all modules in the ITSO Bank EAR file. In the “Clusters and Servers” field, select **ITSOBankServer1**. Then, click **Apply** to assign all modules to the ITSOBankServer1 application server. If you deploy to a cluster, select the cluster instead of the single application server. See Figure 21-34.

Web servers: If you have a web server defined, select both the web server and ITSOBankServer1 in the server list. Press and hold the Ctrl key to select multiple servers. Mapping web modules to web servers ensures that the web server plug-in is generated properly.

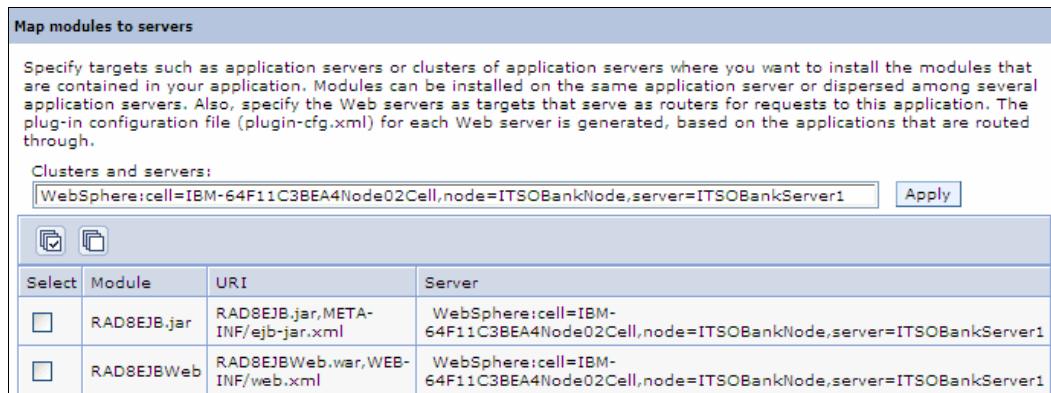


Figure 21-34 Mapping modules to application servers

After you map the module, click **Next**.

6. Step 3: Provide JSP reloading options for web modules.

This setting allows you to configure if and how often WebSphere should check for updates to JSP files and whether they are reloaded. In a production environment, you might want to disable this setting to improve performance. Click **Next**.

7. Steps 4 and 5: Map shared libraries, and Map shared library relationships.

If your application depends on shared libraries, you can specify them here. Click **Next**.

8. Step 6: Initialize parameters for servlets.

For servlets that honor initialization parameters (specified by the `init-param` tag in the web module's `web.xml` deployment descriptor), you can configure the value of the parameters. Click **Next**.

9. Step 7: Provide JNDI names for beans.

Use this window to bind the enterprise beans in your application or module to a JNDI name. Because our application is at EJB 3.1, we can leave it blank to have WebSphere Application Server use the default names. Click **Next**.

10. Step 8: Bind EJB Business interfaces to JNDI names.

Specify a JNDI name for the business interfaces of your EJB beans. Because our application is at EJB 3.1, we can leave this setting blank to have WebSphere Application Server assign default JNDI name. Click **Next**.

11.Step 9: Map EJB References to beans.

Each EJB reference that is defined in your application must be mapped to an enterprise bean. We can leave it blank to have WebSphere Application Server use the default names. If the reference was in an EJB 2 or earlier or Web 2.3 or earlier module, you can select the **Allow EJB reference targets to resolve automatically** option and then not need to specify a target JNDI name either. Click **Next**.

12.Step 10: Map virtual hosts for web modules.

Select the virtual host that you created for the application (itsobank_host).

Virtual host definitions: Only those virtual host definitions that are defined to the WebSphere environment are configurable at this step. To map the web modules to the virtual host that is defined in the Enhanced EAR file, you need to configure that host after deploying the application.

Click **Next**.

13.Step 11: Map context roots for web modules.

Select the context root against which to bind the module. Click **Next**.

14.Step 12: Metadata for modules.

Selecting the metadata-complete attribute tells WebSphere Application Server to ignore any deployment information that is specified in source code annotations. Leave both options cleared to use the information from the annotations. Click **Next**.

15.Step 13: Summary.

The Summary window gives an overview of the application deployment settings. If those settings are fine, click **Finish** to deploy the application.

16.Save the configuration.

If you are working in a distributed server environment, make sure that you synchronize the changes with the nodes so that the application is propagated to the target application server or servers.

If you mapped the web modules to a web server, make sure that the web server plug-in is regenerated and propagated to the web server. For a quick refresh, restart the web server.

17.Start the application:

- a. Click **Applications** → **Application Types** → **WebSphere Enterprise Applications**.
- b. Select the **RAD8EJBWebEAR** application, then click **Start**.

Deployment of the RAD8EJBWebEAR application is now complete. You can verify that the application works by pointing your browser to:

<http://www.itsobank.ibm.com:9080/RAD8EJBWeb>

If successful, the web page for the ITSO Bank application displays. Click the **RedBank** button and provide customer number 111-11-1111 to see an example of a customer's accounts, as shown in Figure 21-35.

The screenshot shows a web application titled "ITSO RedBank". At the top, there is a logo consisting of two red circles. Below the title, there is a form with fields for SSN (111-11-1111), Title (Mr), First name (John), and Last name (Doe). There are three buttons at the bottom of this form: "Update", "New Customer", and "Delete Customer".

Account Number	Balance
001-111001	12,345.67
001-111002	6,543.21
001-111003	98.76
001-574252	0.00

At the bottom of the page, there are two buttons: "Add Account" and "Logout".

Figure 21-35 ITSO Bank web application

If you have any problems related to virtual hosts, restart the server and try again. WebSphere Application Server might need a restart to pick up virtual host changes.

21.9 Deploying business-level applications

A *business-level application* (BLA) is a concept that aims to expand the notion of an “application” beyond Java EE. Its administration model provides the entire definition of an application as it makes sense to the business. In contrast with an enterprise application (EAR file), a business-level application is only a logical WebSphere configuration artifact, similar to a server or cluster, that is stored in the configuration repository.

In this section, we introduce business-level applications and show how you can deploy a Java EE application by creating a business-level application and adding the Java EE application to it as an asset.

Support for business-level applications: Business-level applications are supported only on WebSphere Application Server V7 or later nodes. They are not supported on any previous versions of WebSphere Application Server.

Figure 21-36 shows the concept of business-level applications.

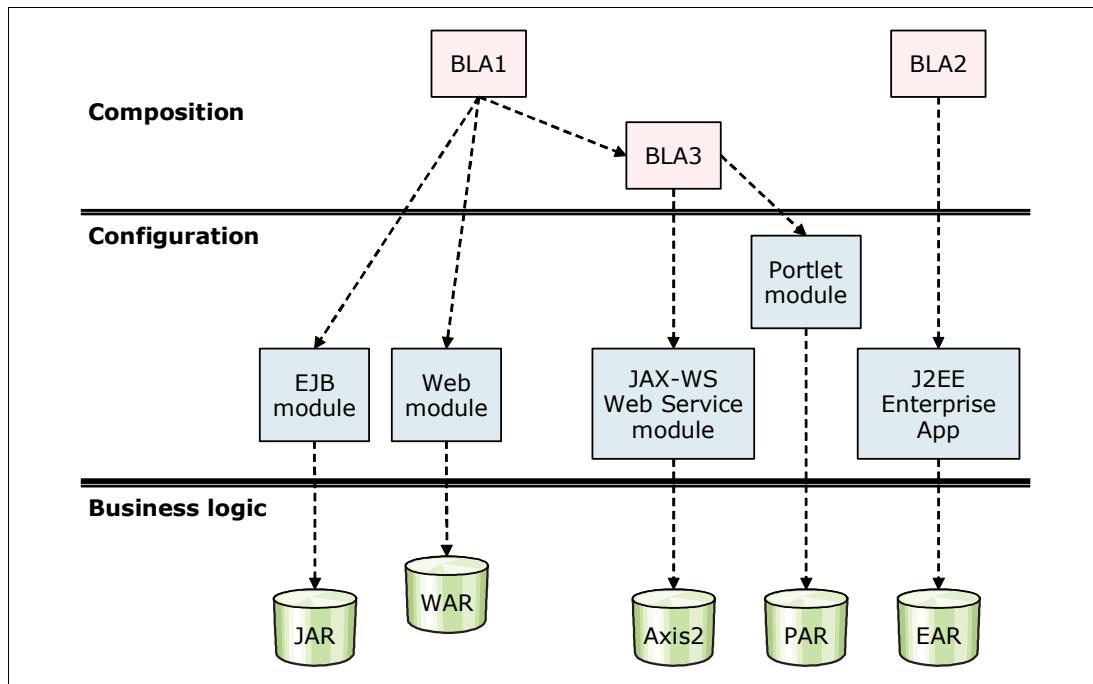


Figure 21-36 Business-level application concept

Business-level applications can be used in several different ways. Often a business application, such as an Order System, does not consist of only one enterprise application (EAR), but rather multiple applications that must all be running for the whole business application to work.

One way of using business-level applications is to group the separate enterprise applications that make up the business application into one manageable unit that can be started, stopped, updated, and so on. However, a business-level application can reference Java EE components as well as assets that are not part of the Java EE concept, for example, CORBA (C++) executables that are hosted in a generic server or files on the file system that are not managed by WebSphere but that are required by the application.

A business-level application does not represent or contain application binary files. Instead, it is a configuration that lists one or more composition units that represent the application binary files. A business-level application uses the binary files to run the application business logic. Administration of binary files is done using the normal methods for managing modules (for example, web or EJB modules) and is separate from administration of the application definition.

A business-level application does not introduce any new programming, runtime, or packaging models:

- You do not need to change your application business logic. The business-level application function does not introduce new application programming interfaces (APIs).
- You do not need to change your application runtime settings. WebSphere supports all of the runtime characteristics, such as security, class loading, and isolation, required by individual programming models to which business components are written.
- You do not need to change your application packaging. There is no specific unique packaging model that provides a business-level application definition.

The terminology for business-level applications introduces two new terms:

- ▶ An *asset* represents one or more application binary files that are stored in an asset repository. Typical assets include application business logic, such as EAR files, EJB modules, web modules, service component architecture (SCA) modules, shared library files, static content, and other resource files. The asset repository is managed by WebSphere Application Server and does not require any third-party software.

You must register files as assets before you can add them to one or more business-level applications. At the time of asset registration, you can import the physical application files into WebSphere's configuration repository or you can specify an external location where the asset resides.

- ▶ A *composition unit* represents a configured asset in a business-level application. Configured in this context means installed, so a configured web module means a web module which is installed.

WebSphere Application Server handles the following types of composition units:

- ▶ Asset composition units
Composition units created from assets by configuring each deployable unit of the asset to run on deployment targets.
- ▶ Shared library composition units
Composition units created from JAR-based assets by ignoring all the deployable objects from the asset and treating the asset JAR file as a library of classes.
- ▶ Business-level application composition units
Composition units created from business-level applications that are added to existing business-level applications.

Figure 21-37 shows the relationship between assets, composition units, and business-level applications.

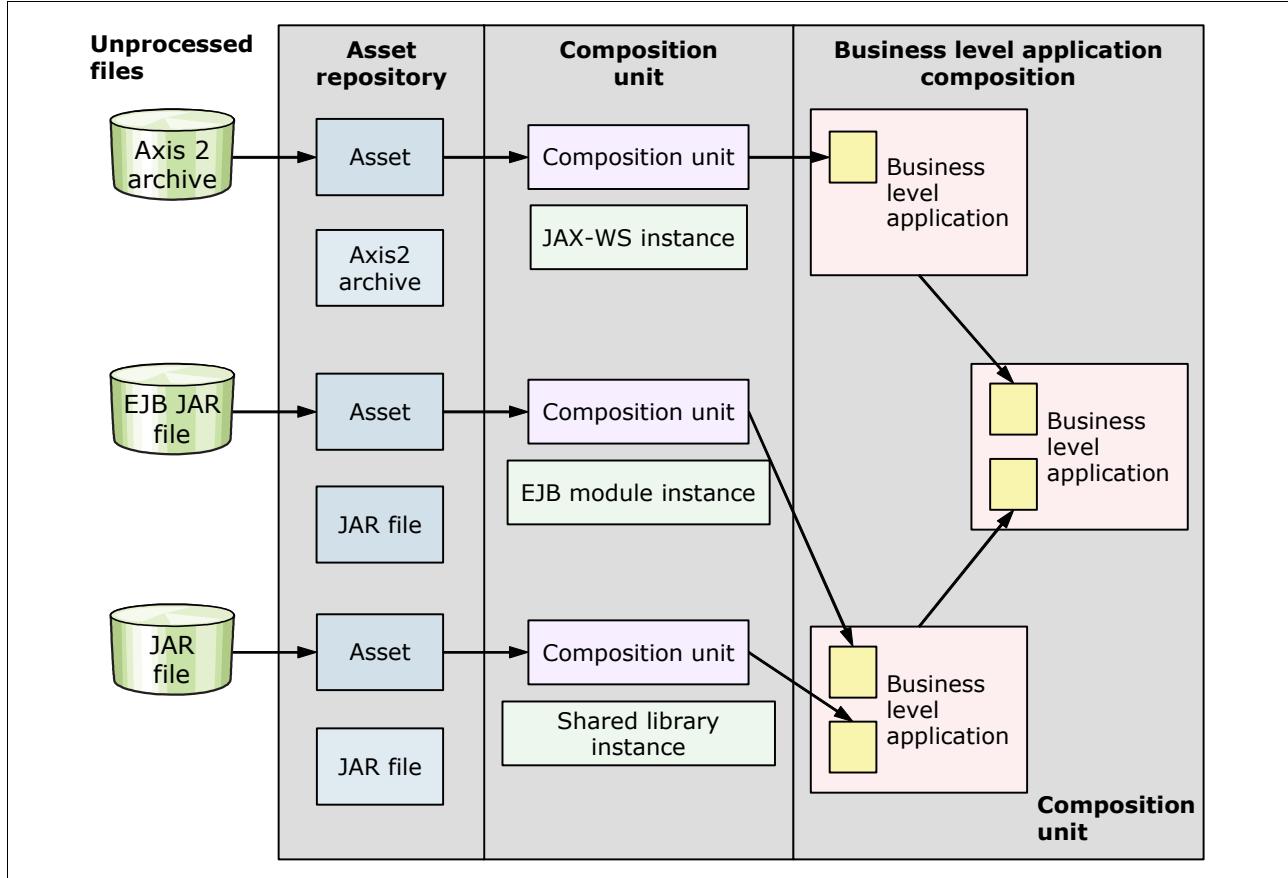


Figure 21-37 Relationship between business-level application artifacts

If an asset depends on another asset, you can create a relationship between them to allow, for example, a web module asset to reference classes in a shared library.

So, to summarize, a business-level application consists of composition units. When you add an asset to a business-level application, a composition unit is created for the asset. The composition unit is the configured (installed) asset.

21.9.1 Creating a business-level application

As a basic example of how to create a business-level application, we use the RAD8EJBWeb application (see 21.2, “Preparing to use the sample application” on page 788) and combine it with the WebSphere DefaultApplication’s web module (which includes, for example, the SnoopServlet) that can be extracted from the DefaultApplication’s EAR file. We create one asset for the RAD8EJBWeb application and another for the DefaultWebApplication. We then create a business-level application containing these two assets.

When you create a business-level application and assets to the application, those assets are deployed as part of the process. The configuration of the runtime environment and the resources that the deployed assets will need should be done first. This example assumes that the configuration described in 21.7, “Preparing the runtime environment for the application” on page 805 has been completed.

To create the two assets:

1. Open the WebSphere administrative console and click **Applications** → **Application Types**. Click the **Assets** link.
2. Click **Import**. Select the **Local file system** option, and then click **Browse** to locate the RAD8EJBWebEAR.ear file. Select the file and click **Open**. Then, click **Next**.
3. For Step 1: Select options for importing an asset., enter a brief description of the asset, if wanted. If you want to import the asset to a specific path on your file system, enter the path in the “Asset binaries destination URL” field. If you leave this field blank, the file is imported to its default location, which is *profile_root*/installedAssets/asset_name/BASE/. Click **Next**.

Tip: If specifying another name for the asset, make sure to keep the asset's file extension (such as .ear or .war). Otherwise, WebSphere cannot keep track of the asset type and fails to import the asset.

4. On the Summary window, click **Finish**. The asset is now imported into WebSphere's asset repository, but it is not yet configured (thus, it is still just an asset and not a composition unit).
5. Repeat steps 1 to 4 and import the DefaultWebApplication.war file. (As mentioned, this file was extracted from the DefaultApplication EAR file and stored on our local operating system).

Because the two assets do not depend on each other and do not require any shared library, you do not need to set up any relationships.

6. Select **Save to master configuration** when finished.

After the assets are imported into the asset repository, you can create a business-level application.

7. Click **Applications** → **Application Types** → **Business-level applications**. Click **New**.
8. Enter a name, such as ITSOBank System, and click **Apply**.

- On the Business-level applications page, click **Add** under the Deployed assets section and select the **Add Asset** option, as shown in Figure 21-38.

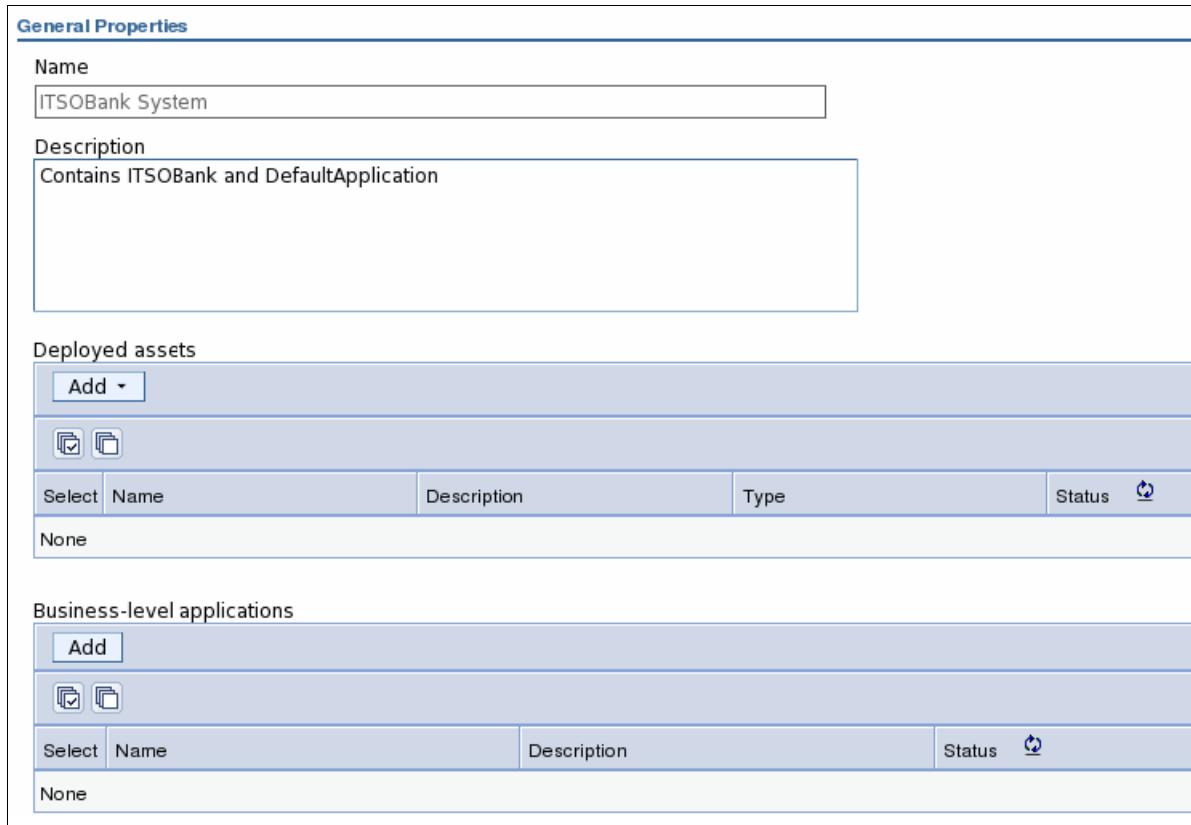


Figure 21-38 Business-level application configuration window

- On the next window, select the **RAD8EJBWebEAR.ear** asset, and click **Continue**.
- Next, configure (install) this asset with the necessary deployment options. The installation process follows the same steps for installing an application to WebSphere Application Server.

For detailed step-by-step instructions, refer to 21.8, “Deploying the application” on page 818. Proceed through the installation windows until the Summary window opens, and click **Finish**. WebSphere now installs the application, and it is a composition unit, that is, an asset that has been configured.

Installation note: In Step 1: Select installation options, WebSphere generates a unique application name, such as app1143018803114601914, for the application. You might want to change this name to a more descriptive name.

- Click **Applications → Application Types → Business-level applications**, and click the link for the ITSOBank System application.
- Repeat steps 9 and 10 and add the DefaultWebApplication.war file to the ITSOBank System business-level application as well.
- Save to the master configuration when done.

You can now stop and start the business-level application by clicking **Applications → Application Types → Business-level applications** and then clicking the corresponding links.

Deleting a business-level application: To delete a business-level application, you must first unmap (delete) the composition units (configured assets) that belong to the business-level application. Select the business-level application, and select the deployed assets. Then, click **Delete**. After you delete all assets from the business-level application, you can delete the business-level application itself. The assets, however, still remain in WebSphere's asset repository and can be used to configure other business-level applications.

You can manage the individual applications, the composition units, that make up the business-level application individually by clicking **Applications** → **Application Types** → **WebSphere enterprise applications** and using the links to start, stop, update, and so on.

21.10 Deploying application clients

To run a Java-based client/server application, the client application executes in a client container of some kind. You might, for example, use a graphical Swing application that calls EJB beans on an application server. WebSphere Application Server V8 supports several different application client environments. The following application client environments are available:

► **Java EE client**

This client uses services that are provided by the Java EE client container.

This client is a Java application program that accesses EJB beans, JDBC databases, and JMS queues. The Java EE application client depends on the application client run time to configure its execution environment, and it uses the JNDI name space to access resources, the same as you would in a server application (similar to a servlet).

The Java EE application client provides the following components:

- XML deployment descriptors
- Java EE naming (`java:comp/env`), including EJB references and resource references

The Java EE application client is launched using the `launchClient` script, which sets up the environment with the necessary class paths and other settings for you.

► **Java thin client**

This client does not use services that are provided by the Java EE client container.

The Java thin client runtime environment provides the support needed by full-function Java SE client applications but does not support a Client Container that provides easy access to these services. The Java thin client is designed to support those users who want a full-function Java SE client application programming environment, to use the supplied IBM JRE, without the impact of the Java Platform, Enterprise Edition (Java EE) platform on the client machine. The Java thin client does not perform initialization of any of the services that the client application might require. For example, the client application is responsible for the initialization of the naming service, either through CosNaming or JNDI APIs. The Java thin client runtime environment does provide support for Java SE client applications to access remote enterprise beans, and provides the implementation for various enterprise bean services. Client applications can also use the Java thin client runtime environment to access CORBA objects and CORBA based services.

The thin client supports JVMs from IBM, Sun and HP-UX. When launching the thin application client, you must set up the correct class paths yourself and make sure that the required libraries for your application and the WebSphere libraries are included.

- ▶ Applet application client

In the Applet client model, a Java applet that is embedded in an HTML document executes in a web browser. With this type of client, the user accesses an enterprise bean in the application server through the Java applet in the HTML document.

- ▶ ActiveX to EJB Bridge application client

The ActiveX application client allows ActiveX programs to access enterprise beans through a set of ActiveX automation objects. The ActiveX application client uses the Java Native Interface (JNI) architecture to programmatically access the Java virtual machine (JVM) API. Therefore, the JVM code exists in the same process space as the ActiveX application (Visual Basic, VBScript, or Active Server Pages files) and remains attached to the process until that process terminates. The ActiveX to EJB Bridge is supported on Windows systems only.

For detailed capabilities of each client container, search the Information Center for *Client Applications*, or go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.nd.multiplatform.doc/info/ae/ae/ccli_clientapps.html

21.10.1 Installing application clients

You install the application client environments from the WebSphere installation wizard by selecting the **Launch the installation wizard for WebSphere Application Clients** option. The installation package contains the following installable components:

- ▶ IBM Java Runtime Environment (JRE) or an optional full Software Development Kit
- ▶ Java EE application client and Java thin application client
- ▶ ActiveX to EJB Bridge run time for ActiveX to EJB Bridge application client applications (only for Windows systems)
- ▶ Pluggable Client (deprecated)
- ▶ Samples for the various application client containers

Java EE client installation: The Java EE client is installed automatically as part of a full WebSphere installation. In other words, if you run the client application on a system that already has WebSphere installed, you do not need to install the WebSphere Java EE client.

21.10.2 Preparing the sample application

The ITSO Bank application also has a stand-alone application client that we use to demonstrate the WebSphere Application Server application client container.

We need to import the project that contains the stand-alone application client into IBM Assembly and Deploy Tools for WebSphere Administration and export it as an EAR file. This process is similar to the one covered in detail in 21.2, “Preparing to use the sample application” on page 788.

Complete the following steps:

1. Import the j2eeclient\RAD8AppClient.zip Project Interchange file from the 7835codesolution folder. Select both projects when importing.

2. Because the RAD8AppClient project has a specific binding that is configured for the EJBBankBean in the RAD8EJBEAR file, you need to modify this binding to point to the same bean in the RAD8EJBWebEAR file, which is the application that you installed on your server. To modify the binding:
 - a. Expand the **RAD8AppClient** project, and double-click the **RAD8AppClient** deployment descriptor.
 - b. Then, click the **Open WebSphere Bindings Descriptor** link from the right pane.
 - c. Select **EJB Reference (ejb/bank)**, and change the name from `ejb/RAD8EJBEAR/RAD8EJB.jar/EJBBankBean#itso.bank.service.EJBBankRemote` to `itso.bank.service.EJBBankRemote`. By using the short name, we rely on the Autolink feature to search for and invoke a matching EJB interface. This method works in our environment where we only have one instance of the EJB installed and running.
 - d. Press **Ctrl+s** to save the deployment descriptor.
3. Export the RAD8AppClientEAR as described in 21.5, “Creating WebSphere Enhanced EAR files” on page 793 to `/apps/RAD8AppClientEAR_withModifiedBinding.ear`.

21.10.3 Launching the J2EE client

A Java EE client application needs a container in which to run. In this example, we use the Java EE application client container. You can start this container using the **LaunchClient** program in the *install_root/bin* directory. The launchClient program has the following syntax:

```
Usage: launchClient [-profileName pName | -JVMOptions options | -help | -?]
<userapp.ear> [-CC<name>=<value>] [app args]
```

The elements of syntax are:

-profileName	Defines the profile of the application server process in a multi-profile installation. The -profileName option is not required for running in a single profile environment or in an application client installation. The default is default_profile .
-JVMOptions	Specifies a valid Java standard or nonstandard option string. Insert quotation marks around the option string.
-help, -?	Prints the usage information.
<userapp.ear>	Enter the path name and file name of the .ear file that contains the client application.

The **-CC** properties are for use by the application client run time. Numerous parameters are available. We describe only the commonly used parameters here. For full explanation of all parameters, run **LaunchClient -help**. The **-CC** properties include the following parameters:

-CCverbose	Specify this option with <code><true false></code> to display additional informational messages. The default value is false .
-CCclasspath	Defines a class path value. When an application is launched, the system class path is not used. If you need to access classes that are not in the EAR file or that are not part of the resource class paths, specify the appropriate class path here. Multiple paths can be concatenated.

-CCjar	Specifies the name of the client JAR file within the EAR file that contains the application that you want to launch. This argument is necessary only when you have multiple client JAR files in the EAR file.
-CCBootstrapHost	Defines the name of the host server to which you want to connect initially.
-CCBootstrapPort	Specifies the server port number. If not specified, the WebSphere default value (2809) is used.
-CCproviderURL	Provides bootstrap server information that the initial context factory can use to obtain an initial context. A WebSphere Application Server initial context factory can use either a CORBA object URL or an IIOP URL. CORBA object URLs are more flexible than IIOP URLs and are the recommended URL format to use. This value can contain more than one bootstrap server address. This feature can be used when attempting to obtain an initial context from a server cluster. In the URL, you can specify bootstrap server addresses for all servers in the cluster. The operation will succeed if at least one of the servers is running, eliminating a single point of failure. The address list does not process in a particular order. For naming operations, this value overrides the -CCBootstrapHost and -CCBootstrapPort parameters. The following example shows a CORBA object URL that specifies multiple systems: <code>-CCproviderURL=corbaloc:iiop:myserver.mycompany.com:9810,:mybackupserver.mycompany.com:2809</code>
-CCtrace	Specify this option with <true false> to have WebSphere write debug trace information to a file. The value true is equivalent to a trace string value of com.*=all=enabled . Instead of the value true , you can specify a trace string, for example -CCtrace=com.ibm.ws.client.*=all=enabled . You can specify multiple trace strings by separating them with a colon (:). You might need this information when reporting a problem to IBM Service. The default value is false .
-CCtracefile	Defines the name of the file to which to write trace information. The default is to output to the console.
-CCprofile	Defines the name of a properties file that contains the launchClient properties. In the file, specify the properties without the -CC prefix, for example, verbose=true .

The application arguments are for use by the client application and are ignored by WebSphere.

To start the ITSO Bank stand-alone application client using the **launchClient** command, execute the command shown in Example 21-3.

Example 21-3 Launching the ITSO Bank stand-alone application client

```
/opt/IBM/WebSphere/AppServer/profiles/ITSOBank/bin>./launchClient.sh
/apps/RAD8AppClientEAR_withModifiedBinding.ear
```

IBM WebSphere Application Server, Release 8.0
Java EE Application Client Tool
Copyright IBM Corp., 1997-2011
WSCL0012I: Processing command line arguments.
WSCL0013I: Initializing the Java EE Application Client Environment.

```
[2011-07-08 16:32:05:421 CEST] 00000000 W UOW=null  
source=com.ibm.ws.ssl.config.SSLConfig org=IBM prod=WebSphere  
component=Application Server thread=[P=321234:  
0=0:CT]  
CWPKI0041W: One or more key stores are using the default password.  
WSCL0035I: Initialization of the Java EE Application Client Environment has  
completed.  
WSCL0014I: Invoking the Application Client class  
itso.rad8.client.control.BankDesktopController
```

The application opens and displays a graphical window. Enter 111-11-1111 as the social security number. The results should look like Figure 21-39.

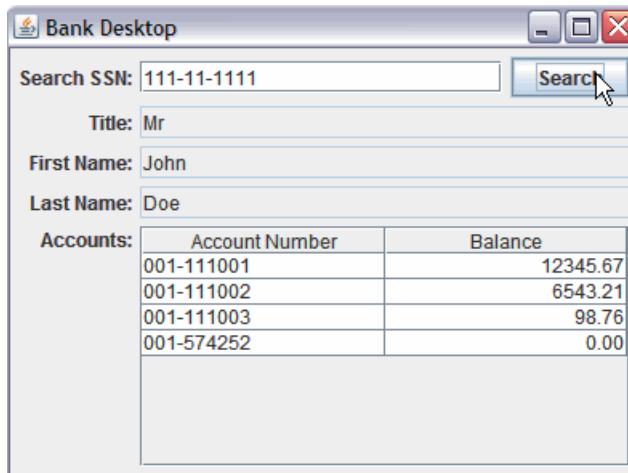


Figure 21-39 Running the ITSO Bank stand-alone application client



Updating Java EE applications

WebSphere Application Server V8 has features that allow applications to be updated and restarted at a fine-grained level. It is possible to update only parts of an application or module and to restart only the necessary parts. With these features, you can perform the following tasks:

- ▶ Replace an entire application (.ear file).
- ▶ Replace, add, or remove a single module (.war, EJB .jar, or connector .rar file).
- ▶ Replace, add, or remove a single file.
- ▶ Replace, add, and remove multiple files by uploading a compressed file that describes the actions to take.
- ▶ Use a monitored directory to deploy or update enterprise applications.

In this chapter, we explain these features in the following topics:

- ▶ Working with applications
- ▶ Replacing an entire application EAR file
- ▶ Replacing or adding an application module
- ▶ Using a monitored directory

22.1 Working with applications

If an application is running when it is updated, WebSphere Application Server stops the application or the affected components automatically, updates the application, and then restarts the application or components.

When updating an application, only the portion of the application code that is changed needs to be presented to the system. The application management logic calculates the minimum actions that the system needs to execute to update the application. Under certain circumstances, the update can occur without stopping any portion of the running application.

WebSphere Application Server also has support for managing applications in a cluster for continuous availability. The Rollout Update action updates sequentially an application that is installed on multiple cluster members across a cluster. After you update an application's files or configuration, use the Rollout Update option to install the application's updated files or configuration on all cluster members of a cluster on which the application is installed.

The Rollout Update option completes the following actions for each cluster member in sequence:

1. Saves the updated application configuration.
2. Stops all cluster members on a given node.
3. Updates the application on the node by synchronizing the configuration.
4. Restarts the stopped cluster members on that node.

This action updates an application on multiple cluster members while providing continuous availability of the application.

WebSphere Application Server V8 supports an additional OSGi application type that has a modular and dynamic design. WebSphere Application Server V8 features additional functionality that is designed to be used to manage the OSGi application life cycle. We explain this feature in more detail in 24.1.2, “OSGi application life cycle” on page 874.

22.2 Replacing an entire application EAR file

To replace a full EAR file with a newer version, complete the following steps:

1. Click **Applications** → **Application Types** → **WebSphere enterprise applications**.
Select the application to update, and then click **Update**.

2. On the “Preparing for the application update” window (Figure 22-1), select the **Replace the entire application** option (this is the default option).

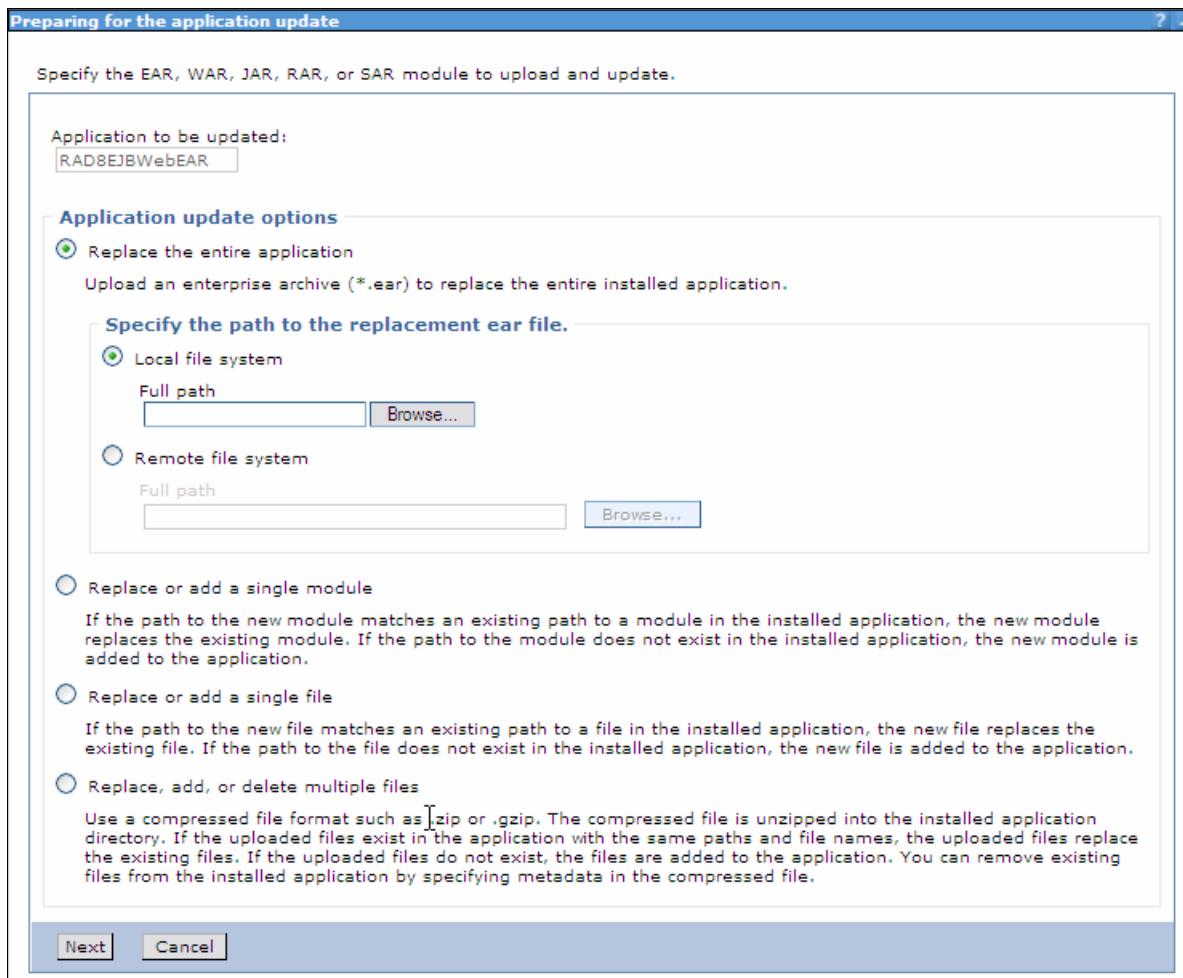


Figure 22-1 Application update options

3. Select either the **Local file system** or **Remote file system** option. Click **Browse** to select the updated EAR file. Then, click **Next**.
4. Proceed through the application installation windows, and make any necessary changes. The options for an application installation are described in 21.8, “Deploying the application” on page 818. On the Summary window, click **Finish**.
5. To preview the files that are changed after the update installation, click the **Review** link (Figure 22-2).

Changes have been made to your local configuration. You can:

- [Save](#) directly to the master configuration.
- [Review](#) changes before saving or discarding.

Figure 22-2 Saving the application to the master repository

Figure 22-3 shows a changed files list.

The screenshot shows a software interface titled 'Enterprise Applications'. At the top, there's a navigation bar with 'Enterprise Applications > Enterprise Applications > Save'. Below it, instructions say 'Save your workspace changes to the master configuration.' and 'Click Save to update the master repository with your changes. Click Discard to discard your changes and begin work again using the master repository configuration. Click Cancel to continue working with your changes.' A section titled 'Total changed documents: 21' contains a table with 11 rows of file changes. The columns are 'Changed Items' and 'Status'. The items listed are various XML and JAR files under 'cells / aix-target1Cell01 / nodes / aix-target1Node01 / serverindex.xml' and 'cells / aix-target1Cell01 / applications / DefaultApplication.ear / deltas / DefaultApplication / delta-1213634107218'.

Changed Items	Status
cells / aix-target1Cell01 / nodes / aix-target1Node01 / serverindex.xml	Updated
cells / aix-target1Cell01 / applications / DefaultApplication.ear / deltas / DefaultApplication / delta-1213634107218	Deleted
cells / aix-target1Cell01 / applications / DefaultApplication.ear / DefaultApplication.ear	Updated
cells / aix-target1Cell01 / applications / DefaultApplication.ear / deltas / DefaultApplication / delta-1310495461173	Added
cells / aix-target1Cell01 / applications / DefaultApplication.ear / deployments / DefaultApplication / Increment.jar / META-INF / ibm-ejb-jar-bnd.xmi	Updated
cells / aix-target1Cell01 / applications / DefaultApplication.ear / deployments / DefaultApplication / META-INF / MANIFEST.MF	Updated
cells / aix-target1Cell01 / applications / DefaultApplication.ear / deployments / DefaultApplication / META-INF / application.xml	Updated
cells / aix-target1Cell01 / applications / DefaultApplication.ear / deployments / DefaultApplication / Increment.jar / META-INF / ibm-ejb-jar-ext.xmi	Updated

Figure 22-3 Change log of the updated application files

After you have reviewed the files, save the configuration by clicking **Save** at the bottom of the page.

6. If you are working in a distributed server environment, make sure that you also synchronize the changes with the nodes.
7. If the application update changes the set of URLs that are handled by the application (that is, if servlet mappings are added, removed, or modified), make sure that the web server plug-in is regenerated and propagated to the web server.

Timing note: It might take a few seconds for the WebSphere run time to pick up the changes and to restart the application as necessary. If your changes do not seem to have effect, wait and try again. You can also look at the SystemOut.log file for the application server to see when it has restarted the application.

22.3 Replacing or adding an application module

To replace only a module, such as an EJB or web module of an application, complete the following steps:

1. Click **Applications** → **Application Types** → **WebSphere enterprise applications**. Select the application to update, and then click **Update**.
2. On the “Preparing for the application update” window (Figure 22-1 on page 835), select the **Replace or add a single module** option.
3. In the “Specify the path beginning with the installed application archive file to the module to be replaced or added” field, enter the relative path to the module to replace. For example, if you were to replace the HelloWeb module, enter HelloWeb. If you enter a path or file that does not exist in the EAR file, the module will be added.
4. Select either the **Local file system** or **Remote file system** option, and then click **Browse** to select the updated module.

5. Click **Next**.
6. Proceed through the remaining windows, and make any necessary changes. On the Summary window, click **Finish**.

Note: If you are adding a web module, make sure that you select the detailed installation option. This option allows you to select the correct target server for the module in the “Map modules to servers” step and to specify a context root for the module.

7. To update the configuration in the master repository, select the **Save** link.
8. If you are working in a distributed server environment, make sure that you also synchronize the changes with the nodes.
9. If the application update changes the set of URLs that are handled by the application (that is, servlet mappings are added, removed, or modified), make sure that the web server plug-in is regenerated and propagated to the web server.

Tip: Modules can also be managed using the Manage Modules page. Click **Applications** → **Application Types** → **WebSphere enterprise applications**, and then click the link for the application. Click the **Manage Modules** link in the Modules section. Select the module to modify, and then click the **Remove**, **Update**, or **Remove File** buttons.

22.3.1 Replacing or adding single files in an application or module

To replace a single file, such as a GIF image or a properties file in an application or module, complete the following steps:

1. Click **Applications** → **Application Types** → **WebSphere enterprise applications**. Select the application to update, and then click **Update**.
2. On the “Preparing for the application installation” window (Figure 22-1 on page 835), select the **Replace or add a single file** option.
3. In the “Relative path to file” field, enter the relative path to the file to replace in the EAR file. For example, to replace the logo.gif file in the images directory of the HelloWeb.war web module, enter HelloWeb.war/images/logo.gif. If you enter a path or file that does not exist in the EAR file, it will be added.
4. Select either the **Local file system** or **Remote file system** option, and then click **Browse** to locate the updated file. Click **Next**.
5. On the Updating Application window, click **OK**.
6. To update the configuration in the master repository, select the **Save** link.
7. If you are working in a distributed server environment, make sure that you also synchronize the changes with the nodes.

22.3.2 Removing application content

You can remove files either from an EAR file or from a module in an EAR file, as we describe in the sections that follow.

Removing files from an EAR file

To remove a file from an EAR file, complete the following steps:

1. Click **Applications** → **Application Types** → **WebSphere enterprise applications**. Select the application to remove the file from, and then click **Remove File**.
2. In the “Select the file to be removed” dialog box, select the file to be removed, and then click **OK**.
3. Save the configuration.

Removing files from a module

To remove a file from a module, complete the following steps:

1. Click **Applications** → **Application Types** → **WebSphere enterprise applications**, and click the link for the application to which the module belongs.
2. Click the **Manage Modules** link under the Modules section.
3. Select the module from which to remove the file, and click **Remove File**.
4. In the “Select the file to be removed” dialog box, select the file to be removed, and then click **OK**.
5. Save the configuration.

22.3.3 Performing multiple updates to an application or module

Multiple updates to an application and its modules can be packaged in a compressed file, in .zip or .gzip format, and uploaded to WebSphere Application Server. The uploaded file is analyzed, and the necessary actions to update the application are taken.

Depending on the contents of the compressed file, this method to update an application can replace files in, add new files to, and delete files from the installed application all in one single administrative action. Each entry in the compressed file is treated as a single file, and the path of the file from the root of the compressed file is treated as the relative path of the file in the installed application. To perform these multiple updates, the following conditions must be true:

- ▶ To replace a file, a file in the compressed file must have the same relative path as the file to be updated in the installed application.
- ▶ To add a new file to the installed application, a file in the compressed file must have a different relative path than the files in the installed application.
- ▶ To remove a file from the installed application, specify metadata in the compressed file using a file named META-INF/ibm-partialapp-delete.props at any archive scope.

The ibm-partialapp-delete.props file must be an ASCII file that lists files to be deleted in that archive with one entry for each line. The entry can contain a string pattern, such as a regular expression that identifies multiple files. The file paths for the files to be deleted must be relative to the archive path that has the META-INF/ibm-partialapp-delete.props file.

- ▶ To delete a file from the EAR file (not a module), include a META-INF/ibm-partialapp-delete.props file in the root of the compressed file. In the .props file, list the files to be deleted. File paths are relative to the root of the EAR file.

For example, to delete a file named docs/readme.txt from the root of the HelloApp.ear file, include the docs/readme.txt line in the META-INF/ibm-partialapp-delete.props file in the compressed file.

- ▶ To delete a file from a module in the EAR, include a `module_uri/META-INF/ibm-partialapp-delete.props` file in the compressed file. The `module_uri` part is the name of the module, such as `HelloWeb.war`.
For example, to delete `images/logo.gif` from the `HelloWeb.war` module, include the `images/logo.gif` line in the `HelloWeb.war/META-INF/ibm-partialapp-delete.props` file in the compressed file.
- ▶ Multiple files can be deleted by specifying each file on its own line in the metadata `.props` file.

Regular expressions can also be used to target multiple files. For example, to delete all JavaServer Pages (`.jsp` files) from the `HelloWeb.war` file, include the line `.*jsp` in the `HelloWeb.war/META-INF/ibm-partialapp-delete.props` file. The line uses a regular expression, `.*jsp`, to identify all `.jsp` files in the `HelloWeb.war` module.

As an example, assume we have prepared the compressed `HelloApp_update.zip` file shown in Figure 22-4.

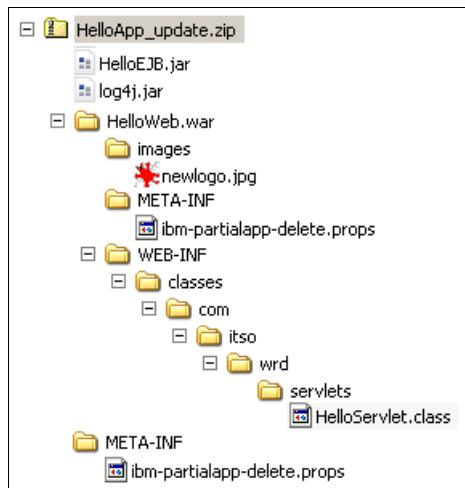


Figure 22-4 `HelloApp_update.zip` compressed file

The `META-INF/ibm-partialapp-delete.props` file contains the following line:
`docs/readme.txt`

The `HelloWeb.war/META-INF/ibm-partialapp-delete.props` contains the following line:
`images/logo.gif`

When performing the partial application update using the compressed file, WebSphere performs the following actions:

- ▶ Adds the `log4j.jar` file to the root of the EAR.
- ▶ Updates the entire `HelloEJB.jar` module.
- ▶ Deletes the `docs/readme.txt` file (if it exists) from the EAR file but not from any modules.
- ▶ Adds the `images/newlogo.jpg` file to the `HelloWeb.war` module.
- ▶ Updates the `HelloServlet.class` file in the `WEB-INF/classes/com/itso/wrd/servlets` directory of the `HelloWeb.war` module.
- ▶ Deletes the `images/logo.gif` file from the `HelloWeb.war` module.

To perform the actions specified in the HelloWeb_updated.zip file, complete the following steps:

1. Click **Applications** → **Application Types** → **WebSphere enterprise applications**. Select the application to update, and then click **Update**.
2. On the “Preparing for the application installation” window (Figure 22-1 on page 835), select the **Replace, add, or delete multiple files** option.
3. Select either the **Local file system** or **Remote file system** option, and click **Browse** to select the compressed file with the modifications that you have created. Click **Next**.
4. On the Updating Application window, click **OK**.
5. To update the configuration in the master repository, select the **Save** link.
6. If in a distributed server environment, make sure the **Synchronize changes with Nodes** option is selected so that the application is distributed to all nodes. Click **Save**. The application is distributed to the nodes, is updated, and is restarted as necessary.
7. If the application update changes the set of URLs that are handled by the application (that is, if servlet mappings are added, removed or modified), make sure that the web server plug-in is regenerated and propagated to the web server.

22.3.4 Rolling out application updates to a cluster

The Rollout Update feature allows you to roll out a new version of an application or part of an application to a cluster. The Rollout Update feature stops the cluster members, distributes the new application, synchronizes the configuration, and restarts the cluster members. The operation is done sequentially over all cluster members to keep the application continuously available.

When stopping and starting the cluster members, the Rollout Update feature works at the node level. Thus, all cluster members on a node are stopped, updated, and then restarted before the process continues to the next node.

Because the web server plug-in module cannot detect that an individual application on an application server is unavailable, the Rollout Update feature always restarts the application server that is hosting the application. Thus, if HTTP session data is critical to your application, either persist that data to a database or replicate it to other cluster members using the memory-to-memory replication feature.

The order in which the nodes are processed and the cluster members are restarted is the order in which they are read from the cell configuration repository. The Rollout Update feature cannot process the nodes and cluster members in any particular order.

Assume that we have an environment with two nodes, *aix-target1Node01* and *aix-target2Node01*, and a cluster called *cluster01*, which has one cluster member on each node (member1 on *aix-target1Node01* and member2 on *aix-target2Node01*). Assume also that we have an application named JSFSample that is deployed and running on the cluster.

To update this application using the Rollout Update feature, complete the following steps:

1. Click **Applications** → **Application Types** → **WebSphere enterprise applications**. Select the application to update, and then click **Update**.
2. On the “Preparing for the application installation” window, select the appropriate action, depending on the type of update. In this example, we update the entire application EAR file to a new version. So, select the **Replace the entire application** option.
3. Select either the **Local file system** or **Remote file system** option, and then click **Browse** to select the updated EAR file. Click **Next**.

4. Proceed through the remaining windows, and make any changes necessary. On the Summary window, click **Finish**.
5. After the application is updated in the master repository, the status window shown in Figure 22-5 opens.

ADMA5013: Application JSFSample installed successfully.

Application JSFSample installed successfully.

If you want to do a rolling update of the application on the cluster(s) on which it is installed, then click Rollout Update. A rolling update will save all changes made in this session to the master configuration, then synchronize and recycle the cluster members on each node, one node at a time.

[Rollout Update](#)

To start the application, first save changes to the master configuration.

The application might not be immediately available while being started on all servers.

Changes have been made to your local configuration. You can:

[Save](#) directly to the master configuration.

[Review](#) changes before saving or discarding.

To work with installed applications, click the "Manage Applications" link.

[Manage Applications](#)

Figure 22-5 Preparing for application rollout

You then have the following options to start the rollout action:

- Click the **Rollout Update** link.
- Click the **Manage Applications** link. Then, on the Enterprise Applications window, select the application, and click **Rollout Update**.

Saving directly to the master configuration: Do not click the **Save directly to the master configuration** link or otherwise save the configuration yourself. The Rollout Update feature does that task for you. If you save the configuration yourself, the Rollout Update action will be canceled, and it will be handled as a normal application update.

During the rollout, the window in Figure 22-6 opens in the status window.

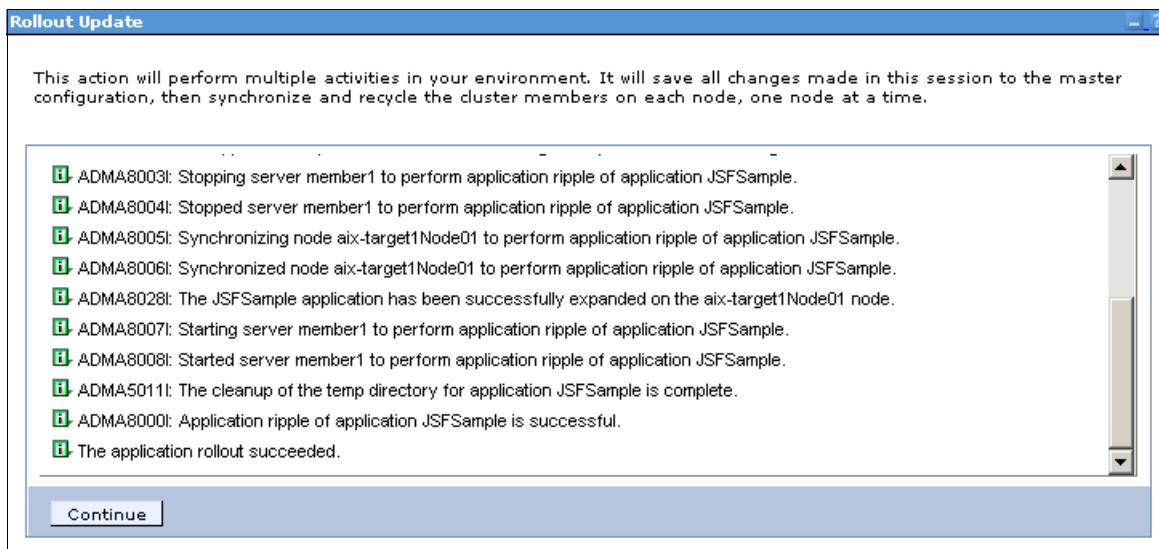


Figure 22-6 Rolling out an application

For each node, the cluster members are stopped, the application is distributed to the node, and the cluster members are restarted. When the rollout is complete (that is, when the “The application rollout succeeded” message displays), click **Continue**.

6. If the application update changes the set of URLs that are handled by the application (that is, if servlet mappings are added, removed, or modified), make sure that the web server plug-in is regenerated and propagated to the web server.

Automatic synchronization: The automatic file synchronization of the node agent is disabled temporarily during the rollout process and then re-enabled afterwards, if it was previously enabled. The Rollout Update feature works regardless of the automatic file synchronization setting. However, in production systems, the automatic synchronization is often disabled to give the administrator greater control over exactly when changes made to the cell configuration are distributed to the nodes.

Although the rollout update feature makes it easy to roll out an application to a cluster while keeping the application continuously available, make sure that your application can handle the rollout.

For example, assume that you have Version 1.0 of an application running in a cluster. The cluster consists of two application servers named server1 and server2, and HTTP session data is persisted to a database. When you roll out Version 2.0 of the application and server1 is stopped, the web server plug-in redirects users on server1 to server2. Then, when server1 is started again, starting Version 2.0 of the application, the plug-in distributes requests to server1 again. Now, if the application update incurred a change in the interface of any class that is stored in the HTTP session, when server1 tries to get these session objects from the database, it might run into a deserialization or class cast exception, preventing the application from working properly.

Another situation to consider is when the database structure changes between application versions, as when tables or column names change name or content. In that case, you might need to stop the entire application and migrate the database before you can deploy the new version. The Rollout Update feature is not suitable in this scenario.

So, it is important to understand the changes made to your application before rolling it out.

Note: If you require more advanced rollout and application versioning capabilities, consider WebSphere Virtual Enterprise.

WebSphere Virtual Enterprise extends an existing WebSphere infrastructure and brings, in addition to other features, functions to validate new versions of applications in production environments and then to roll out the new versions seamlessly with features to drain application servers from existing users before taking them offline for the update.

You can find information about WebSphere Virtual Enterprise at the following website:

<http://www-01.ibm.com/software/webservers/appserv/extend/virtualenterprise>

22.3.5 Hot deployment and dynamic reloading

Hot deployment and dynamic reloading characterize how application updates are handled when updates to the applications are made by directly manipulating the files on the server. In either case, updates do not require a server restart, although they might require an application restart. These features provide the following capabilities:

- ▶ Hot deployment of new components

Hot deployment of new components is the process of adding new components, such as WAR files, EJB JAR files, servlets, and JSP files to a running application server without having to stop and then restart the application server.

However, in most cases, such changes require the application itself to be restarted so that the application server run time reloads the application and its changes.

- ▶ Dynamic reloading of existing components

Dynamic reloading of existing components is the ability to change an existing component without the need to restart the application server for the change to take effect. Dynamic reloading can involve changes to the following types of changes:

- Implementation of an application component, such as changing the implementation of a servlet
- Settings of the application, such as changing the deployment descriptor for a web module

To edit the application files manually, locate the binaries in use by the server (in most cases these binaries will be in the application server `installedApps` directory). Although the application files can be edited manually on one or more of the nodes, these changes will be overwritten the next time the node synchronizes its configuration with the deployment manager. Therefore, perform manual editing of an application's files only in the master repository, which is located on the deployment manager system.

Tip: If you are not familiar with updating applications by manipulating the server files directly, consider using the administrative console Update wizard.

The following settings can affect dynamic reload:

- ▶ Reload classes when application files are updated

For application files to be reloaded automatically after an update, the “Override class reloading settings for Web and EJB modules” setting must be enabled, and the “Polling interval for updated files” setting must be greater than 0.

Click **Applications** → **Application Types** → **WebSphere enterprise applications**, and click the link for the application. In the Detail properties section, click the **Class loading and update detection** link.

- ▶ Application Server class loader policy

Set the application server’s class loader policy to *Multiple*. If it is set to Single, you will need to restart the application server after an application update.

Click **Servers** → **Server Types** → **WebSphere application servers**, and click the server name. The setting is found in the General Properties section.

- ▶ JSP Reload options for web modules

A web container reloads a web module only when this setting is enabled, which is a default setting.

Click **Applications** → **Application Types** → **WebSphere enterprise applications**, and click the link for the application. In the Web Module Properties section, click **JSP and JSF options**, and then select the **JSP enable class reloading** option. Enter a polling interval.

For more information about using hot deployment and dynamic reload, refer to the Information Center at the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.express.iseries.doc/info/iseriesexp/ae/trun_app_hotupgrade.html

22.4 Using a monitored directory

With WebSphere Application Server V8, you can deploy or update applications automatically by copying an application archive to a specified, *monitored directory*. Monitored directories support the following types of applications:

- ▶ Enterprise archive (EAR)
- ▶ Web archive (WAR)
- ▶ Java archive (JAR)
- ▶ Session Initiation Protocol (SIP) module (SAR)

The monitored directory feature is disabled by default.

IBM i: Using a monitored directory for application deployment is not supported on IBM i operating systems.

The default monitored directory for WebSphere Application Server V8 is *profile_root/monitoredDeployableApps/servers/server_name* directory for stand-alone servers. For distributed systems, the following directories are used:

Deployment managers use the following default monitored directories:

- ▶ *dmgr_profile_root/monitoredDeployableApps/servers/server_name* directory for all servers named *server_name*

- ▶ `dmgr_profile_root/monitoredDeployableApps/nodes/node_name/servers/server_name` directories for a specific `server_name` on a specific `node_name`
- ▶ `dmgr_profile_root/monitoredDeployableApps/clusters/cluster_name` directory for the members of `cluster_name`

Figure 22-7 shows a cell topology with two nodes (`aix-target1Node01` and `aix-target2Node01`). Each node has one stand-alone server (`server1`) and one server that is a member of a cluster (`member1` and `member2`).

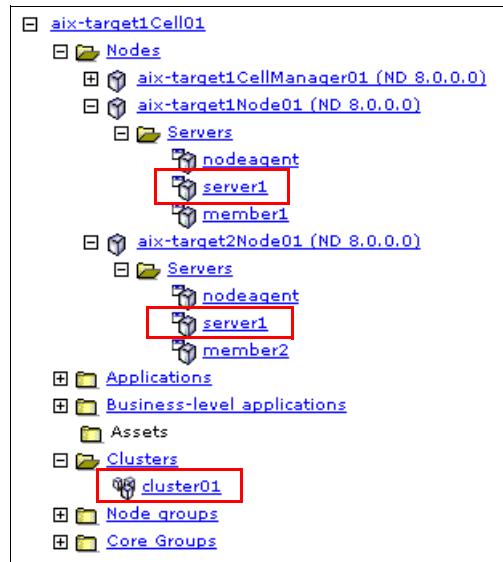


Figure 22-7 Sample topology with three possible run times

Figure 22-8 illustrates the monitored directory structure for the topology shown in Figure 22-7. The directories marked with boxes in Figure 22-8 are the directories where you can copy an application to deploy it on specified run time.

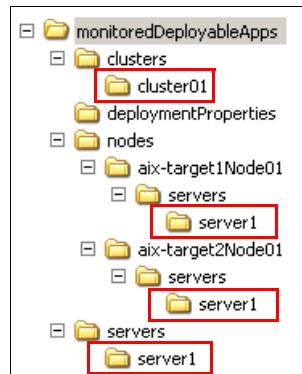


Figure 22-8 Sample monitored directory structure

22.4.1 Setting up a monitored directory

To enable and set up the monitored directory in WebSphere Application Server, complete the following steps:

1. In the administrative console, click **Applications** → **Global deployment settings**.
2. Select the **Monitor directory to automatically deploy applications** option, as illustrated in Figure 22-9. You can also change the default location of the monitored directory in the “Monitored directory” field.

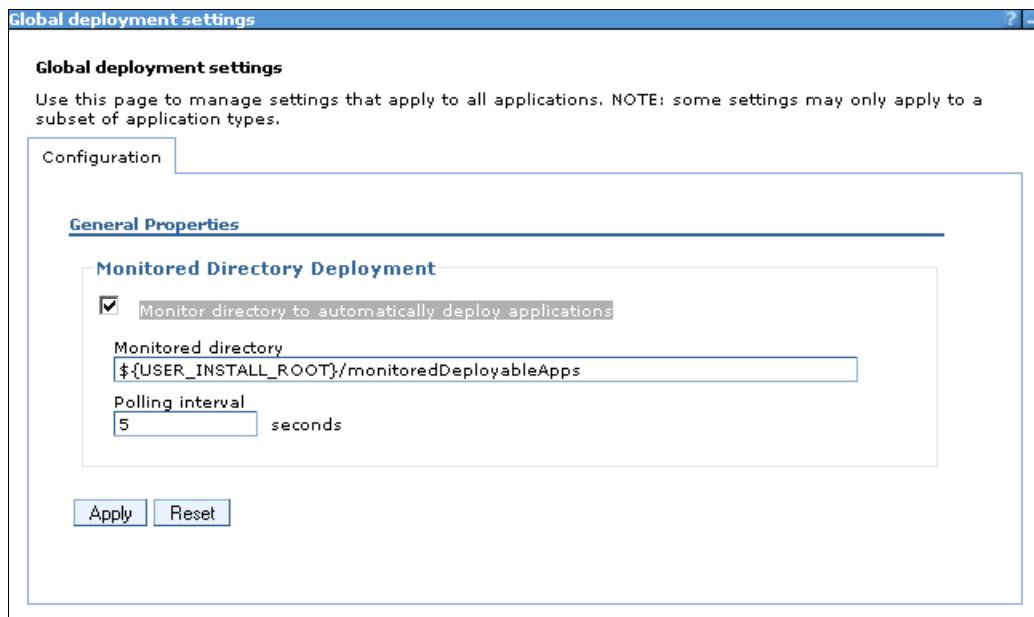


Figure 22-9 Configuring the monitored directory

Important: the WebSphere Application Server will not create a configured monitored directory for you. It must already exist on the system.

3. To change the standard polling interval setting, specify a number of seconds in the “Polling interval” field. Note that minimum value is 5 seconds. If you specify a lower or negative value, WebSphere Application Server will use 5 seconds automatically.
4. Click **Apply** to save the configuration.

For more information about configuring the monitored directory using **wsadmin** commands, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.nd.doc/info/ae/ae/trun_app_set_dragdrop.html

22.4.2 Working with a monitored directory

When an application is copied into the monitored directory, WebSphere Application Server creates a temporary copy of that application in another directory and installs it on the server. If you copy an application with a name that is already deployed on the server, it will be updated.

The monitored directory works only when the application server or cluster is running.

Binding values: Installing an EAR, JAR, WAR, or SAR file by adding it to a monitored directory does not change existing Java Naming and Directory (JNDI) or other application bindings. If you need to set binding values during deployment, install the file using the administrative console application installation wizard, a `wsadmin` script, or a properties file that sets the bindings (see “Deploying applications and adding properties files to a monitored directory” on page 849).

The following examples demonstrate how to work with a monitored directory. We use a sample JavaServer Faces 2.0 JSFSample.ear application. You can download this application from the WebSphere Application Server V8 samples site:

<http://publib.boulder.ibm.com/infocenter/radhelp/v7r5/index.jsp?topic=/com.ibm.tools.webtools.examples.doc/topics/webtoolsExamples.html>

Deploying and removing an EAR file on a stand-alone application server

To deploy and remove an EAR file on a stand-alone application server, complete the following steps:

1. Ensure that your server is running and that you configured the monitored directory.
2. Locate the monitored directory. For example, for a profile called AppSrv01, copy your files to the following directory:
install_root/profiles/AppSrv01/monitoredDeployableApps/servers/server1
install_root is the directory where you installed the WebSphere Application Server product.
3. Copy the JSFSample.ear application to the monitored directory.
4. Verify that the application was successfully deployed and started in the SystemOut.log file. Example 22-1 lists the key information that is logged during the application deployment.

Example 22-1 Installation application log when using a monitored directory

```
[7/12/11 17:28:43:274 GMT-06:00] 00000016 WatchService I CWLDD0007I: Event id 23253832-1. Start of processing. Event type: Added, File path: /opt/IBM/WebSphere/AppServer/profiles/AppSrvST/monitoredDeployableApps/servers/server1/JSFSample.ear.  
...  
[7/12/11 17:28:45:280 GMT-06:00] 00000016 AppManagement I CWLDD0014I: Event id 23253832-1. Installing application JSFSample.ear...ntrolOpDefs.xml is created.  
...  
[7/12/11 17:28:53:409 GMT-06:00] 00000016 AdminHelper A ADMN1009I: An attempt is made to start the JSFSample.ear application.  
...  
[7/12/11 17:28:54:982 GMT-06:00] 00000016 webcontainer I com.ibm.ws.webcontainer.VirtualHostImpl addWebApplication SRVE0250I: Web Module SampleAjax has been bound to default_host[:9084,:80,:9447,:5069,:5068,:443].  
...  
[7/12/11 17:28:55:841 GMT-06:00] 00000016 ApplicationMg A WSVR0221I: Application started: JSFSample.ear
```

You can also verify that the application is available in the administrative console from the **Application → Application Types → WebSphere enterprise application** view.

5. Invoke the application using a URL in a browser. For the JSFSample application, use the following URL:

`http://<your target host>:<your port>/SampleTemplating/page1.faces`

A page similar to that shown in Figure 22-10 displays in the browser.



Figure 22-10 JSFSample application output

After you deploy the application, note that the application still exists in the monitored directory. If you overwrite this file with a new file, the application is updated with your changes. However, the file must have the same name.

To uninstall the application from the server using the monitored directory, remove the JSFSample.ear file from the monitored directory. This removal triggers the monitoring service, and the application is uninstalled. Inspect the SystemOut.log file for information about the application status.

Deploying an EAR file on a federated node

To install and remove an application on a federated node, you can use the same procedure as we described previously in “Deploying and removing an EAR file on a stand-alone application server” on page 847. However, you use the deployment manager monitored directory instead of the stand-alone server monitored directory.

After configuring the monitored directory in deployment manager, copy the file to one of the following directories:

- ▶ The `dmgr_profile_root/monitoredDeployableApps/servers/server1` directory to deploy the application on all servers named server1
- ▶ The `dmgr_profile_root/monitoredDeployableApps/nodes/node01/servers/server1` directory to deploy the application only on server1 on node01.

Deploying an EAR file on a cluster

To install an application on a cluster, use the cluster directory in the deployment manager monitored directory. For example, if your cluster name is cluster01, to deploy an application to this cluster, copy the application to the following directory:

`dmgr_profile_root/monitoredDeployableApps/clusters/cluster01`

To update the application, overwrite it with a new EAR file with the same name. To remove the application from cluster, delete it from the monitored directory.

Note that if a server is a member of a cluster, you cannot deploy an application by copying it to its directory. If you do, you receive the following error:

A server named member1 was targeted by a monitored directory application deployment operation but was skipped because it is a member of cluster cluster01

Deploying applications and adding properties files to a monitored directory

Installing applications by copying them to a monitored directory does not change any bindings or JNDI configurations that the applications might use. To configure application resources and deploy an application using a monitored directory, use the properties file.

The monitoring service scans both the server directory and the property directory. If it finds a valid property file, it runs a the `wsadmin applyConfigProperties` command.

As an example, to install an application on cluster using a property file, complete the following steps:

1. Create the property file that defines the deployment task that you want to complete. In this example, we install the JSFSample application on a cluster using the file shown in Example 22-2.

Example 22-2 Example of property file that installs the JSFSample application

```
#Header
ResourceType=Application
ImplementingResourceType=Application

# Properties
Name=JSFSample
EarFileLocation=/tmp/install1/JSFSample.ear
TargetCluster=cluster01

EnvironmentVariablesSection

#Environment Variables
cellName=aix-target1Cell01
```

2. Ensure that the target server or cluster is running and that the monitored service is enabled and configured.
3. Copy the properties file to the deploymentProperties directory in the monitored directory.

WebSphere Application Server reads the property file and executes its tasks against its runtime environment. It then installs and configures the application.

Note that if you remove the property file from the monitored directory, the monitoring service notices this event but will not uninstall the application. Example 22-3 shows information that is logged when removing a property file from a monitored directory.

Example 22-3 Monitor service processing logs

```
[7/13/11 9:34:50:636 GMT-06:00] 0000002d WatchService I CWLDD0007I: Event id
2017945305-11. Start of processing. Event type: Deleted, File path:
/opt/IBM/WebSphere/AppServer/profiles/Dmgr8_01/monitoredDeployableApps/depl
oymentProperties/install_JSFSample.properties.
[7/13/11 9:34:50:636 GMT-06:00] 0000002d WatchService I CWLDD0061I: Event id
2017945305-11. The property file
```

```
/opt/IBM/WebSphere/AppServer/profiles/Dmgr8_01/monitoredDeployableApps/deploymentProperties/install_JSFsample.  
properties is deleted from the folder. No action is performed.  
[7/13/11 9:34:50:701 GMT-06:00] 0000002d WatchService I CWLDD0008I: Event id  
2017945305-11. End of processing.
```

To update an application, you can copy the property file and overwrite an existing property file (if not deleted). This action triggers the monitoring service to reinstall the application.

You have to uninstall an application manually or prepare a property file with uninstallation instructions. For the JSFSample application, you can use the file shown in Example 22-4.

Example 22-4 Example of property file that uninstalls the JSFSample application

```
#Header  
ResourceType=Application  
ImplementingResourceType=Application  
DELETE=true  
  
# Properties  
Name=JSFSample
```

To learn more about options and capabilities of using property files, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.nd.doc/info/ae/ae/trun_app_install_dragdrop_prop.html



Working with SCA applications

In this chapter, we discuss how to work with Service Component Architecture (SCA) applications. We introduce the basic elements of an SCA application and how to package and export SCA applications.

This chapter contains the following topics:

- ▶ SCA application introduction
- ▶ Preparing to use the sample application
- ▶ Packaging an SCA application for deployment
- ▶ Deploying an SCA application

Additional information about developing, testing, and deploying SCA applications can be found in the following publications:

- ▶ *Rational Application Developer for WebSphere Software V8 Programming Guide*, SG24-7835
- ▶ *Getting Started with WebSphere Application Server Feature Pack for Service Component Architecture*, REDP-4633

23.1 SCA application introduction

Support for SCA offers a simple and powerful way to construct applications based on service-oriented architecture (SOA). The support in WebSphere Application Server V8 uses the Apache Tuscany open-source technology to provide an implementation of the published SCA specifications. You can find information about Apache Tuscany at the following website:

<http://tuscany.apache.org/>

Figure 23-1 shows a technical overview of an SCA domain.

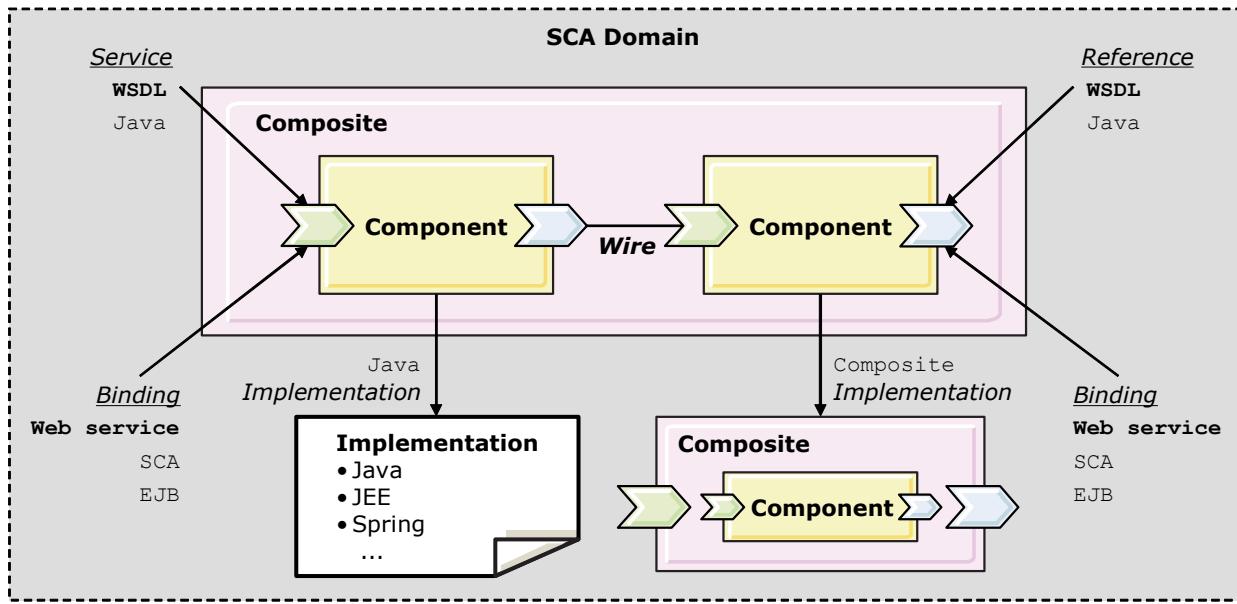


Figure 23-1 SCA technical overview

Briefly, the functions of these components are:

- ▶ SCA *components* implement business functions in the form of services.
- ▶ The SCA component is a configured instance of an *implementation*, which is program code that implements one or more business functions, such as Java classes.
- ▶ SCA components are grouped into *composites*, which are the deployment unit for an SCA in WebSphere Application Server.
- ▶ An SCA *domain* consists of the definitions of composites, components, their implementations, and the nodes on which they run.
- ▶ Components that are deployed into a domain can directly *wire* to other components within the same domain. The SCA domain is typically the cell on multiple-server installations and the server scope on single-server installations.

23.1.1 SCA component

Components both provide and consume services. Figure 23-2 shows part of an SCA component.

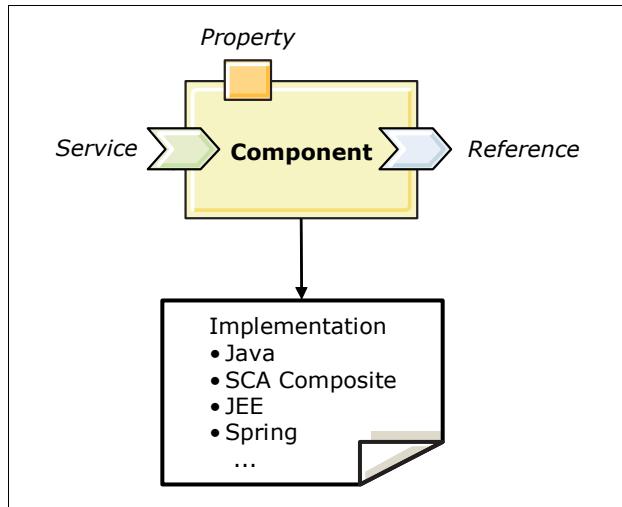


Figure 23-2 SCA component

An SCA component consists of the following parts:

- ▶ The service
 - The component provides a *service* or business function to its client.
- ▶ The reference
 - The component can have a *reference* to a service that is provided by another component.
- ▶ An implementation
 - The component contains a property that declares the implementation. The component reads the property value from the configuration file when the component is instantiated. In WebSphere Application Server V8, the service implementation includes the following components:
 - Java POJO
 - Java EE integration
 - Other SCA composites
 - Spring 2.5.5 containers

23.1.2 SCA composite

Assemblies of components are formally composed into *composites*. A composite is the set of components and wires (that is, the assembly of services). It is the basic deployment unit for SCA in WebSphere Application Server. The components, assemblies, internal wires, and service and reference definitions are written in an open XML language called Service Component Definition Language (SCDL).

The composite provides a scoping mechanism that defines a local boundary for components but that can also hide services that are provided in components that are not intended for other SOA applications. When defined, a composite can be reused to provide the implementation for other components in a nested fashion.

Services and references in a composite are bound to specific protocols (such as web services) using bindings. The bindings are part of the SCDL definition, and the business logic (implementation) does not need this detail.

As illustrated in Figure 23-3, an SCA composite includes the following elements:

- ▶ SCA components
One or more SCA components wired together.
- ▶ Interfaces
Services and references have an *interface* definition that points to the location of the WSDL or Java that describes the interface. The *service interface* provides the information that clients use to call the service. A *reference interface* provides the information that the component needs to call another service.
- ▶ Bindings
Services and references are bound to specific protocols through the usage of *bindings*. The use of bindings removes the details of connection from the business logic. The bindings supported in the WebSphere Application Server v8 are SCA, web service, EJB, JMS, Atom, and HTTP.
- ▶ Policies and intents
Policy and quality of service intents can decorate services or references (called *interaction intents*) and can decorate components (called *implementation intents*).

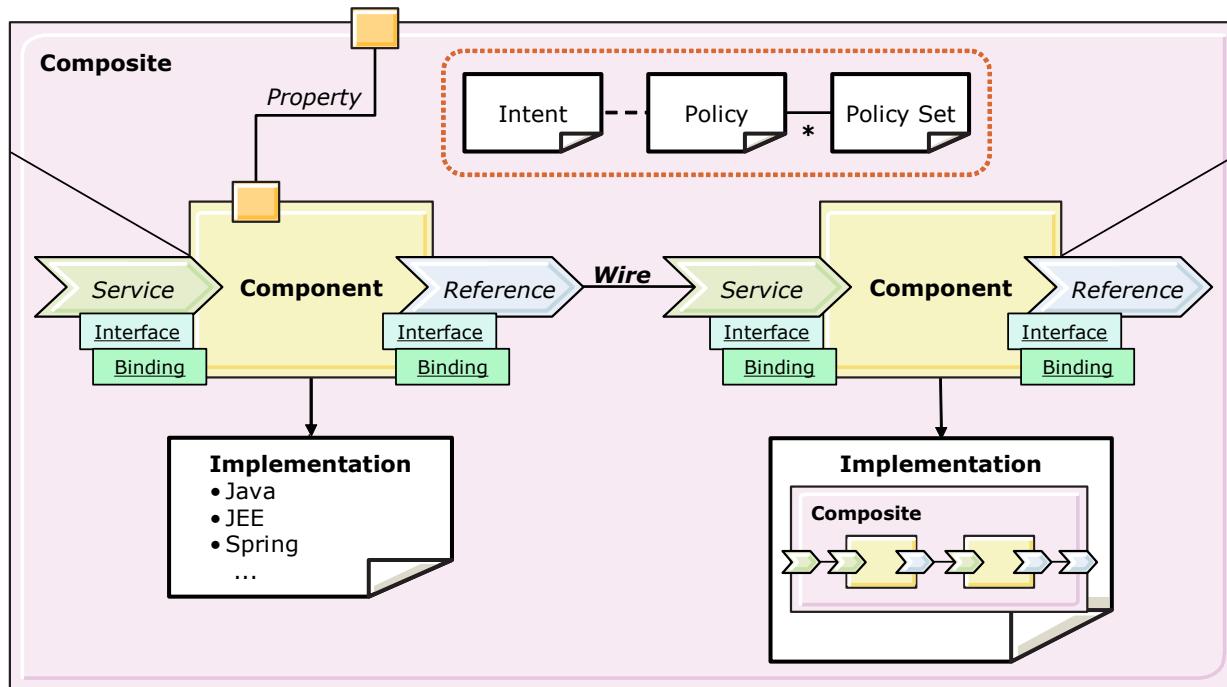


Figure 23-3 SCA composite

An application can contain one composite or several different composites. The components of a composite can run in a single process on a single computer or can be distributed across multiple processes on multiple computers. The components might all use the same implementation language or they can use different languages.

An SCA composite is typically described in a configuration file, the name of which ends in .composite. Figure 23-4 shows a composite definition named SupportFeeds.composite that is located in the project root folder of the AtomContentFeed composition unit in the RAD8AtomFeeds sample application.

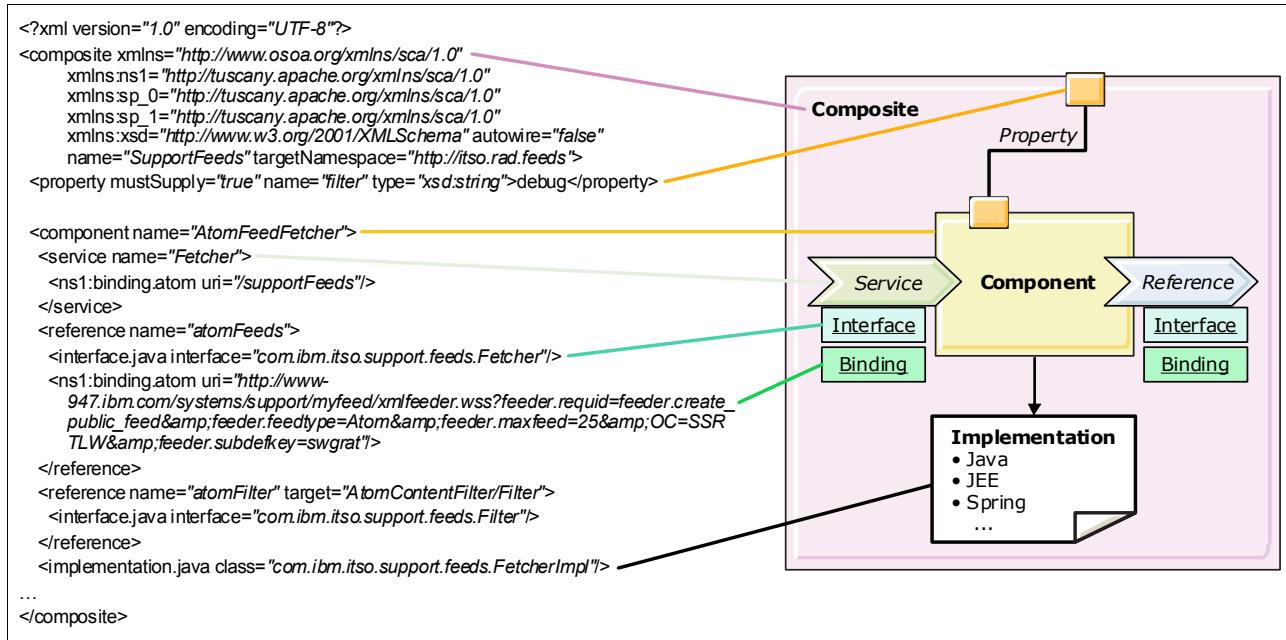


Figure 23-4 SCA composite file

Naming note: A composite file in a WAR file must be named default.composite. A composite file that is not in a WAR file can have any name.

23.1.3 SCA contribution

An SCA *contribution* contains artifacts that are needed for an SCA domain. Contributions are sometimes self-contained, in that all of the artifacts necessary to run the contents of the contribution are found within the contribution itself. However, the contents of the contribution can make one or many references to artifacts that are not contained within the contribution. These references might be to SCA artifacts or to other artifacts, such as Web Services Description Language (WSDL) files or XSD files or to code artifacts such as Java class files.

Figure 23-5 shows composites in an `sca-contribution.xml` file in an SCA domain.

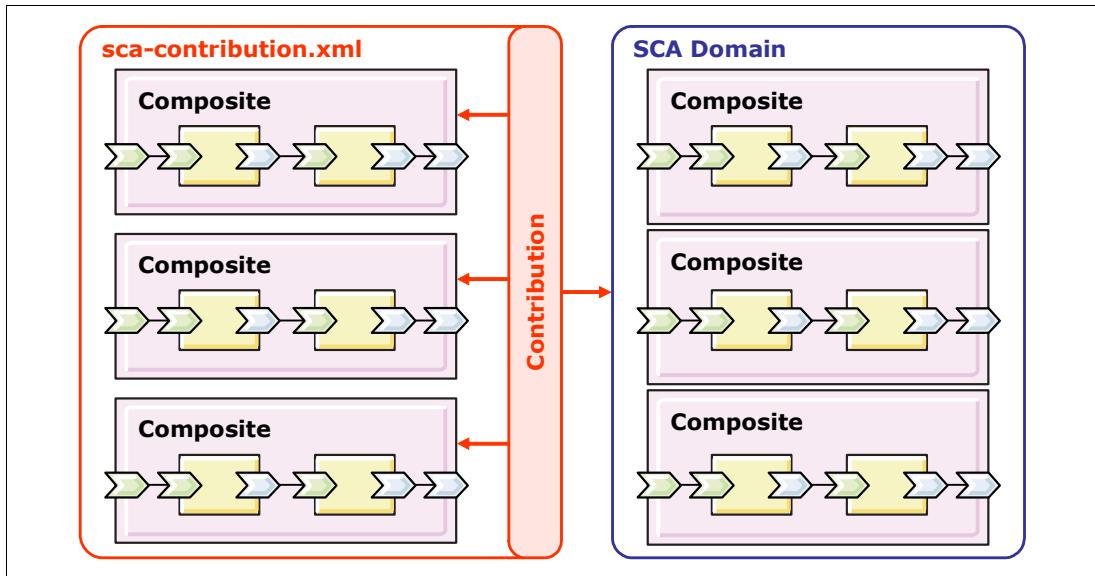


Figure 23-5 SCA contribution

An SCA contribution is typically described in a contribution file, named `sca-contribution.xml` in the META-INF directory. Example 23-1 shows the contribution file for the `SupportFeeds` composition unit.

Example 23-1 Contribution file

```
<?xml version="1.0" encoding="UTF-8"?>
<contribution xmlns="http://www.osoa.org/xmlns/sca/1.0">
    <deployable composite="ns1:SupportFeeds"
    xmlns:ns1="http://itso.rad.feeds"></deployable>
</contribution>
```

23.2 Preparing to use the sample application

The concepts in this chapter are illustrated using the RAD8AtomFeeds sample application, which is included with *Rational Application Developer for WebSphere Software V8 Programming Guide*, SG24-7835.

23.2.1 Downloading the application

To download the sample application:

1. Go to the following website
<http://www.redbooks.ibm.com/abstracts/sg247835.html?Open>
2. Click the **Additional Material** link.
3. Click the `sg247835.zip` file, and select **Save** to save the compressed file to your computer.
4. Extract the contents of the compressed file.

The SCA artifacts are in the `7835codesolution\sca` directory.

23.2.2 Importing the application to the development tool

To use the sample application for our exercise, import both files into IBM Assembly and Deploy Tools for WebSphere Administration by completing the following steps:

1. Start IBM Assembly and Deploy Tools for WebSphere Administration.
2. To import the code, click **File → Import** and then expand the **General** section. Select **Existing projects into workspace**. Click **Next**.
3. Click **Browse** next to the “Select Archive file” field, and browse to the sca directory where you extracted the sample code. Select the **RAD8AtomFeeds.zip** file, and click **Open**.
4. Click **Select All** to select all projects in the file. Then, click **Finish**.

23.2.3 Completing the service definition

The last action is to make sure the interface for the composite is complete by completing the following steps:

1. Open the composite by double clicking **SCA Content → Composites → SupportFeeds**.
2. Click the Service icon (green arrow) in the AtomFeedFetcher component, and then select the **Properties** view.
3. Switch to the Interface tab and verify that the Interface is of type Java and that it has the value **com.ibm.itso.support.feeds.Fetcher**. If not, correct the fields, and save and close the composite.

These steps are illustrated in Figure 23-6.

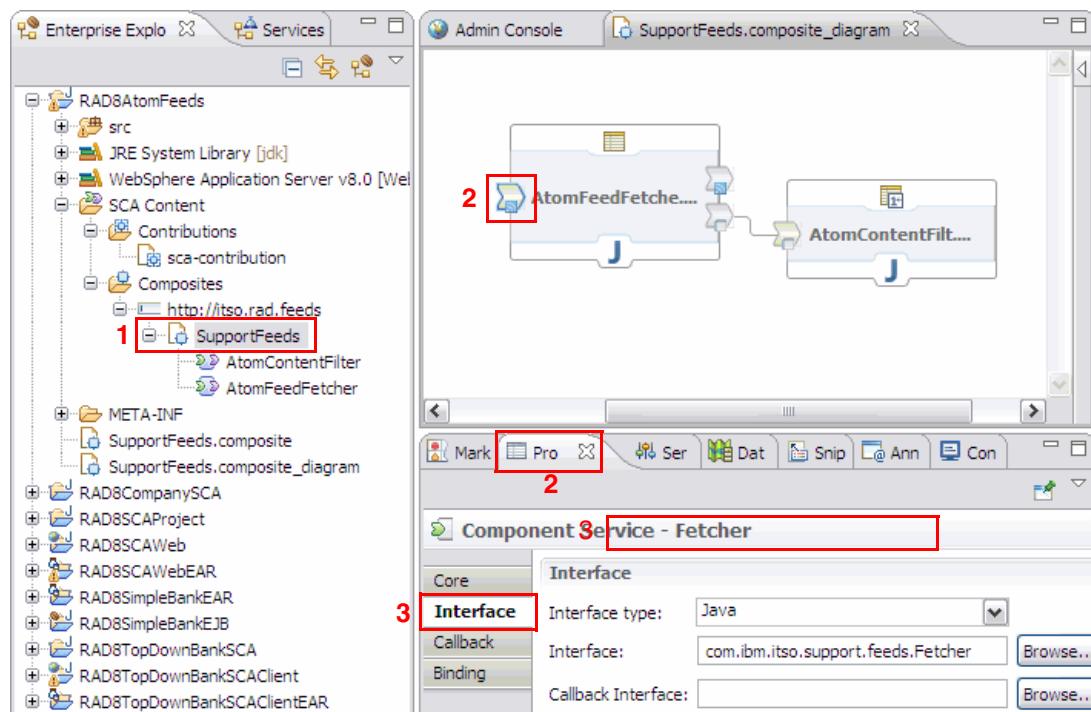
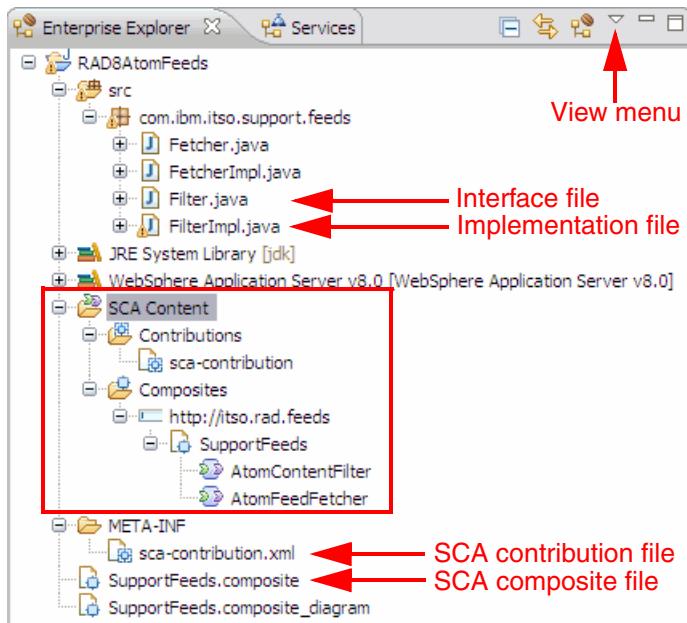


Figure 23-6 Open the *SupportFeeds* composite

23.3 Packaging an SCA application for deployment

Figure 23-7 shows the files that are contained in the sample SCA application package, as seen in the Enterprise Explorer view in IBM Assembly and Deploy Tools for WebSphere Administration.



View menu

Interface file

Implementation file

SCA contribution file

SCA composite file

Figure 23-7 SCA application package

Tip: SCA applications are displayed in the Enterprise Explorer view of the IBM Assembly and Deploy Tools for WebSphere Administration with features in a manner that makes working with the components easy. The SCA Content section (highlighted in the figure) contains entries for the contributions and composites, making them easily visible and selectable to open for editing. This is simply a structural view of the files. The SCA contribution file, SCA composite, and SCA composite diagram files (all located below the META-INF folder in Figure 23-7) are not normally visible as raw files.

We changed the view properties for the Enterprise Explorer view to show them to you in this format. So, you see these files twice, once by file name, and once as an element under the SCA Content folder. Double-clicking **SCA Content/Contributions/sca-contribution** opens the same file that would open if you double-clicked **META-INF/sca-contribution.xml**. The same is true for the composite.

To use these settings, click the **View** menu. Then click **Customize view** Click the **Filter** tab and clear **SCA resources**.

The composites are created by the developer as part of the application development process. The developer adds components to the composite, writes the implementation code for the components, and adds the interface files. These development activities are independent of deployment activities. The contribution file is related to packaging for deployment and must be created to deploy the application.

You can see the composite and components in Figure 23-8. Selecting any of the components in the composite shows the properties in the lower window. In the figure, the AtomFeedFetcher component is selected, and in the properties window, you can see the implementation class listed.

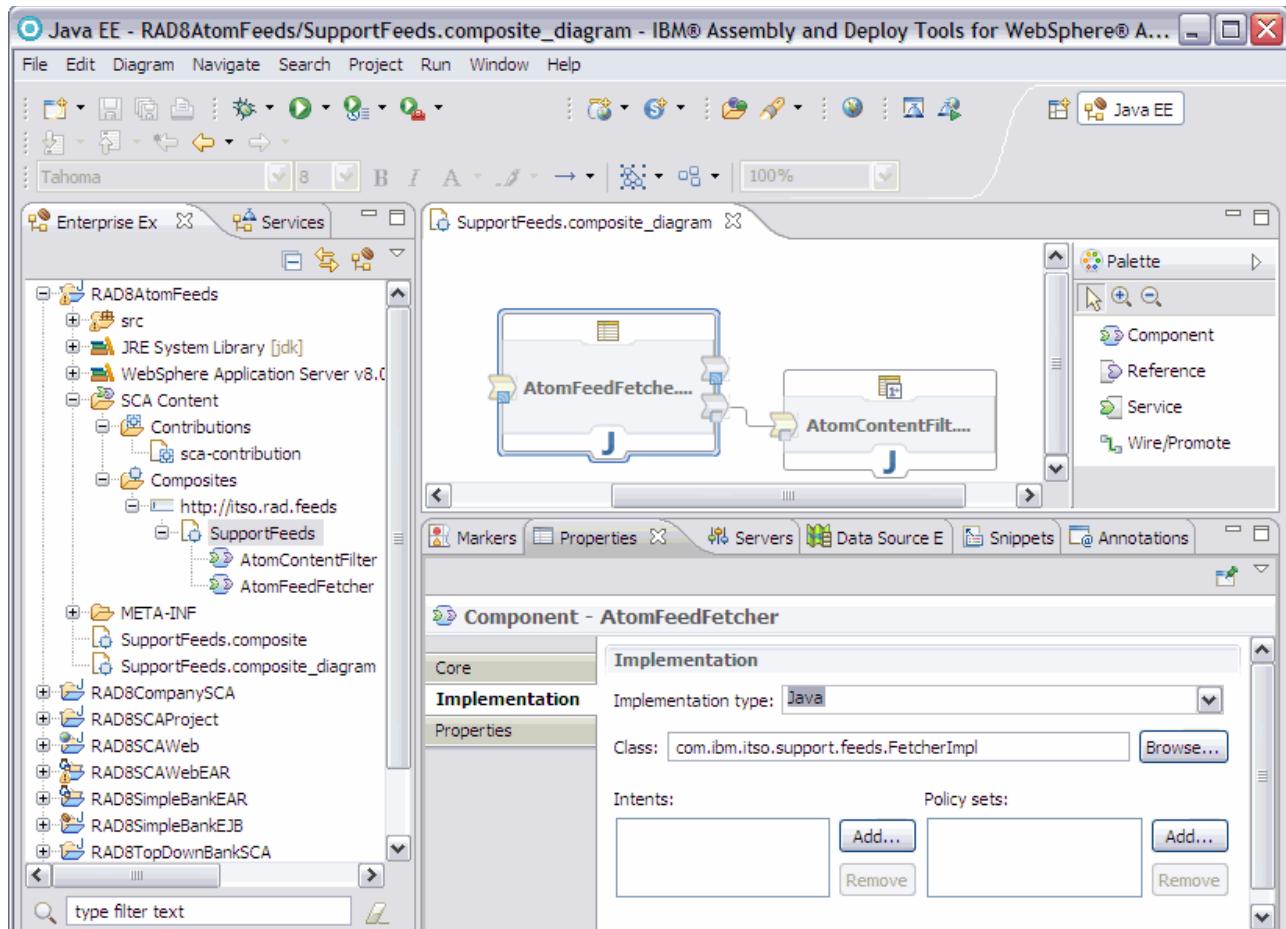


Figure 23-8 SupportFeeds composite

23.3.1 Creating the contribution

Before you can export the project for deployment, you need to create the contribution file. This is a simple process using the IBM Assembly and Deploy Tools for WebSphere Administration.

Complete the following steps:

1. In the Enterprise Explorer, expand **SCA Content**. Right-click **Contributions** and click **New → SCA Contribution**. (Figure 23-9).

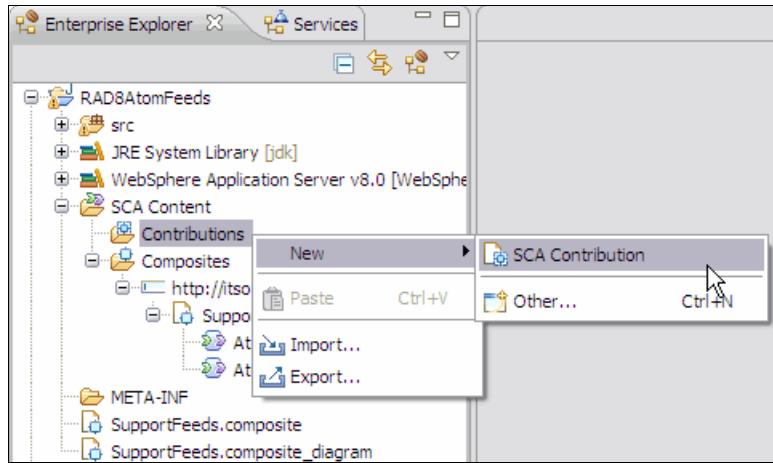


Figure 23-9 Contribution file creation

2. In the New Contribution Wizard window, select the composites to be deployed (Figure 23-10) and click **Finish**.

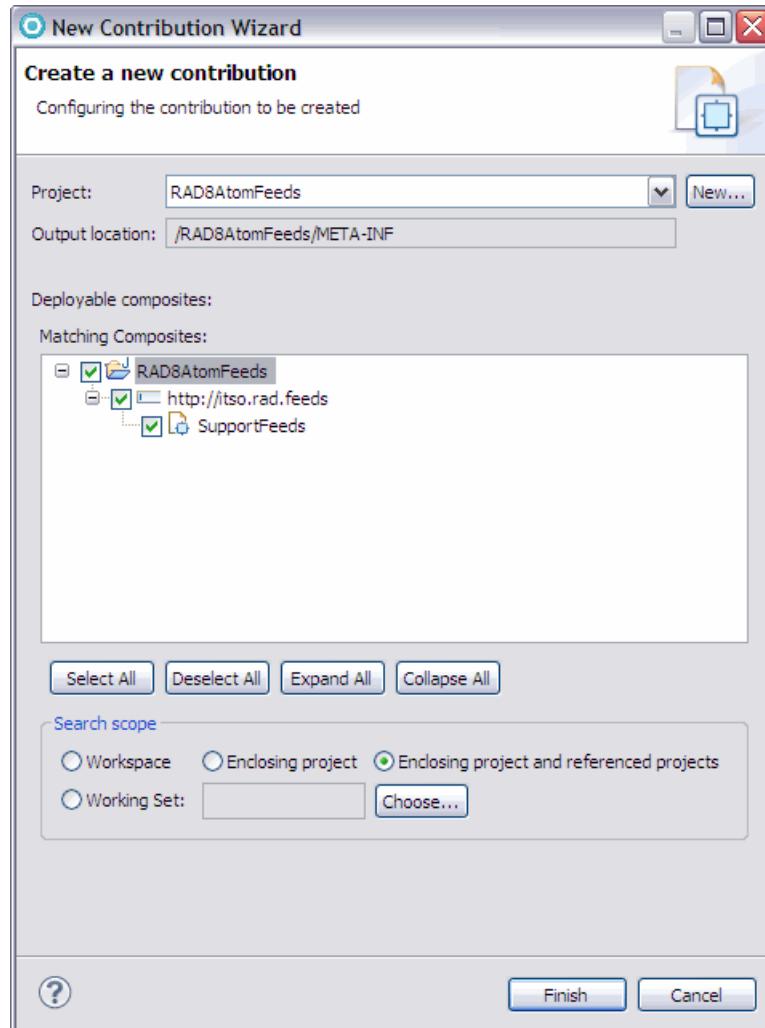


Figure 23-10 Add composites to the contribution

23.3.2 Exporting the SCA application for deployment

To export the contribution as an SCA archive file, complete the following steps:

1. Right-click the project and select **Export**, as shown in Figure 23-11.

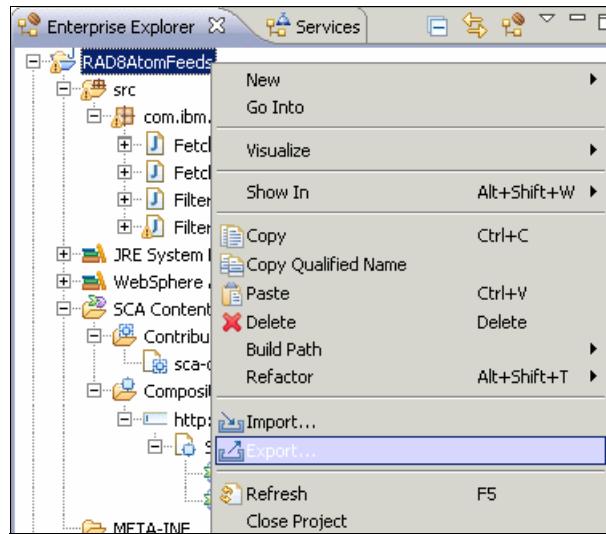


Figure 23-11 Exporting an SCA application

2. In the Export dialog box, select **Service Component Architecture → SCA Archive File**, as shown in Figure 23-12, and then click **Next**.

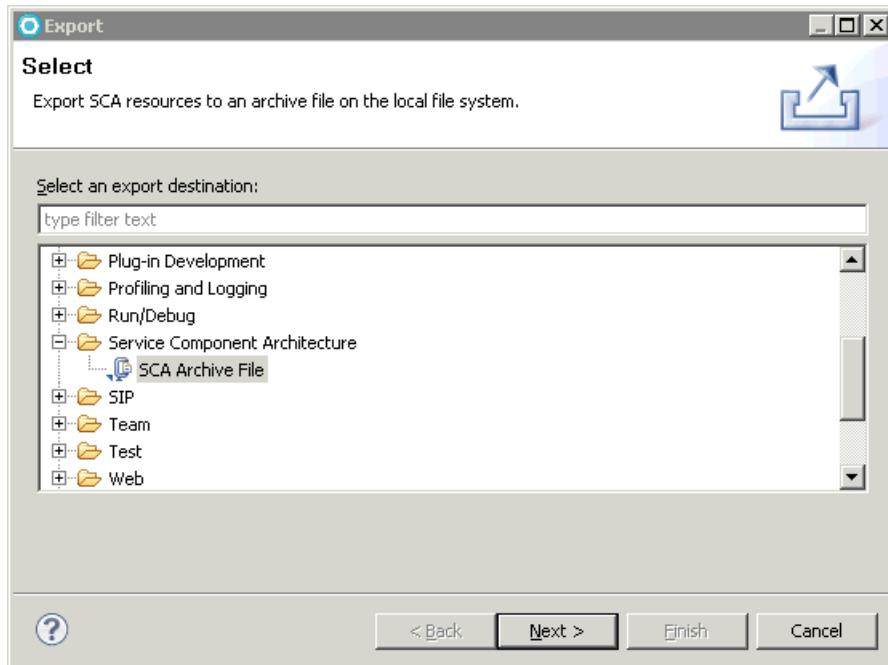


Figure 23-12 Selecting the archive file

3. Select the project to export (Figure 23-13) and click **Finish**.

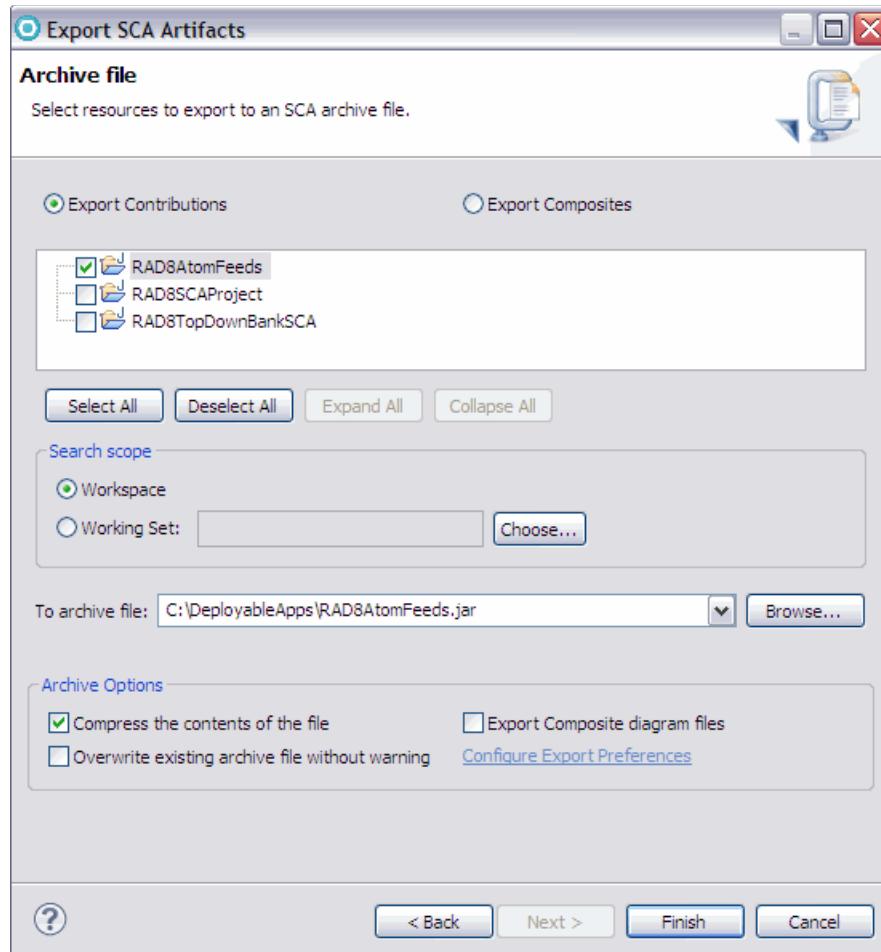


Figure 23-13 Exporting project

Now, you can deploy the RAD8AtomFeeds.jar SCA archive file.

23.4 Deploying an SCA application

A typical SCA solution consists of a combination of web, EJB, and SCA applications. The web and EJB modules are deployed as enterprise applications, and the SCA composites are deployed as assets in a business-level application.

This section describes how to deploy an SCA composite as an asset in a business-level application.

23.4.1 Importing the SCA archive file as an asset

The first step in deploying the application is to import the SCA archive file into the application server environment as an asset.

Complete the following steps:

1. Click **Applications** → **Application Types** → **Assets**. Then, in the Assets window, shown in Figure 23-14, click **Import**.

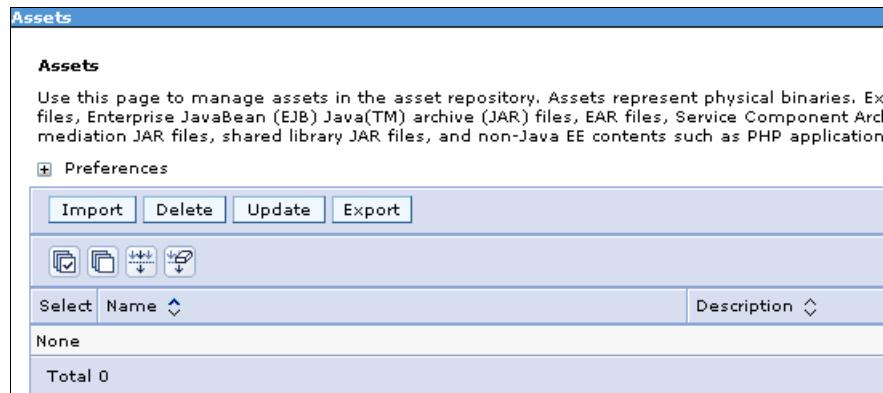


Figure 23-14 Import an asset

2. Select the SCA archive file, as shown in Figure 23-15, and then click **Next**.

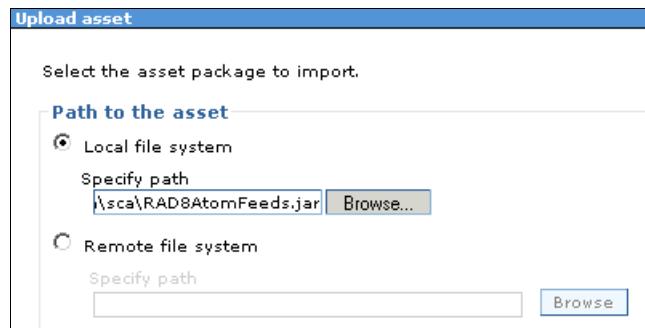


Figure 23-15 Identify the jar file location

3. Provide the required options. In this example, take the default values, as shown in Figure 23-16.

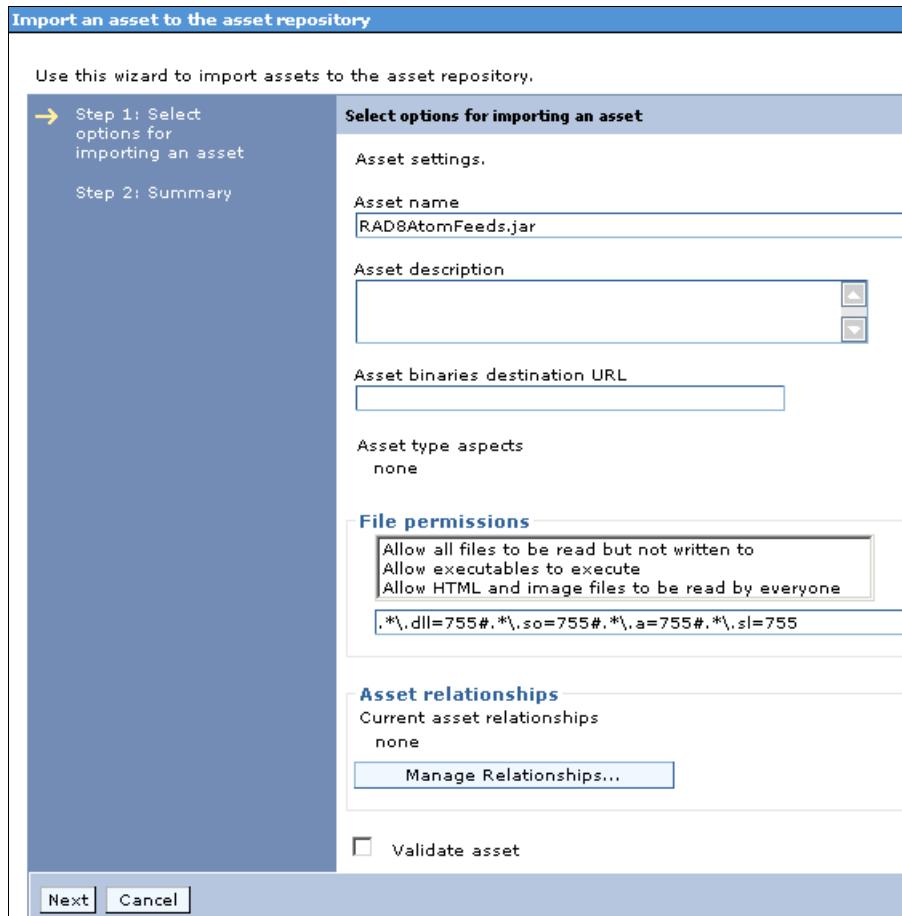


Figure 23-16 Option for importing an asset

Fields to note are:

- Asset binaries destination URL: When the asset is imported, the binaries are extracted by default to *profile_root*/installedAssets/*asset_name*/BASE. You can change this location by specifying a new one here.
- File permissions: The default setting for the file permissions to be assigned the extracted asset binaries is the “Allow executables to execute setting”. You can select another option, or provide a customized setting.
- Asset relationships: Click the **Manage Relationships** button to specify assets to which this asset is related.
- Validate asset: Selecting this option will enable validation of the references specified in the asset when it is imported. The asset is examined to ensure that references, for example, data sources or references contained in deployment descriptors (resource and resource environment references) are defined in the scope of the deployment target of the asset.

Click **Next**.

4. In the Summary page, click **Finish** to import the file, and save the modification to the master configuration. The new SCA archive file is now listed as an asset, as shown in Figure 23-17.



Figure 23-17 Asset imported

23.4.2 Creating a new business-level application

To create a new business-level application, complete the following steps:

1. Click **Applications** → **Application Types** → **Business-level applications**. In the Business-level applications window, shown in Figure 23-18, click **New**.

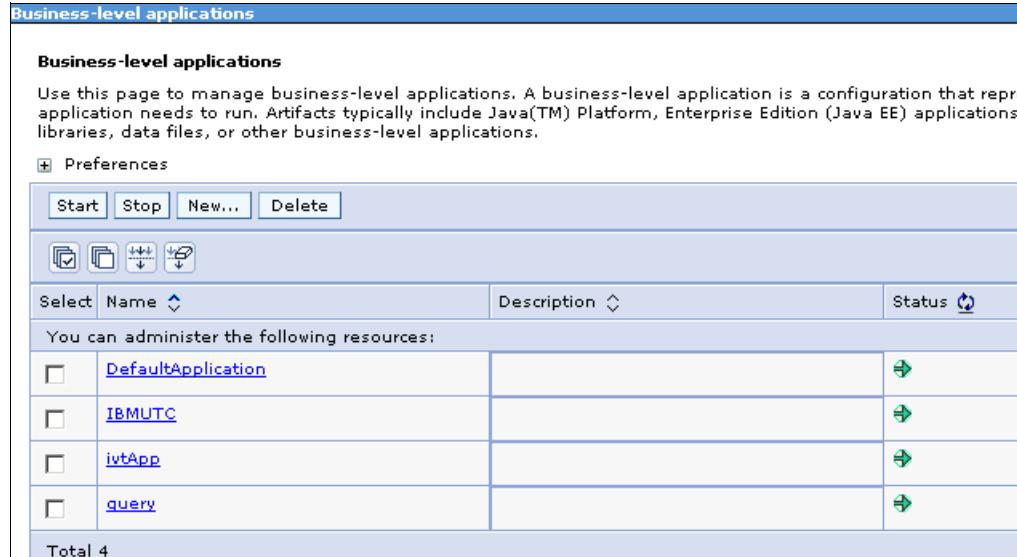


Figure 23-18 Business-level application window

- Enter a new name for the business-level application, as shown in Figure 23-19, and then click **Apply**. Save the change to the master configuration.

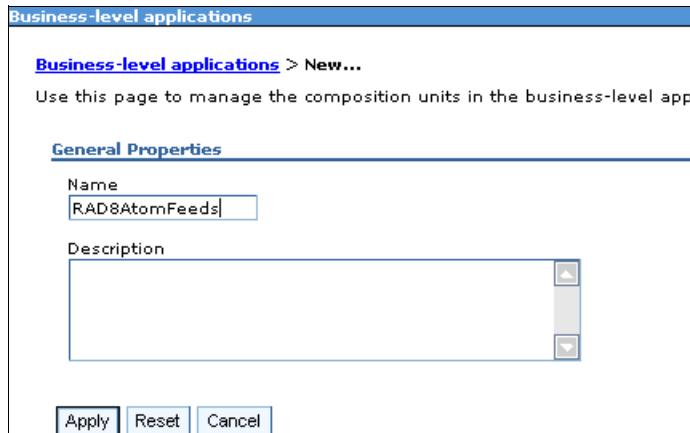


Figure 23-19 Create a new business-level application

23.4.3 Adding the new asset to the business-level application

Add the new asset as a composition unit to the RAD8AtomFeed business-level application by completing the following steps:

- Click **Applications** → **Application Types** → **Business-level applications**, and then click **RAD8AtomFeeds**, as shown in Figure 23-20.

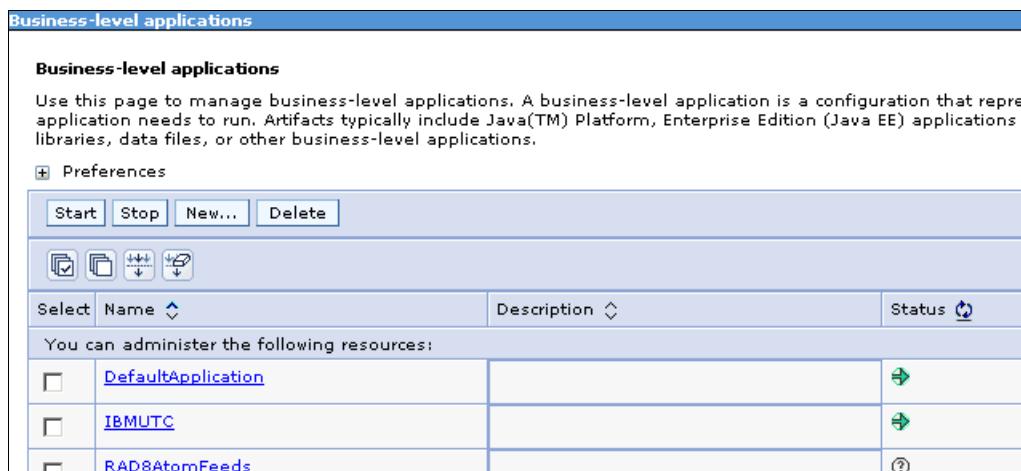


Figure 23-20 Click the business-level application

- In the Deploy assets section, click **Add** → **Add Asset**, as shown in Figure 23-21.

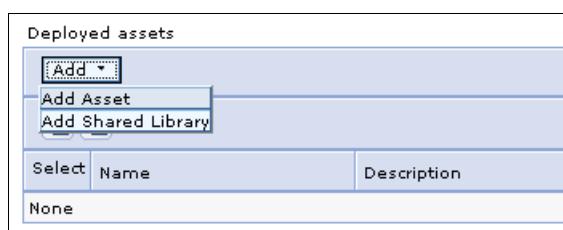


Figure 23-21 Add asset

3. Choose the asset that you want to deploy to this business-level application, as shown in Figure 23-22, and click **Continue**.



Figure 23-22 Choose the asset

4. Review the options for the composite unit, as shown in Figure 23-23. In this example, we use the default values.

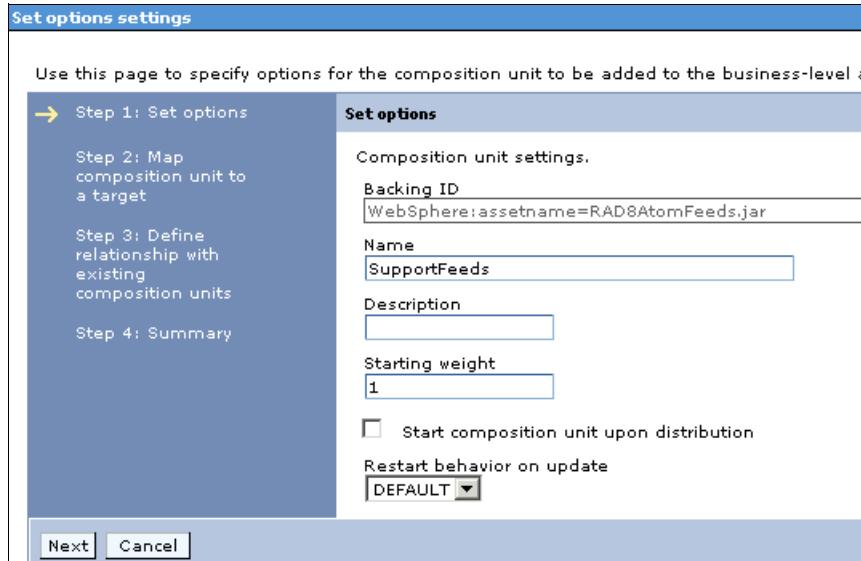


Figure 23-23 Review the options

The fields on this window are:

- Backing ID: Displays the unique identifier for the composition unit that will be registered in the application domain. You cannot change this field.
- Name: Name for the composition unit.
- Starting weight: Specifies the order in which composition units are started when the server starts. The composition unit with the lowest number is started first.
- Start composition unit upon distribution: Specifies whether to start the composition unit when it is distributed to other locations. This setting applies only to assets and shared library composition units.
- Restart behavior on update (of the composition unit):
 - ALL: Restarts the composition unit after the entire composition unit is updated.
 - DEFAULT: Restarts the composition unit after the part of the composition unit is updated.

- NONE: Does not restart the composition unit after the composition unit is updated.

Click **Next**.

- The next window allows you to select the target server for deployment. Select the server in the Available column and click the right arrow to move it to the Selected column. Click **Next**.

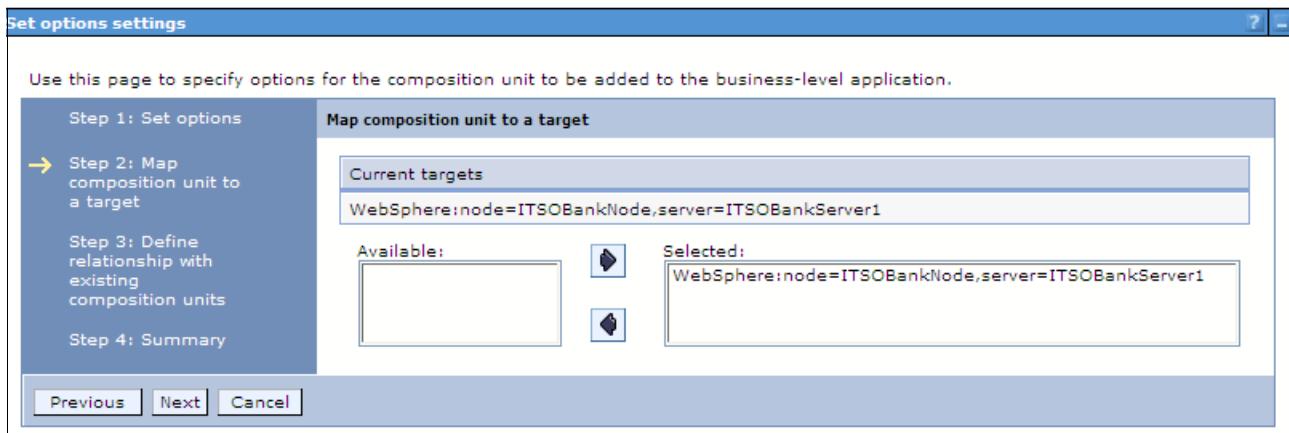


Figure 23-24 Map the composition unit to a target

- The next window allows you to manage relationships with existing composition unit. A relationship declares a dependency of this composition unit to another asset deployed as a shared library. In this example, we have none to declare. Click **Next**.

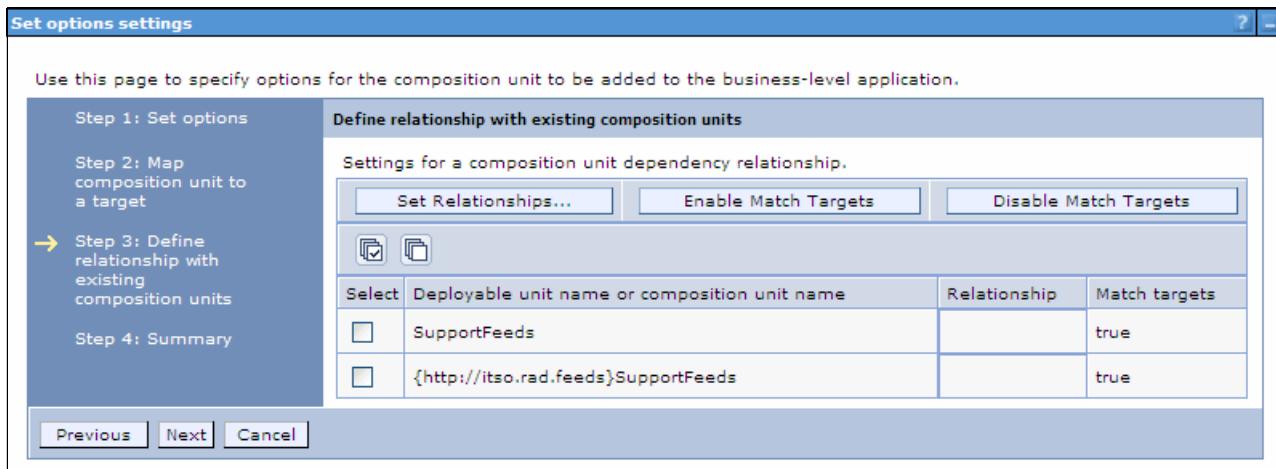


Figure 23-25 Define relationships

- The last window is a summary window. Review your selections, then click **Finish**.
- Save the update to the master configuration.

23.4.4 Starting and verifying the business-level application

The last step is to start the business-level application and to verify that it is available by completing the following steps:

1. Click **Applications** → **Application Types** → **Business-level applications**, choose the **RAD8AtomFeed** application, and then click **Start**. The application starts, as shown in Figure 23-26.

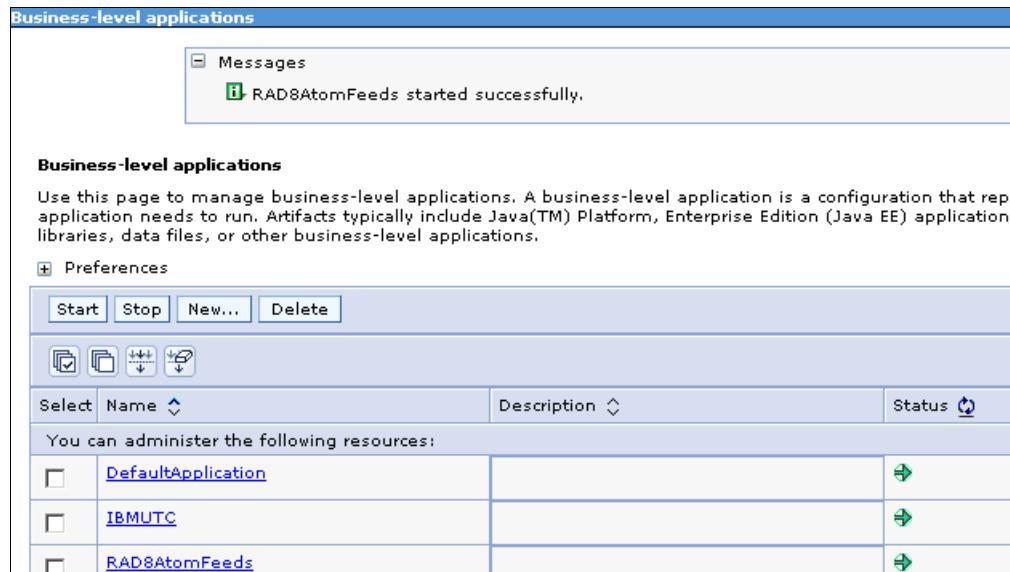


Figure 23-26 Business-level application started

2. Verify the offered service from a web browser by entering the following URL:

`http://hostname:port/supportFeeds`

Typically, the *hostname* is localhost and the *port* has the value 9080.



Working with OSGi applications

This chapter provides information about WebSphere Application Server V8 OSGi capabilities. We describe what an OSGi application is, the OSGi life cycle, and how to package and prepare an OSGi application for deployment to WebSphere Application Server.

A detailed explanation of the OSGi framework is out of the scope of this publication. Refer to the following sources for more information:

- ▶ The home page of the OSGi alliance:
<http://www.osgi.org>
- ▶ Eclipse Equinox OSGi implementation:
<http://eclipse.org/equinox>
- ▶ The WebSphere Application Server V8 Information Center:
http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.osgi.doc/topics/ca_about.html

This chapter includes the following topics:

- ▶ OSGi overview
- ▶ Using the sample application
- ▶ Packaging OSGi applications
- ▶ Exporting OSGi applications
- ▶ Deploying OSGi applications
- ▶ Updating OSGi applications

24.1 OSGi overview

OSGi is a framework based on Java technology that enables you to write modular and dynamic applications. There are several OSGi implementations. WebSphere Application Server V8 uses the Eclipse Equinox project and OSGi V4.2.

One of the OSGi goals is to provide a greater level of modularity than standard Java between the application and class levels. OSGi modules also can be resolved dynamically and can depend on other OSGi resources in a highly dynamic manner.

24.1.1 OSGi application model

To provide a dynamic model and greater modularity for applications, OSGi introduces a concept called a *bundle*. A bundle is a standard JAR file with additional metadata in the manifest file, as shown in Example 24-1. Because the standard Java environment ignores these additional properties, you can also use an OSGi bundle with classic Java applications.

Example 24-1 Example of an OSGi descriptor file

```
Manifest-Version: 1
Bundle-ManifestVersion: 2
Bundle-SymbolicName: my.very.useful.library
Bundle-Version: 42.0.0
Import-Package: org.osgi.framework;version="1.0.0,2.0.0)"
Export-Package:
my.very.useful.library.stringops;version=23.2.1,my.very.useful.library.interop;ver
sion=5.0.0
```

The following additional properties are shown on Example 24-1:

► **Bundle-ManifestVersion**

This header must be set to “2” to indicate that the bundle is written to revision 4 or later of the OSGi specification, rather than previous revisions.

► **Bundle-SymbolicName** and **Bundle-Version**

These two headers define the identity of the module. Every bundle in an OSGi system has a unique identity that is determined by its symbolic name and version. The **Bundle-Version** header is optional. If the header is not present, the bundle version defaults to “0.0.0”.

► **Import-Package**

This header defines the packages that are visible to a bundle. A bundle always has access to all `java.*` packages, all the packages inside the bundle, and depending on the OSGi framework configuration, the `javax.*` packages that are part of the JDK. All other packages must be imported.

Every package import carries a version range that defines the accepted versions of the dependencies. This version range is entirely independent on the bundle version.

Example 24-1 declares that the bundle needs only `org.osgi.framework` between version 1.0.0 (inclusive, denoted by a square bracket) and 2.0.0 (exclusive, denoted by a round bracket) in addition to its own classes and the `java.*` classes. If no version range is specified, which is not a best practice, all versions of the package are allowed. Note that “1.0.0” is a version range of Version 1 and above, not an exact version number.

- ▶ Export-Package

This header declares the packages that are visible outside the bundle. Only the specified packages can be used by other bundles. Every exported package carries a version, which defaults to “0.0.0” if unspecified.

OSGi applications versioning

It is important to understand the versioning notation of OSGi applications because it is the foundation for the OSGi modules *dependencies* and *capabilities*. Dependencies have version ranges, and capabilities have just versions. Without these versions, a bundle is on one of the following extremes regarding changes to a dependency:

- ▶ Accept only a single version of a dependency, and fail to use newer compatible versions.
- ▶ Accept all versions of a dependency, and suffer from incompatible changes being made to the dependency.

With versioning, bundles can allow different versions of a dependency in a window of compatibility. To allow versioning, OSGi versions must be semantic versions, meaning that they carry commonly understood meanings. A bundle author can rely on these meanings when declaring the versions of a bundle’s dependencies.

This versioning capability provides the ability to write modules today that can interoperate with current libraries and with future versions of those libraries but that will fail to resolve against incompatible future versions of those libraries. However, to reap the full benefits a bundle, the author needs to understand and follow the OSGi semantic versioning practices. OSGi distinguishes between bundle versions and package versions. Versioning a bundle does not version the packages and vice versa. Both follow the following form:

`<major>. <minor>. <micro>. <qualifier>`

- ▶ The `<qualifier>` component carries no semantics and is often used to denote build numbers. In Rational Application Developer, the qualifier can be replaced by a build time stamp during export.
- ▶ A change in the major version denotes a breaking API change, for example, the parameters of a method changed. Such a change is breaking because a client cannot work (non-reflectively) with both the previous and the new version of the API.
- ▶ A change in the minor version denotes a backwards compatible API change, for example, adding a new method to an interface. Existing clients can function both with the old and new versions. Implementations must be updated to support the new method.
- ▶ A change in the micro version denotes a bug fix, and no change to the API is allowed.

In addition to single versions, OSGi defines the concept of version *ranges*, for example, as used on Import-Package statements. Version ranges come in the following forms:

- ▶ `[1.0.0, 2.0.0)`

Defines a version range from 1.0.0 (inclusive) to 2.0.0 (exclusive). A square bracket denotes an inclusive endpoint, whereas a round bracket denotes an exclusive endpoint.

- ▶ `1.0.0`

Defines an open version range of 1.0.0 and above. Do not confuse open version ranges with single versions. Whether the version is a single version or a range is determined by the context. For example, the Export-Package and Bundle-Version statements have single versions, whereas the Import-Package and Application-Content statements have version ranges.

Consideration: Despite being a good fit for most development scenarios, the OSGi version policy is not appropriate in all cases. For example, OSGi versions are conceived as linear, that is, Version 1.7 must contain all the features of Version 1.6 (otherwise, a breaking change occurs and Version 1.7 would actually be Version 2.0). As a consequence, you can introduce new features only on the very latest version within one major package version. You cannot introduce features at earlier versions.

This limitation can be a problem with large products that maintain a stable API (for example, infrequent major version changes) where you want to combine the product with multiple minor versions that are supported and actively extended.

Also note that it is up to the developer to adjust package versions.

24.1.2 OSGi application life cycle

All artifacts in OSGi are equipped with support of dynamics in the form of defined life cycles. At the beginning of the life cycle, a bundle is installed into an OSGi runtime environment called the *OSGi framework*. A bundle that is installed is known to the framework, but otherwise is useless. A bundle whose dependencies can all be satisfied moves to the resolved state. Usually, a bundle further transitions to an active state, either when any class from the bundle is loaded or when some external agent starts the bundle. At the end of its life, a bundle can be uninstalled from the framework again. However, when uninstalling a bundle, OSGi ensures that any packages that the bundle provides to other still resolved or active bundles remain available. Figure 24-1 illustrates these states.

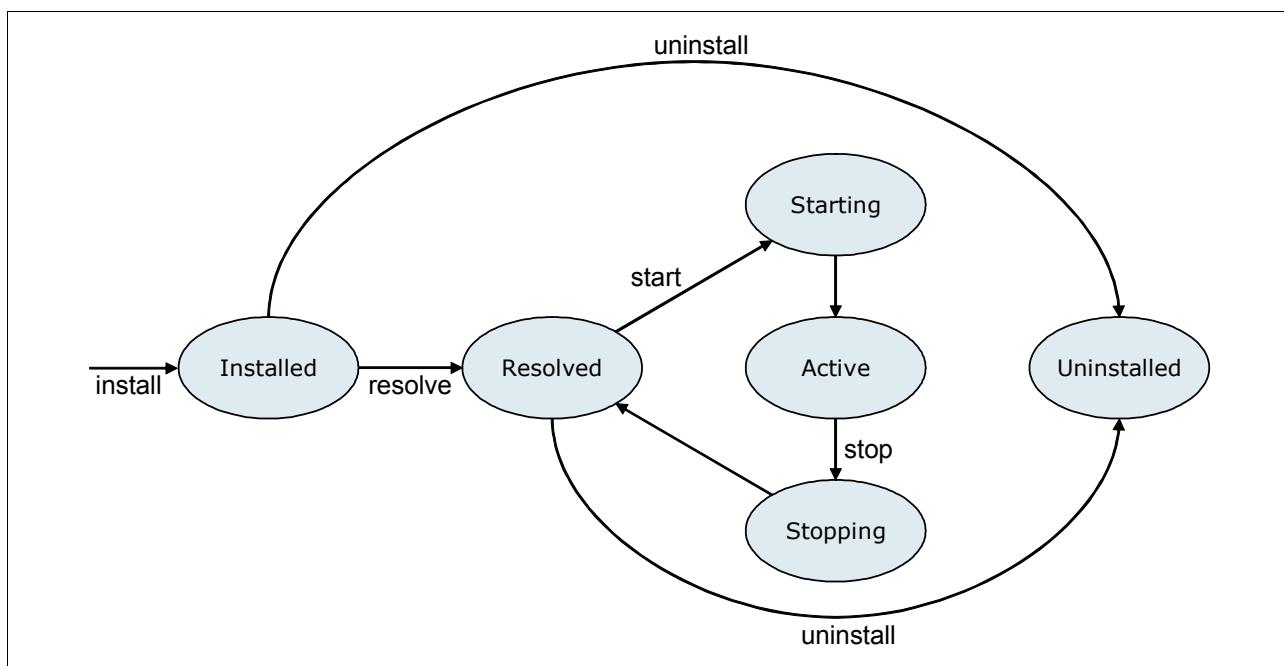


Figure 24-1 Bundle life cycle

When a bundle moves to active state, the OSGi framework optionally invokes a special class called the *bundle activator* that is specified in the bundle manifest. The bundle activator allows an OSGi bundle to be more than just a provider of classes by actively executing tasks and registering and consuming services, and other tasks.

Finally, note that OSGi is built with support for laziness. Bundles are not started unnecessarily but only when explicitly requested or first needed. Even when started explicitly, a bundle author can defer activation to when the first class is loaded from the bundle. This is called *lazy activation*.

Service life cycle

OSGi also introduces a service registry layer. Bundles publish services to the service registry, and other bundles can discover these services from the service registry.

These services are the primary means of collaboration between bundles. An OSGi service is a plain old Java object (POJO) that is published to the service registry under one or more Java interface names with any optional metadata stored as custom properties. A discovering bundle can look up a service in the service registry by an interface name and can potentially filter the services that are looked up based on the custom properties.

Services are fully dynamic and typically have the same life cycle as the bundle that provides them. OSGi applications in WebSphere Application Server usually interact with the OSGi service registry through a Blueprint¹ module definition. POJO bean components that are described in the Blueprint module definition can be registered as services through a `<service>` element or can have service references injected into them through a `<reference>` element in the Blueprint descriptor XML file.

As an immediate consequence of bundles having a life cycle, services must also have a life cycle. The life cycle of a service, by default, is framed by the life cycle of the bundle that provides it. However, bundles can choose to publish and retract services dynamically due to changes in the environment, for example, due to one required service going away or coming back.

Updates

Bundle and service life cycles, along with the event notification support that OSGi defines around them, gives developers the tools to build truly dynamic applications. However, even with this support, it remains far from trivial to write code that can cope appropriately with a truly dynamic environment in which services can come and go at any time.

The OSGi model allows you to perform live updates of the module during which time the environment normally runs the applications. This method forces OSGi to cope with a bundle that might disappear and then re-appear or be present and active more than once for some interval. This method minimizes down time of server side applications but alternatively also requires a good OSGi design.

24.1.3 Enterprise OSGi

OSGi was originally targeted to support Java Platform, Standard Edition, but Version 4.2 of the OSGi specification introduces additional support for Java Platform, Enterprise Edition. The following Java EE technologies are integrated with the OSGi framework:

- ▶ Web Application Specification

Defines how to support the Servlet 2.5 and JSP 2.1 specifications in OSGi. Bundles that are built on this support are called *web application bundles*. At the very least, web application bundles must be marked by the Web-ContextPath bundle manifest header.

¹ For more information on this topic, go to the following website:
http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.osgi.nd.multiplatform.doc/topics/ca_blueprint.html

- ▶ JNDI Services Specification
Defines how OSGi bundles can access javax.naming services and how JNDI can be used to access the OSGi service registry.
- ▶ JPA Service Specification
Defines the basic support for unmanaged JPA in an OSGi bundle, called a *persistence bundle*, which must be marked by the Meta-Persistence manifest header. In particular, the specification defines packaging requirements around persistence bundles as well as provider selection and integration with the JPA run time.
- ▶ Blueprint Container Specification
Based upon the Spring dynamic modules project, Blueprint provides a light-weight, XML-based POJO injection model with special support for the OSGi service registry.

These specifications allow developers to write an OSGi enterprise application using methods that are similar to writing Java EE applications. Because the OSGi model is non-invasive to the classic Java EE model, both applications can be used in the same run time. Application designers can decide on a given application architecture.

24.2 Using the sample application

In this section, we use an ITSOBank sample application to explain the OSGi packaging model. To work with the OSGi application, we use IBM Assembly and Deploy Tools for WebSphere Administration, which has support on an OSGi V4.2 application. However, you can use any other tool that supports the OSGi framework.

To download the sample application, download the sg247835.zip file from the following website:

<ftp://www.redbooks.ibm.com/redbooks/SG247835>

After extracting the file, the OSGi_ITSO_example.zip file in the 7835codesolution/osgi directory contains the ITSOBank OSGi source code.

To import OSGi projects to IBM Assembly and Deploy Tools for WebSphere Administration, extract the OSGi_ITSO_example.zip file to your local directory, and then complete the following steps:

1. Start IBM Assembly and Deploy Tools for WebSphere Administration.
2. Click **File → Import**, and select **Existing projects into workspace** under the **General** section. Click **Next**.
3. Click **Browse** next to the **Select root directory**, and point to the root folder where you extracted the OSGi_ITSO_example.zip file. Then, click **OK**.

4. The process discovers the five projects shown in Figure 24-2. Ensure that you select all the projects. Then, select **Copy projects into workspace**.

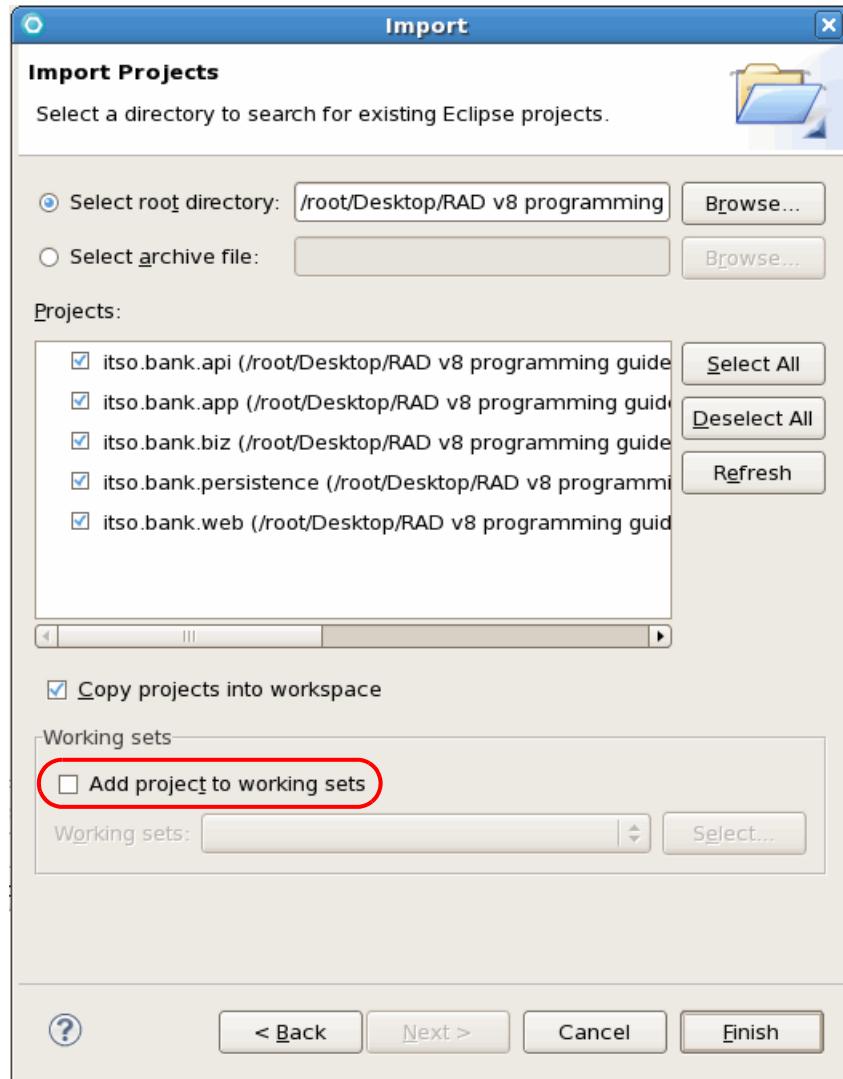


Figure 24-2 Importing an OSGi application

5. Click **Finish**.
6. After the projects are loaded to your workspace, copy the JAR files that the itso.bank.web bundle uses. The JAR files are located in the jar_files directory, which is also unpacked in the root directory. Because this is not an Eclipse project, it is not imported to your workspace.

Copy all the JAR files from the jar_files directory. Then, in the Enterprise Explorer view of the workspace, expand the itso.bank.web application, and paste the files into the WEB-INF/lib directory. Figure 24-3 shows the result. Note that this method is only one way to supply libraries to the web project.

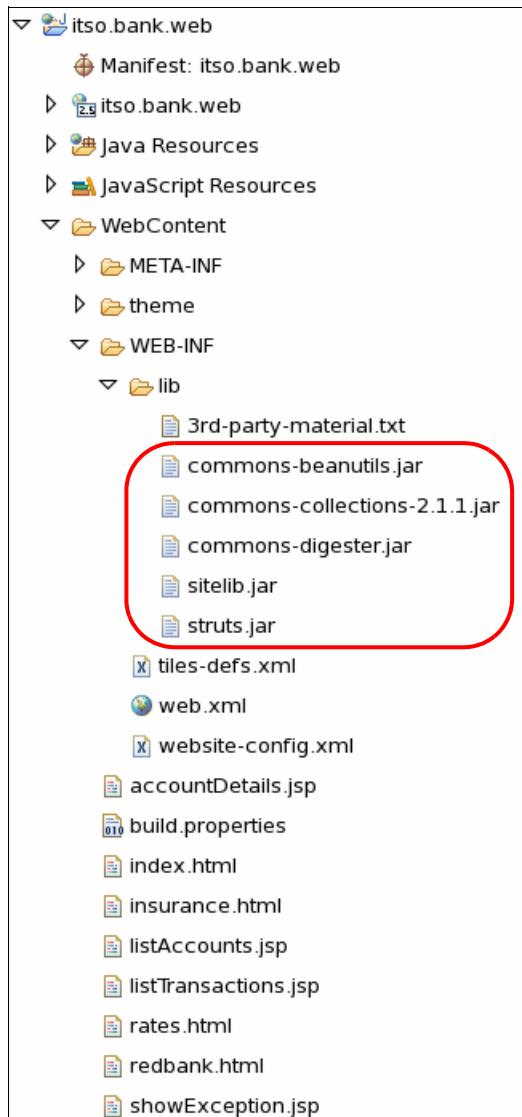


Figure 24-3 JAR files copied into the WEB-INF/lib directory

Other configuration: To have a fully operational OSGi ITSOBank application, additional configuration is required, such as configuring the data source on the server. Refer to 21.7, “Preparing the runtime environment for the application” on page 805 for more information.

After importing the projects, you can use the Enterprise Explorer perspective to preview the ITSOBank application projects, as shown in Figure 24-4.

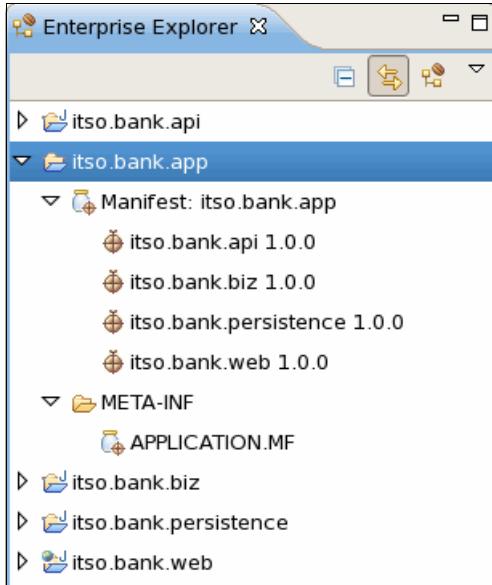


Figure 24-4 OSGi ITSOBank application projects in the workspace

The OSGi application uses the same database that the Java EE sample application uses. For more information, refer to 21.2.4, “Creating the ITSO Bank DB2 database” on page 790.

24.3 Packaging OSGi applications

In this section, we describe the two kinds of OSGi bundle archives.

24.3.1 Enterprise bundle archives

An enterprise bundle archive (EBA) is an archive of more than one bundle that is deployed as a single OSGi application. EBAs are isolated from the bundles and services of other OSGi applications and run under their own instance of the framework. Bundles that belong to an OSGi application can reference other bundles that are in the shared bundle repository (that are not included within the application) as long as these external bundles export packages that these bundles import.

The archive contains the following information:

- ▶ A set of JAR files that represent bundles within the EBA
- ▶ An application manifest
- ▶ An optional deployment manifest, which is a file that WebSphere Application Server generates when resolving an application.

This file defines the exact packages and their versions to which the application is resolved after deployment. IBM Assembly and Deploy Tools for WebSphere Administration allows the user to view this file, and it even persists the file within the application to guarantee that the application is resolved to the exact same bundles in a separate environment.

24.3.2 Composite bundle archives

A composite bundle archive (CBA) is a group of bundles that act as a single bundle from the user perspective. CBAs can be deployed to the WebSphere internal repository, and they can be used potentially by multiple OSGi applications. When the run time resolves a package to a bundle within a CBA, it has the affinity to resolve the remainder of the packages within the same CBA if at all possible. The user creates these archives when the run time is required to pick particular versions of packages that work together in a predictable way. A CBA uses the .cba extension and contains the following information:

- ▶ Composite bundle contents, which is a set of JAR files that represent the contained bundles
- ▶ Composite bundle manifest

24.3.3 Common OSGi patterns

Consider using the following common OSGi patterns that have special support in the packaging of OSGi applications:

- ▶ Web content is packaged in web application bundles.
- ▶ Business logic is written as POJOs.
- ▶ Bundles share interfaces and services rather than concrete implementations.
- ▶ Other bundles depend on services.
- ▶ Persistence is achieved through JPA managed persistence.
- ▶ Declarative transactions are provided by a custom Blueprint extension.

24.3.4 Sample application packaging

The ITSOBank is a simple application, but it presents a good approach about how to design OSGi modules. This application is built from four bundles that follow a three-tier architecture of front end, business logic, and back end, as shown in Figure 24-5.

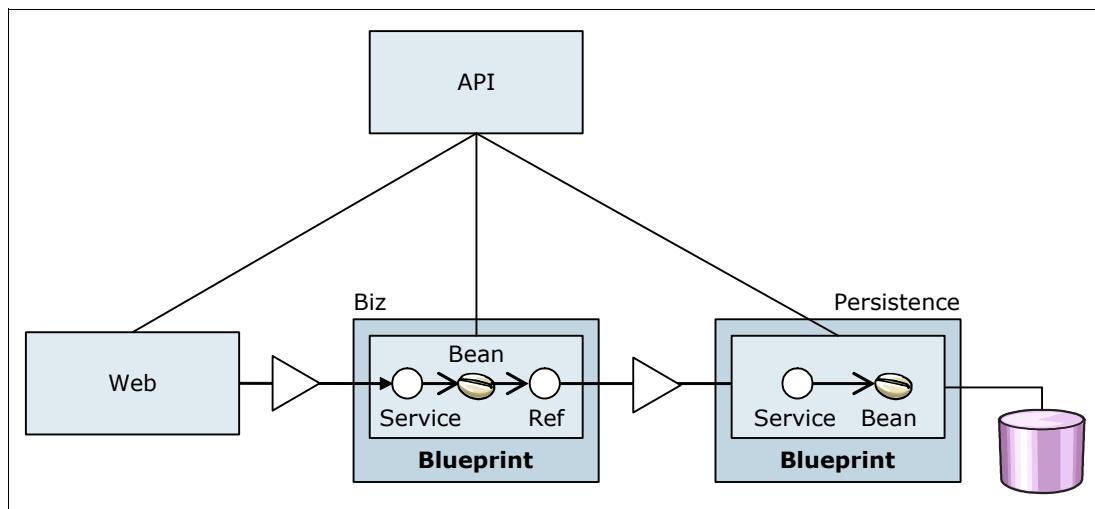


Figure 24-5 OSGi ITSOBank application architecture

A single main application project, `itso.bank.app`, holds a descriptor `META-INF/APPLICATION.MF` file, which configures bundles that are used by the application (Example 24-2).

Example 24-2 Application.mf file

```
Application-Name: itso.bank.app
Application-SymbolicName: itso.bank.app
Application-ManifestVersion: 1.0
Application-Version: 1.0.0.qualifier
Manifest-Version: 1.0
Application-Content: itso.bank.api;version="1.0.0",
                     itso.bank.biz;version="1.0.0",
                     itso.bank.persistence;version="1.0.0",
                     itso.bank.web;version="1.0.0"
```

The other projects define each bundle as follows:

<code>itso.bank.api</code>	An API bundle that contains the interfaces that connect the web logic to the business logic and the business logic to persistence.
<code>itso.bank.biz</code>	A business bundle that contains the business logic and acts as an intermediary between the presentation logic and the database.
<code>itso.bank.persistence</code>	A persistence bundle that encapsulates the JPA-based database access pattern.
<code>itso.bank.web</code>	A web bundle for servlets, JSPs, and static content that delegates the actual business functionality to the business bundle.

The three-tier application split is a standard pattern, but in an OSGi model, it contains a number of OSGi specific twists.

Firstly, all the implementation layers are connected through interfaces only. Depending only on interfaces is a widely acknowledged preferred practice that helps to minimize dependencies of a class and to ensure unit testability. OSGi helps to make sure that concrete classes are not visible. Thus, individual module developers can choose to use the provided interfaces only. To obtain concrete implementation of the service interfaces, we use the OSGi service registry.

However, even using the OSGi APIs for accessing the service registry bloats the business classes unnecessarily with OSGi specific code. Instead, as second step, we use Blueprint to provide the fine-grained dependency injection that separates the wiring of business beans to and from services from their implementation.

Finally, there is the API bundle, which warrants a closer look. In a traditional Java EE style development, we likely would have included the interfaces in the bundles that also provide the implementation. Alternatively, in idiomatic OSGi development, a key criteria for deciding on packaging and module boundaries are responsibilities and frequency of change across the entire module life cycle.

In this case, we note that providing the service and entity interfaces for, say, persistence and implementing the interfaces are two separate concerns. Besides a JPA-based persistence mechanism, we could envisage a mechanism that accesses a no-SQL persistence store or a flat file. Rather than packaging the interfaces in each separate implementation, separating the concerns early on pays off.

Furthermore, the frequency of change also distinguishes interfaces and implementation. When in initial development, interfaces and implementation might change simultaneously, which is not true for the life cycle beyond the first release. Interfaces are less likely to change because the cost for change is high and because client code can be broken easily. Alternatively, the hidden implementation classes will likely change more frequently without requiring interface changes, in particular as result of defects. Thus, separate packaging conceptually means we can update the implementation without changing the interface bundles. With regard to OSGi dynamics, this difference is significant. Replacing interface classes can never be done seamlessly, but replacing a service implementation can.

24.4 Exporting OSGi applications

To export an OSGi application from the IBM Assembly and Deploy Tools for WebSphere Administration, complete the following steps from the Enterprise application view:

1. Select the main OSGi application (itso.bank.app in this case), right-click this application, and then click **Export → Export**.
2. From the context window, click **OSGi Application (EBA)** application type.
3. Enter the location of the exported file in the “To EBA file” field. The .eba extension is added automatically to this file. Click **Finish**. Figure 24-6 shows the export creator.

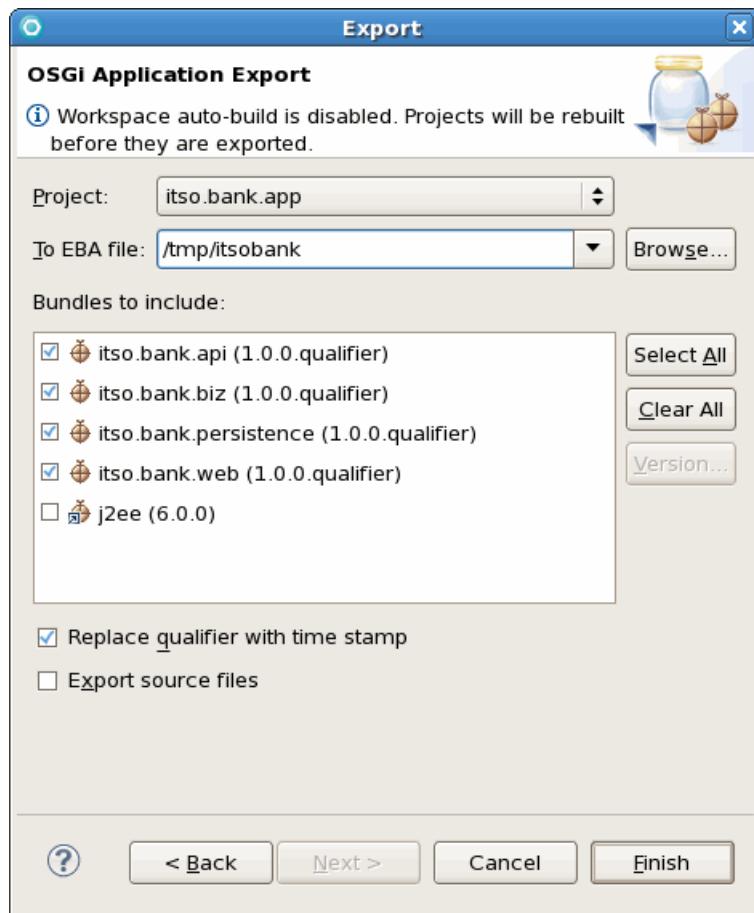


Figure 24-6 Exporting an OSGi application

An OSGI package for deployment is built and validated. In this case, a file called /tmp/itsobank.eba is created. You can inspect this file for its content and generated descriptor files. Notice that if you select the **Replace qualifier with timestamp** option, the generated versions and bundles name qualifier part will be a time stamp, as shown in the following examples

- ▶ Application-Version: 1.0.0.201107151612
- ▶ itso.bank.biz_1.0.0.201107151612.jar

When generating an application without using this option, the files are saved with the following names:

- ▶ Application-Version: 1.0.0.qualifier
- ▶ itso.bank.biz_1.0.0.qualifier.jar

You can also use other tools to build your OSGI applications, such as Ant or Maven, and automate them with `wsadmin` scripts. To learn more about these technologies, refer to the following websites:

- ▶ <http://ant.apache.org>
- ▶ <http://maven.apache.org>

Additional considerations when exporting OSGI applications

Because of the complexities of the OSGI class loader, give special consideration to importing an exported package. When a package is exported, the package can be used by a separate bundle before the exported bundle is even started. This situation can be a problem when singletons and static fields are used, because an imported class is loaded by a separate class loader than the class loader that is used by the bundle, which creates separate instances of the service object.

For example, suppose our bundle uses and exports a service object that is responsible for generating sequential order numbers. Also, suppose that three other bundles use that service object for generating order numbers. Because those three other bundles imported the package, they use the “framework class loader” and share the same instance of the service object. If our bundle also imports the package, it also uses the same instance. However, if our bundle does not import the package, the “bundle class loader” instantiates a new instance of the service object and possibly produces a duplicate list of order numbers.

Leading practices dictate that you typically must import any packages that you export to reduce the number of copies of that package in memory and to ensure that the object instances come from the same class loader.

For more information about OSGI class loaders, see 20.5, “OSGi class loaders” on page 773.

24.5 Deploying OSGI applications

An OSGI application is deployed using the OSGI admin extensions in the administrative console, or using `wsadmin`. Installation using `wsadmin` is the common method used in a production environment. Although less convenient, installing through the administrative console is useful when dealing with complex application structures involving multiple versions and sharing. Using the administrative console for the first installation is also useful for capturing script commands for future use with `wsadmin`.

In this section, we explain how to deploy OSGI applications.

24.5.1 Deploying the application

OSGi applications are packaged in enterprise bundle archive (EBA) files. Deploying applications packaged this way involves completing the following steps:

1. In the administrative console, click **Applications** → **Application Types** → **Assets**. Click **Import**, and select the .eba file that contains the application. In the wizard, accept all the defaults.

Figure 24-7 shows the import wizard. The EBA asset conversion and resolution warnings, which display in the upper-left corner, do not appear at this stage. These messages display only after introducing shared bundles by selecting the **Using shared bundles** option.

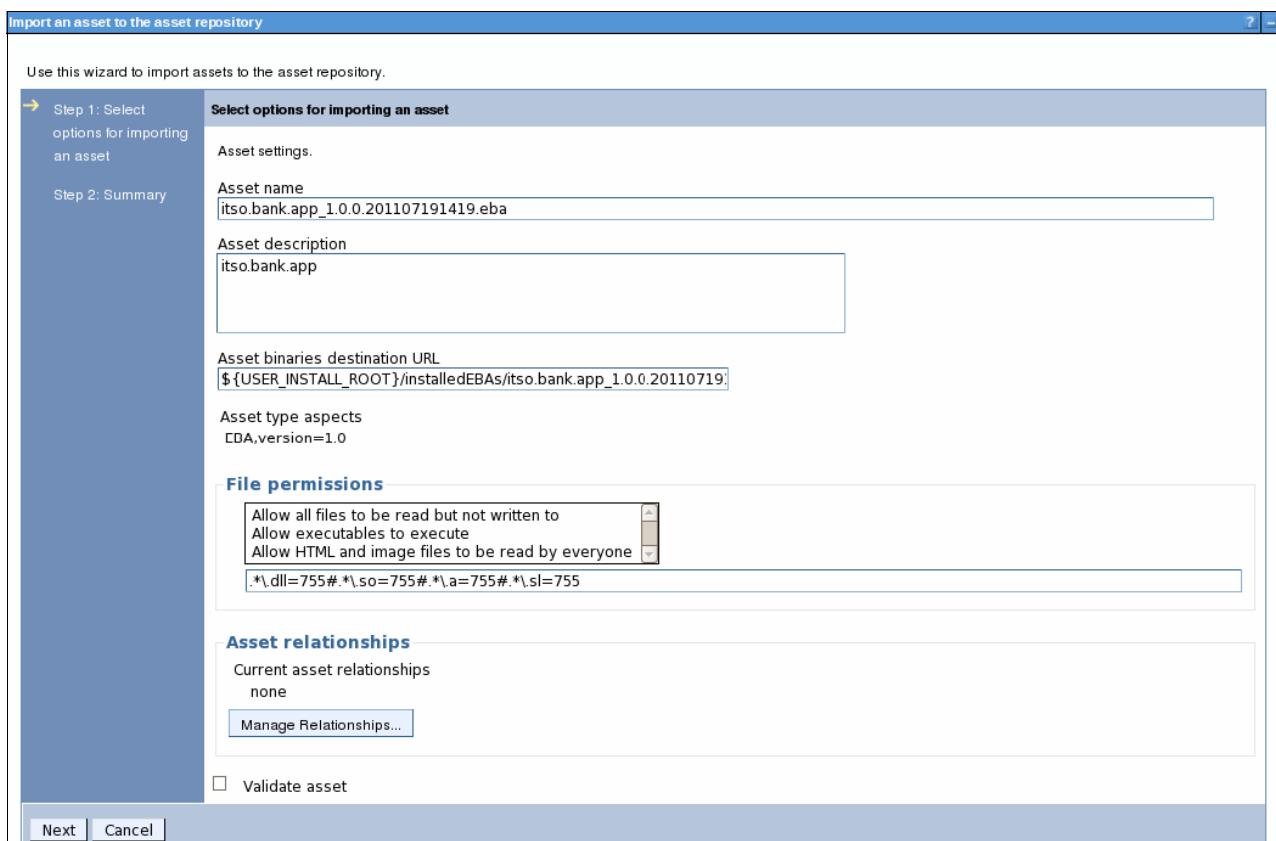


Figure 24-7 Importing assets

2. Click **Next**, then **Finish** to import the asset.
3. Save your changes. For OSGi applications, provisioning happens when importing the asset. As a result of provisioning, bundles might need to be retrieved from a bundle repository (internal or external). This process needs to be complete *before* the asset can be added to a business-level application. However, downloads are triggered only after the asset import is saved. Thus, saving the changes after importing the asset is required.
4. Click **Applications** → **Application Types** → **Business-level applications**.
5. Click **New** to create a new, empty business-level application. Enter a name for the application, **ITS0Bank Systems** in this example, and click **Apply**.

- In the business-level application detail panel, click **Add → Add Asset** under in the Deployed Assets section, as shown in Figure 24-8. Choose the asset that was created in step 2 from the list of assets and click **Continue**.

The screenshot shows the 'General Properties' window for an application named 'ITSOBank System'. The 'Description' field contains the text 'Contains ITSOBank anda DefaultApplication'. Under the 'Deployed assets' section, there is a table with one row labeled 'None'. A dropdown menu next to the table has 'Add Asset' selected.

Select	Name	Description	Type	Status
None				

Figure 24-8 Deployed assets

- A wizard will start, allowing you to enter options for the new composition unit that will be created. In Step 1, set the options for the composition unit:

The fields on this window are:

- Backing ID: Displays the unique identifier for the composition unit that will be registered in the application domain. You cannot change this setting.
- Name: Name for the composition unit.
- Starting weight: Specifies the order in which composition units are started when the server starts. The composition unit with the lowest number is started first.
- Start composition unit upon distribution: Specifies whether to start the composition unit when it is distributed to other locations. This setting applies only to assets and shared library composition units.
- Restart behavior on update (of the composition unit):
 - ALL: Restarts the composition unit after the entire composition unit is updated.
 - DEFAULT: Restarts the composition unit after the part of the composition unit is updated.
 - NONE: Does not restart the composition unit after the composition unit is updated.

Click **Next**.

- In Step 2, select the server on which the `itso.bank.app` application will run and click **Next**.

9. In Step 3, select the context root to which the itso.bank.app is to be mapped (/itsobank) and click **Next**, as shown in Figure 24-9.

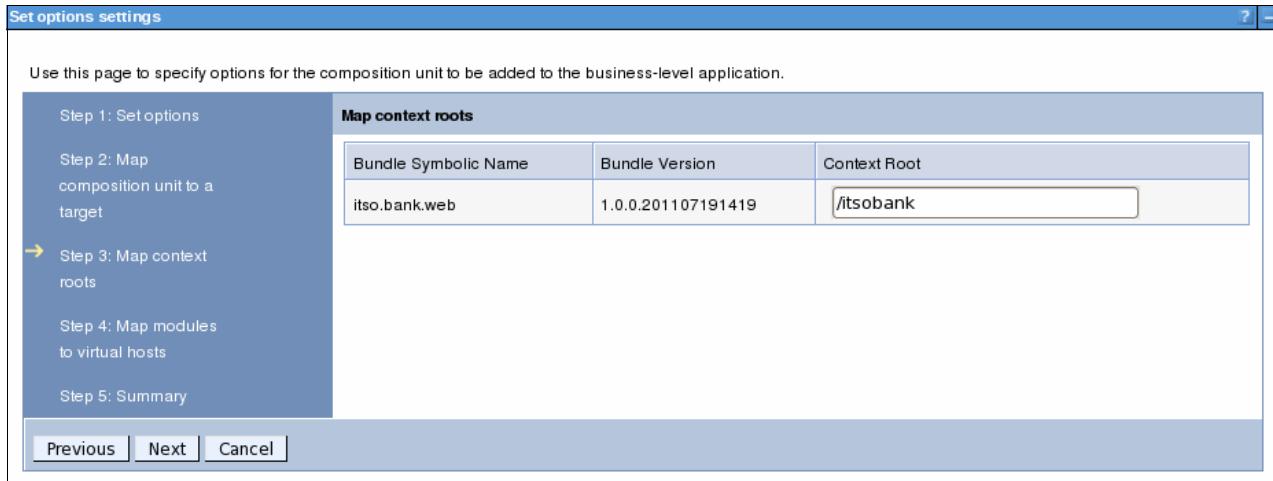


Figure 24-9 Default verification

10. In Step 4, select the desired virtual host and click **Next**.
11. On the Summary page, click **Finish**.
12. Save the changes.
13. Start the business-level application.

24.6 Updating OSGi applications

Before you update an OSGi application, ensure that the new update bundles have a valid and updated version value. If the new bundle version is same as the old bundle version, the OSGi run time treats the original version and the new version interchangeably (despite potentially different build numbers). Because the two bundles have the same version, they are expected to contain exactly the same contents. See 24.1.1, “OSGi application model” on page 872 for more information about the OSGi versioning model.

In this example, we update the ITSO Bank OSGi application, described in 24.3, “Packaging OSGi applications” on page 879.

To update the ITSOBank application, the new bundle will be added to the internal OSGi bundler repository by completing the following commands:

1. In the administrative console, click **Environment** → **OSGi bundle repositories** → **Internal bundle repository**.
2. Click **New**, and in the “Specify path” field, enter the path to the new bundle. You can choose to specify a path on your local computer or on the remote network manager environment.

In this case, we use the itso.bank.biz bundle Version 1.0.1 that was exported from Rational Application Developer. (This bundle is a single JAR file.)

3. Click **OK**, and save the configuration.

After the new bundle has been added to the repository, configure the ITSO Bank application to use it explicitly. Complete the following steps:

1. Click **Applications** → **Application Types** → **Assets**, and choose the ITSO Bank application.
2. Under the Additional Properties section, click the **Update bundle versions in this application** link.
3. From the new form, choose the new version of the `itso.bank.biz` bundle, as shown on Figure 24-10. Click **Preview**.

Symbolic Name	Content Type	Sharing	Deployed Version	New Version
itso.bank.api	Bundle	Isolated	1.0.0.201107141625	No preference
itso.bank.biz	Bundle	Isolated	1.0.1.201107161916	1.0.1.201107161916
itso.bank.persistence	Bundle	Isolated	1.0.0.201107141625	No preference
itso.bank.web	Bundle	Isolated	1.0.0.201107141625	No preference

Figure 24-10 Updating OSGi bundles

Selecting the version: The drop-down menus list all the available versions plus the “No preference” option. Selecting a specific version instructs the provisioning system to use exactly that version for the update. Selecting **No preference** instructs the provisioning system to find the highest version that works with the other selections.

With both selections, the new version must fall into the range that is specified in the application manifest. For example, the following application content specifies that only bundle versions from 1.0.0 but less than 2.0.0 are admissible during provisioning and during updates:

Application-Content: `itso.bank.biz;version="[1.0.0,2.0.0)"`

In our scenario, we want to pull the 1.0.1 version of the `itso.bank.biz` bundle. Note that “No preference” would achieve the same effect, assuming that there are no later versions of the bundle available.

4. If no errors occur during processing, a message displays that selected bundle versions can be resolved, as illustrated on Figure 24-11. Click **Create**.

The screenshot shows the 'Assets' screen in the WebSphere Application Server interface. The URL in the address bar is [Assets > itsobank.eba > Update bundle versions in this application > Preview](#). The page title is 'Assets'. Below the title, it says 'A preview of the result of the proposed changes to the bundle versions in this application.' A note states: 'The selected bundle versions can be resolved, so you can now create a new deployment with the proposed bundle versions. The new deployment will not affect any composition units for this asset until the composition units are updated to use the new deployment.' A table titled 'Application bundle content' lists four bundles: 'itso.bank.api', 'itso.bank.biz', 'itso.bank.persistence', and 'itso.bank.web'. Each row has three columns: Symbolic Name, Deployed Version, and New Version. All 'Deployed Version' and 'New Version' columns show the same values: '1.0.0.201107141625', '1.0.1.201107161916', '1.0.0.201107141625', and '1.0.0.201107141625' respectively. At the bottom are 'Create' and 'Cancel' buttons.

Figure 24-11 Updating the OSGi bundle on the WebSphere Application Server run time

At this time, the configured bundles are downloaded from repositories, in this case from an internal OSGi repository on WebSphere Application Server. The OSGi run time must cache the bundles locally so that applications are not affected by changes to the bundle repositories. This process of downloading the new versions to create local copies can take a small amount of time with external repositories or very large bundles, but is mostly instantaneous when working with the internal bundle repository.

You can check the status of downloads from the asset detail window or the composition unit detail window, as illustrated on Figure 24-12.



Figure 24-12 Download status

5. After all bundle downloads complete, apply the new configuration to the run time by clicking **Update to latest deployment** under **Business-level applications** → **ITSOBank application** → **ITSOBank eba asset**.

For more information about updating OSGi applications (for example, using `wsadmin` scripting), go to the following websites:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.osgi.nd.multiplatform.doc/topics/thread_ta_admin_maint.html

You can also read more about customizing and updating the ITSO Bank OSGi application in *Getting Started with the Feature Pack for OSGi Applications and JPA 2.0*, SG24-7911.



Session management

Session support allows a web application developer to maintain state information across multiple user visits to the application. In this chapter, we discuss HTTP session support in Websphere Application Server V8 and how to configure it. We also discuss the support for stateful session bean failover.

This chapter includes the following topics:

- ▶ HTTP session management
- ▶ Session management configuration
- ▶ Session identifiers
- ▶ Local sessions
- ▶ General properties for session management
- ▶ Session affinity
- ▶ Persistent session management
- ▶ Invalidating sessions
- ▶ Session performance considerations
- ▶ Stateful session bean failover

25.1 HTTP session management

In many web applications, users collect data dynamically as they move through the site, based on a series of selections on pages that they visit. Where the user goes next and what the application displays as the user's next page or next choice, depends on what the user chose previously from the site. For example, if the user clicks a checkout button on a site, the next page must contain the user's shopping selections.

For this type of management to happen, a web application needs a mechanism to hold the user's state information over a period of time. However, HTTP does not recognize or maintain a user's state. HTTP treats each user request as a discrete, independent interaction.

The Java servlet specification provides a mechanism for servlet applications to maintain a user's state information. This mechanism, known as a *session*, addresses some of the problems of more traditional strategies, such as a pure cookie solution. A session allows a web application developer to maintain all user state information at the host, while passing minimal information back to the user through cookies or using another technique known as *URL rewriting*.

25.2 Session management configuration

Session management in WebSphere Application Server can be defined at the following levels:

- ▶ Application server (the default level)
Configuration at this level is applied to all web modules within the server.
- ▶ Application
Configuration at this level is applied to all web modules within the application.
- ▶ Web module
Configuration at this level is applied only to that web module.

25.2.1 Session management properties

With the exception of the *Overwrite session management* parameter, the session management properties are the same at each configuration level:

- ▶ The *Overwrite session management* parameter, for enterprise application and web module level only, determines whether these session management settings are used for the current module or are used from the parent object.
- ▶ *Session tracking mechanism* lets you select from cookies, URL rewriting, and SSL ID tracking. Selecting cookies leads you to a second configuration page that contains further configuration options.
- ▶ *Maximum in-memory session count* specifies the maximum number of sessions to keep in memory and whether to allow this number to be exceeded or to overflow.
- ▶ *Session timeout* specifies the amount of time to allow a session to remain idle before invalidation.
- ▶ *Security integration* specifies that the user ID be associated with the HTTP session.
- ▶ *Serialize session access* determines if concurrent session access in a given server is allowed.

- *Distributed environment settings* determines how to persist sessions (memory-to-memory replication or a database) and set tuning properties. Memory-to-memory persistence is available only in a Network Deployment distributed server environment.

25.2.2 Accessing session management properties

You can access all session management configuration settings using the administrative console.

Application server session management properties

To access session management properties at the application server level, from the administrative console, complete the following steps:

1. Click **Servers** → **Server Types** → **WebSphere application servers**.
2. Click the application server.
3. In the Container Settings section of the Configuration tab, click **Session management**.

Application session management properties

To access session management properties at the application level, from the administrative console, complete the following steps:

1. Click **Applications** → **Application Types** → **WebSphere enterprise applications**.
2. Click the application name to open its configuration page.
3. In the Web Module Properties section of the Configuration tab, click **Session management**.

Web module session management properties

To access session management properties at the web module level, from administrative console, complete the following steps:

1. Click **Applications** → **Application Types** → **WebSphere enterprise applications**.
2. Click the application.
3. In the Modules section of the Configuration tab, click **Manage Modules**.
4. Click the web module.
5. In the Additional Properties section, click **Session Management**.

25.3 Session identifiers

WebSphere session support keeps information about the user's session on the server. WebSphere passes the user an identifier known as a *session ID*, which correlates an incoming user request to a session object that is maintained on the server.

Example session IDs: The example session IDs that we provide in this chapter are for illustrative purposes only and are *not* guaranteed to be absolutely consistent in value, format, and length.

25.3.1 Choosing a session tracking mechanism

WebSphere supports the following approaches to tracking sessions:

- ▶ SSL session identifiers (deprecated)
- ▶ Cookies
- ▶ URL rewriting

You can select all three options for a web application. If you do this, keep the following information in mind:

- ▶ SSL session identifiers are used in preference to cookie and URL rewriting.
- ▶ Cookies are used in preference to URL rewriting.

Tip: If SSL session ID tracking is selected, also select cookies or URL rewriting so that session affinity can be maintained. The cookie or rewritten URL contains session affinity information, enabling the web server to properly route a session back to the same server for each request.

To set or change the session mechanism type, complete the following steps:

1. Open the session management properties for the application server, enterprise application, or web module, as described in 25.2.2, “Accessing session management properties” on page 891.

2. In the General Properties section, select the session tracking mechanism, as shown in Figure 25-1, and then click **OK**.

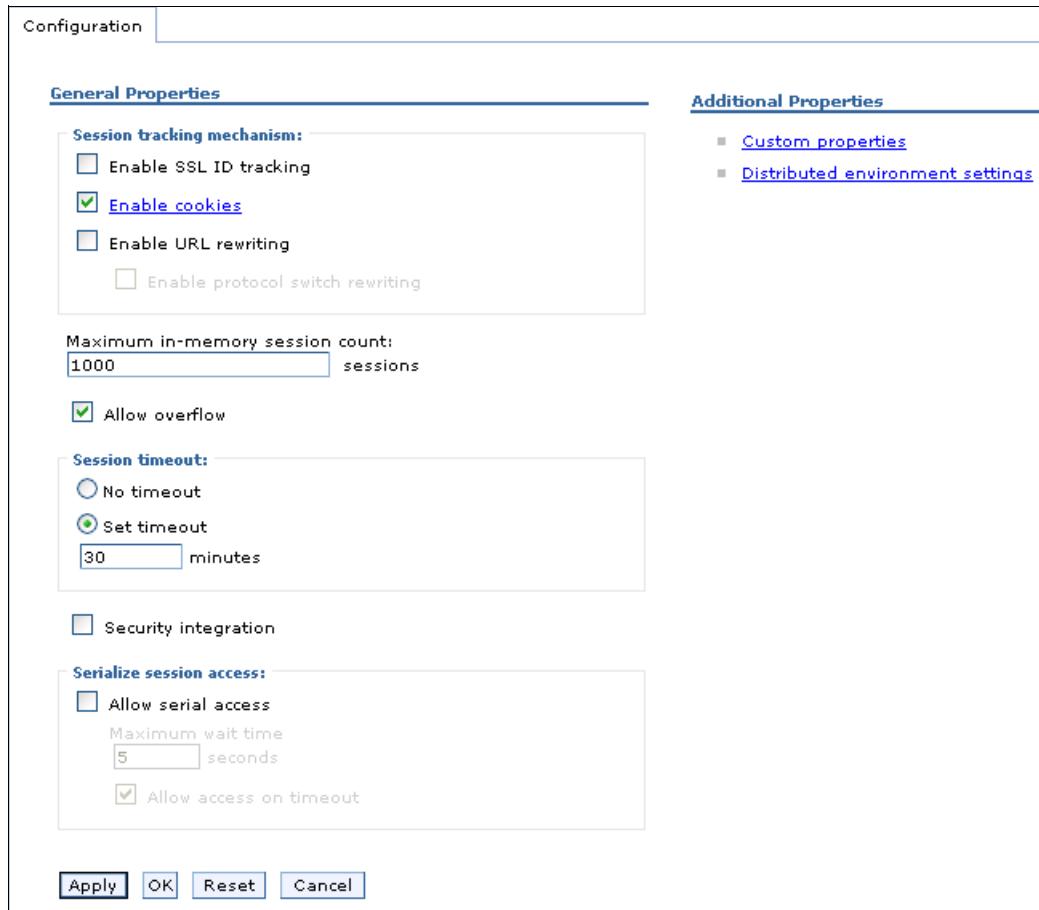


Figure 25-1 Selecting a session tracking mechanism window

3. Save and synchronize the configuration changes.
4. Restart the application server or the cluster.

25.3.2 Cookies

Many sites choose cookie support to pass the user's identifier between WebSphere and the user. WebSphere Application Server session support generates a unique session ID for each user and returns this ID to the user's browser with a cookie, as illustrated in Figure 25-2. The default name for the session management cookie is *JSESSIONID*.

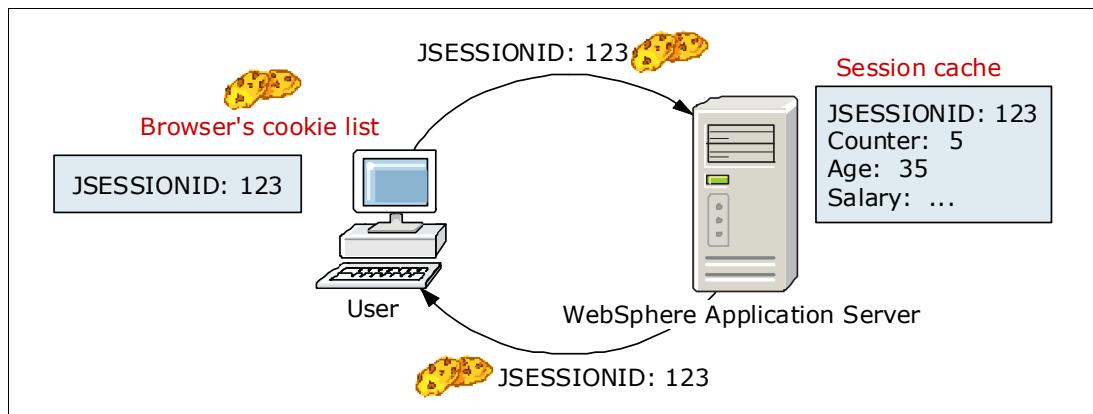


Figure 25-2 Cookie overview

Using cookies

A *cookie* consists of information that is embedded as part of the headers in the HTML stream passed between the server and the browser. The browser holds the cookie and returns it to the server whenever the user makes a subsequent request. By default, WebSphere defines its cookies so they are destroyed if the browser is closed. This cookie holds a session identifier. The remainder of the user's session information resides at the server.

The web application developer uses the HTTP request object's standard interface to obtain the session:

```
HttpSession session = request.getSession(true);
```

WebSphere places the user's session identifier in the outbound cookie when the servlet completes its execution, and the HTML response stream returns to the user. Again, neither the cookie or the session ID within it require any direct manipulation by the web application. The web application sees only the contents of the session.

Be aware that some users, either by choice or mandate, disable cookies from within their browser.

Cookie settings

To configure session management using cookies, complete the following steps from the administrative console:

1. Navigate to the Session Manager window at your preferred level, as described in 25.2.2, "Accessing session management properties" on page 891.
2. Select the **Enable Cookies** option as the session tracking mechanism.
3. If you want to view or change the cookies settings, click the **Enable Cookies** hot link. The following cookie settings are available:
 - **Cookie name**
Use a unique cookie name for session management. The default cookie name is *JSESSIONID*. However, you can configure this value.

- Restrict cookies to HTTPS sessions

Restricts the exchange of cookies to only HTTPS sessions. If this option is enabled, the session cookie's body includes the secure indicator field.

- Set session cookies to HTTPOnly to help prevent cross-site scripting attacks

Specifies that session cookies include the HTTPOnly field. When selected, browsers that support the HTTPOnly attribute do not enable cookies to be accessed by client-side scripts. For security cookies, see the global security settings for web single sign-on (SSO).

- Cookie domain

Dictates to the browser whether to send a cookie to particular servers. For example, if you specify a particular domain, the browser sends back session cookies only to hosts in that domain. The default value in the session manager restricts cookies to the host that sent them.

DNS domain note: The Lightweight Third Party Authentication (LTPA) token or cookie that is sent back to the browser is scoped by a single DNS domain that is specified when security is configured. Thus, *all* application servers in an *entire* WebSphere Application Server domain must share the same DNS domain for security purposes.

- Cookie maximum age

Specifies the amount of time that the cookie will live in the client browser. This option includes the following choices:

- Expire at the end of the current browser session
- Expire at a configurable maximum age

If you choose the maximum age option, specify the age in seconds.

- Cookie path

Sets the paths on the server to which the browser sends the session tracking cookie. Specify any string that represents a path on the server. Use the forward slash (/) to indicate the root directory.

Specifying a value restricts the paths to which the cookie is sent. By restricting paths, you can keep the cookie from being sent to certain URLs on the server. If you specify the root directory, the cookie is sent no matter which path on the given server is accessed.

4. Click **OK** to exit the page and change your settings.
5. Click **OK** to exit the session management settings.
6. Save and synchronize your configuration changes.
7. Restart the application server or the cluster.

25.3.3 URL rewriting

WebSphere also supports URL rewriting for session ID tracking. Although session management using SSL IDs or cookies is transparent to the web application, URL rewriting requires the developer to use special encoding APIs and to set up the site page flow to avoid losing the encoded information.

URL rewriting works by storing the session identifier in the page that is returned to the user. WebSphere encodes the session identifier as a parameter on URLs that are encoded programmatically by the web application developer. The following example shows a web page link with URL encoding:

```
<a href="/store/catalog;jsessionid=DA32242SSGE2">
```

When the user clicks this link to move to the /store/catalog page, the session identifier is passed in the request as a parameter.

URL rewriting requires explicit action by the web application developer. If the servlet returns HTML directly to the requester, without using JavaServer Pages, the servlet calls the API, as shown in Example 25-1, to encode the returning content.

Example 25-1 URL encoding from a servlet

```
out.println("<a href=\"");
out.println(response.encodeURL ("/store/catalog"));
out.println("\>catalog</a>");
```

Even pages using redirection, a common practice, particularly with servlet or JavaServer Pages (JSP) combinations, must encode the session ID as part of the redirect, as shown in Example 25-2.

Example 25-2 URL encoding with redirection

```
response.sendRedirect(response.encodeRedirectURL("http://myhost/store/catalog"));
```

When JSP pages use URL rewriting, the JSP calls a similar interface to encode the session ID:

```
<% response.encodeURL ("/store/catalog"); %>
```

URL rewriting configuration

When you select URL rewriting, the additional “Enable protocol switch rewriting” configuration option is available. This option defines whether the session ID, added to a URL as part of URL encoding, should be included in the new URL if a switch from HTTP to HTTPS or from HTTPS to HTTP is required. For example, if a servlet is accessed over HTTP and that servlet is doing encoding of HTTPS URLs, URL encoding is performed only when protocol switch rewriting is enabled and vice versa.

Considerations for using URL rewriting

The fact that the servlet or JSP developer has to write extra code is a major drawback over other available session tracking mechanisms. URL rewriting limits the flow of site pages exclusively to dynamically generated pages, such as pages that are generated by servlets or JSP pages. WebSphere inserts the session ID into dynamic pages but cannot insert the user's session ID into static pages, .htm, or .html.

Therefore, after the application creates the user's session data, the user must visit dynamically generated pages exclusively until the user finishes with the portion of the site that requires sessions. URL rewriting forces the site designer to plan the user's flow in the site to avoid losing the session ID.

25.4 Local sessions

Many web applications use the simplest form of session management, which is the in-memory, local session cache. The local session cache keeps session information in memory and local to the machine and WebSphere Application Server where the session information was first created.

Local session management does not share user session information with other clustered machines. Users only obtain their session information if they return to the application server. Most importantly, local session management lacks a persistent store for the sessions it manages. A server failure takes down WebSphere instances that are running on the server and also destroys any sessions that are managed by those instances.

WebSphere allows the administrator to define a limit on the number of sessions that are held in the in-memory cache from the administrative console settings on the session manager. This limit prevents the sessions from acquiring too much memory in the Java virtual machine (JVM) that is associated with the application server.

The session manager also allows the administrator to permit an unlimited number of sessions in memory. If the administrator enables the **Allow overflow** setting on the session manager, the session manager permits two in-memory caches for session objects. The first cache contains only enough entries to accommodate the session limit that is defined to the session manager, which is 1000 by default. The second cache, known as the *overflow cache*, holds any sessions that the first cache cannot accommodate and is limited in size only by available memory. The session manager builds the first cache for optimized retrieval, and a regular, unoptimized hash table contains the overflow cache.

For best performance, define a primary cache of sufficient size to hold the normal working set of sessions for a given web application server.

Important: If you enable overflow, the session manager permits an unlimited number of sessions in memory. Without limits, the session caches might consume all available memory in the WebSphere instance's heap, leaving no room to execute web applications. For example, here are two scenarios under which this situation can occur:

- ▶ The site receives greater traffic than anticipated, generating a large number of sessions held in memory.
- ▶ A malicious attack occurs against the site where a user deliberately manipulates the browser so that the application creates a new session repeatedly for the same user.

If you choose to enable session overflow, monitor the state of the session cache closely.

In-memory session cache: Each web application has its own base or *primary* in-memory session cache. With overflow allowed, each application also has its own overflow or *secondary* in-memory session cache.

25.5 General properties for session management

The following session management settings allow the administrator to tune a number of parameters that are important for both local or persistent sessions (see Figure 25-1 on page 893):

- ▶ Maximum in-memory session count

This field specifies the maximum number of sessions to maintain in memory. The meaning differs depending on whether you are using local or persistent sessions. For local sessions, this value specifies the number of sessions in the base session table. Select the **Allow overflow** option to specify whether to limit sessions to this number for the entire session manager or to allow additional sessions to be stored in secondary tables. Before setting this value, review the information in 25.4, “Local sessions” on page 897.

For persistent sessions, this value specifies the size of the general cache. This value determines how many sessions will be cached before the session manager reverts to reading a session from the database automatically. The session manager uses a least recently used (LRU) algorithm to maintain the sessions in the cache.

This value holds when you use local sessions, persistent sessions with caching, or persistent sessions with manual updates. The manual update cache keeps the last *n* time stamps representing the last access times, where *n* is the maximum in-memory session count value.

- ▶ Allow overflow

Choosing this option specifies whether to allow the number of sessions in memory to exceed the value specified in the maximum in-memory session count field. If the “Allow overflow” option is not selected, then WebSphere limits the number of sessions held in memory to this value.

For local sessions, if this maximum is exceeded and the “Allow overflow” option is not selected, then sessions created thereafter are dummy sessions and are not stored in the session manager. Before setting this value, review the information in 25.4, “Local sessions” on page 897.

As shown in Example 25-3, you can use the IBM HttpSession extension to react if sessions exceed the maximum number of sessions that are specified when overflow is disabled.

Example 25-3 Using IBMSession to react to session overflow

```
com.ibm.websphere.servlet.session.IBMSession sess =  
    (com.ibm.websphere.servlet.session.IBMSession) req.getSession(true);  
if(sess.isOverFlow()) {  
    //Direct to a error page...  
}
```

Important: Allowing an unlimited amount of sessions can potentially exhaust system memory and even allow for system sabotage. Someone could write a malicious program that continually hits your site, creating sessions but ignoring any cookies or encoded URLs and never using the same session from one HTTP request to the next.

► Session timeout

If you select the **Set timeout** option, when a session is not accessed for this many minutes it can be removed from the in-memory cache and, if persistent sessions are used, from the persistent store. This setting is important for performance tuning. It directly influences the amount of memory that is consumed by the JVM to cache the session information.

Invalidation process interval: For performance reasons, the session manager invalidation process runs at regular intervals to invalidate any invalid sessions. This interval is determined internally based on the Session timeout interval that is specified in the Session manager properties. For the default timeout value of 30 minutes, the invalidation process interval is around 300 seconds. In this case, it can take up to 5 minutes (300 seconds) beyond the timeout threshold of 30 minutes for a particular session to become invalidated.

The value of this setting is used as a default when the session timeout is not specified in a web module's deployment descriptor.

If you select the **No timeout** option, a session can never be removed from the memory unless explicit invalidation is performed by the servlet. This persistent session can cause a memory leak when the user closes the window without logging out from the system. This option might be useful when sessions should be kept for a while until explicit invalidation is done, for example, when an employee leaves the company. To use this option, make sure that enough memory or space in a persistent store is kept to accommodate all sessions.

► Security integration

When security integration is enabled, the session manager associates the identity of users with their HTTP sessions. Refer to 25.11, “Session security” on page 927 for more information.

Form-based login: Do not enable this property if the application server contains a web application that has form-based login configured as the authentication method and the local operating system is the authentication mechanism. Doing so will cause authorization failures when users try to use the web application.

► Serialize session access

This option is available to provide serialized access to the session in a given JVM. Serialized access ensures thread-safe access when the session is accessed by multiple threads. No special code is necessary for using this option. This option is not recommended when framesets are used heavily because it can affect performance.

You can set an optional property, the “Maximum wait time” property, to specify the maximum amount of time that a servlet request waits on an HTTP session before continuing execution. The default value for this setting is 5 seconds.

If you set the “Allow access on timeout” option, multiple servlet requests that have timed out concurrently will execute normally. If it is false, servlet execution aborts.

25.6 Session affinity

In a clustered environment, any HTTP requests that are associated with an HTTP session must be routed to the same web application in the same JVM. This routing ensures that all of the HTTP requests are processed with a consistent view of the user's HTTP session. The exception to this rule is when the cluster member fails or has to be shut down.

WebSphere ensures that session affinity is maintained. Each server ID is appended to the session ID. When an HTTP session is created, its ID is passed back to the browser as part of a cookie or URL encoding. When the browser makes further requests, the cookie or URL encoding is sent back to the web server. The web server plug-in examines the HTTP session ID in the cookie or URL encoding, extracts the unique ID of the cluster member handling the session, and forwards the request.

Figure 25-3 illustrates this situation, where the session ID from the HTTP header, `request.getHeader("Cookie")`, displays along with the session ID from `session.getId()`. The application server ID is appended to the session ID from the HTTP header. The first four characters of HTTP header session ID are the cache identifier that determines the validity of cache entries.

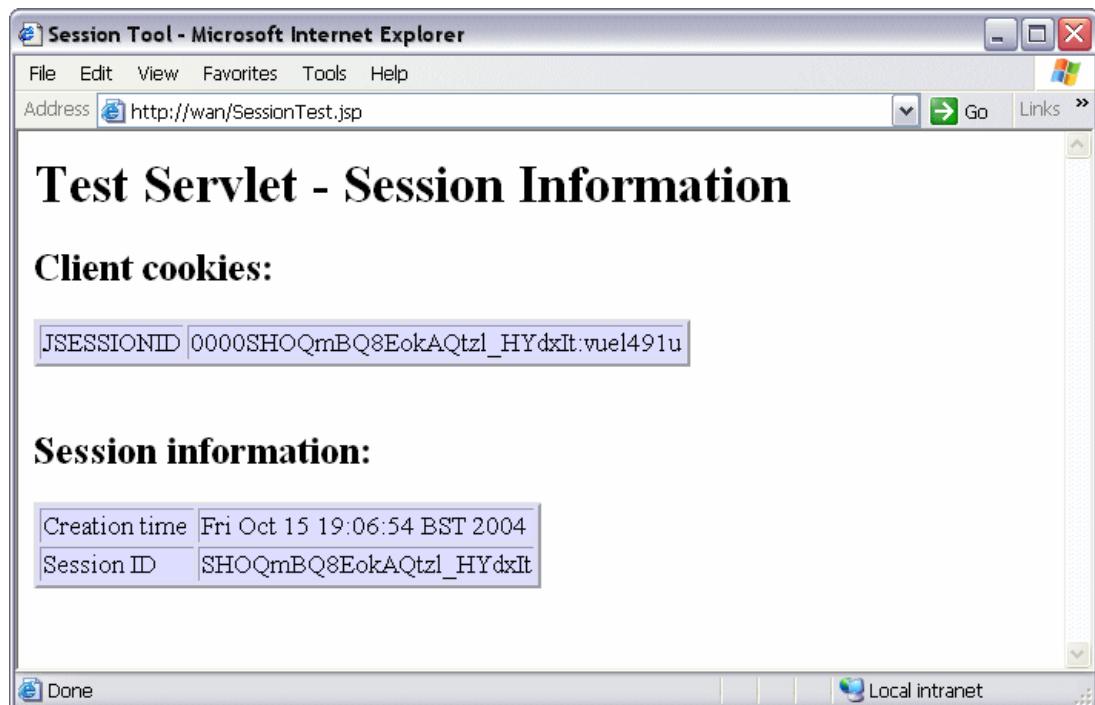


Figure 25-3 Session ID containing the server ID and cache ID

The JSESSIONID cookie can be divided into the following parts:

- ▶ Cache ID
- ▶ Session ID
- ▶ Separator
- ▶ Clone ID
- ▶ Partition ID

The JSESSION ID includes a partition ID instead of a clone ID when memory-to-memory replication in peer-to-peer mode is selected. Typically, the partition ID is a long numeric number.

Table 25-1 shows the cookie mappings based on the example shown in Figure 25-3 on page 900. A clone ID is an ID of a cluster member.

Table 25-1 Cookie mapping

Content	Value in the example
Cache ID	0000
Session ID	SHOQmBQ8EokAQtzl_HYdxlt
separator	:
Clone ID	vuel491u

The application server ID can be seen in the web server plug-in configuration file, the `plug-in-cfg.xml` file, as shown in Example 25-4.

Example 25-4 Server ID from the `plugin-cfg.xml` file

```
<?xml version="1.0" encoding="ISO-8859-1"?><!--HTTP server plugin config file for  
the cell1 ITS0Cell generated on 2004.10.15 at 07:21:03 PM BST-->  
<Config>  
.....  
    <ServerCluster Name="MyCluster">  
        <Server CloneID="vuel491u" LoadBalanceWeight="2" Name="NodeA_server1">  
            <Transport Hostname="wan" Port="9080" Protocol="http"/>  
            <Transport Hostname="wan" Port="9443" Protocol="https"/>  
.....  
</Config>
```

Use persistent session management: Session affinity can still be broken if the cluster member handling the request fails. To avoid losing session data, use persistent session management. In persistent sessions mode, the cache ID and server ID will change in the cookie when there is a failover or when the session is read from the persistent store. Do not rely on the value of the session cookie remaining the same for a given session.

25.7 Session affinity and failover

Server clusters provide a solution for failure of an application server. Sessions created by cluster members in the server cluster share a common persistent session store. Therefore, any cluster member in the server cluster can see any user's session that is saved to persistent storage.

If one of the cluster members fails, the user can continue to use session information from another cluster member in the server cluster. This process is known as *failover*. Failover works regardless of whether the nodes reside on the same machine or several machines. Only a single cluster member can control and access a given session at a time. See Figure 25-4.

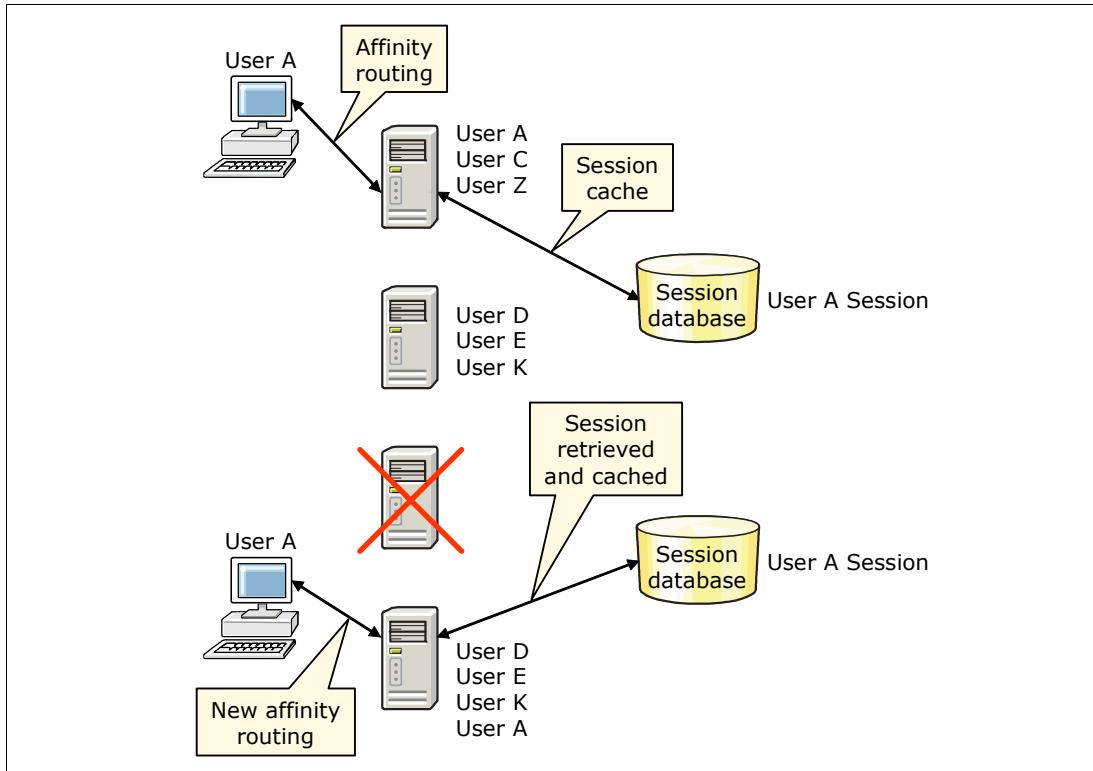


Figure 25-4 Session affinity and failover

After a failure, WebSphere redirects the user to another cluster member, and the user's session affinity switches to this replacement cluster member. After the initial read from the persistent store, the replacement cluster member places the user's session object in the in-memory cache, assuming that the cache has space available for additional entries.

The web server plug-in maintains a cluster member list and picks the cluster member next in the list to avoid the breaking of session affinity. From then on, requests for that session go to the selected cluster member. The requests for the session go back to the failed cluster member when the failed cluster member restarts.

WebSphere provides session affinity on a best-effort basis. There are narrow windows where session affinity fails. These windows are as follows:

- ▶ When a cluster member is recovering from a crash, a window exists where concurrent requests for the same session can end up in different cluster members. The web server is multi-processed, and each process separately maintains its own retry timer value and list of available cluster members. Thus, requests that are processed by different processes might end up being sent to more than one cluster member after at least one process has determined that the failed cluster member is running again.

To avoid or limit exposure in this scenario, if your cluster members are expected to rarely crash and are expected to recover quickly, consider setting the retry timeout to a small value. This smaller value narrows the window during which multiple requests that are handled by different processes are routed to multiple cluster members.

- ▶ A server overload can cause requests that belong to the same session to go to different cluster members. This situation can occur even if all the cluster members are running. For each cluster member, there is a backlog queue where an entry is made for each request that is sent by the web server plug-in that is waiting to be picked up by a worker thread in the servlet engine. If the depth of this queue is exceeded, the web server plug-in receives responses that the cluster member is not available. This failure is handled in the same way by the web server plug-in as an actual JVM crash.

The following examples illustrate how this situation can occur:

- The servlet engine does not have an appropriate number of threads to handle the user load.
- The servlet engine threads take a long time to process the requests because applications are taking a long time to execute, resources that are used by applications are taking a long time, and so on.

25.8 Persistent session management

By default, WebSphere places session objects in memory. However, the administrator has the option of enabling persistent session management, which instructs WebSphere to place session objects in a persistent store. Administrators should enable persistent session management in the following situations:

- ▶ The user's session data must be recovered by another cluster member after a cluster member in a cluster fails or is shut down.
- ▶ The user's session data is too valuable to lose through unexpected failure at the WebSphere node.
- ▶ The administrator desires better control of the session cache memory footprint. By sending cache overflow to a persistent session store, the administrator controls the number of sessions that are allowed in memory at any given time.

Configure session persistence using one of the following methods, as illustrated in Figure 25-5:

- ▶ Database persistence, supported for the web container only
- ▶ Memory-to-memory session state replication using the data replication service available in distributed server environments

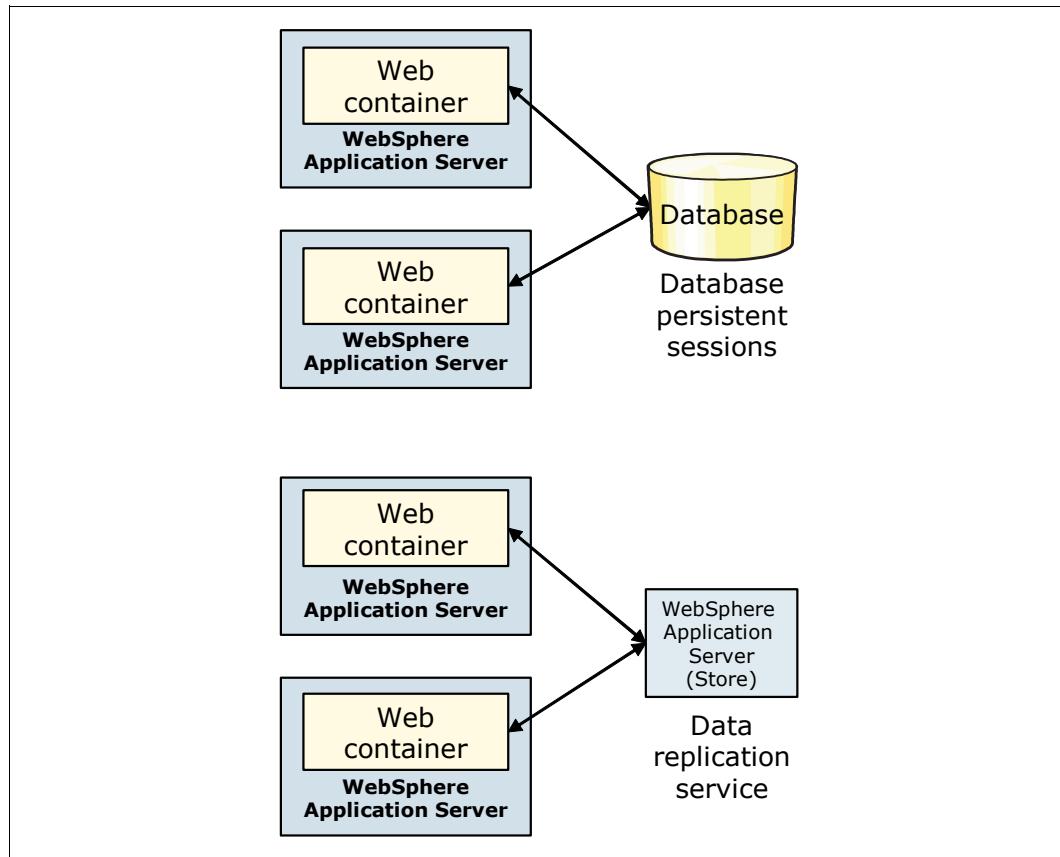


Figure 25-5 Persistent session options

All information that is stored in a persistent session store must be serialized. As a result, all of the objects that are held by a session must implement `java.io.Serializable` if the session needs to be stored in a persistent session store.

In general, consider making all objects that are held by a session serialized, even if immediate plans do not call for the use of persistent session management. If the website grows and if persistent session management becomes necessary, the transition between local and persistent management occurs transparently to the application if the sessions hold only serialized objects. If the sessions do not hold serialized object, a switch to persistent session management requires coding changes to make the session contents serialized.

Persistent session management does not impact the session API, and web applications require no API changes to support persistent session management. However, as mentioned previously, applications storing unserializable objects in their sessions require modification before switching to persistent session management.

If you use database persistence, using multi-row sessions becomes important if the size of the session object exceeds the size for a row, as permitted by the WebSphere session manager. If the administrator requests multi-row session support, the WebSphere session manager breaks the session data across multiple rows as needed. This method allows WebSphere to support large session objects. Also, it provides a more efficient mechanism for storing and retrieving session contents under certain circumstances. See 25.8.5, “Single and multi-row schemas (database persistence)” on page 921 for information about this feature.

Using a cache lets the session manager maintain a cache of most recently used sessions in memory. Retrieving a user session from the cache eliminates a more expensive retrieval from the persistent store. The session manager uses a *least recently used* scheme for removing objects from the cache. Session data is stored to the persistent store based on your selections for write frequency and write option.

A third approach: WebSphere eXtreme Scale can be used as a shared in-memory cache for HTTP Session state instead of using WebSphere traditional shared database or memory-to-memory replication approach. This shared in-memory cache sits in a highly available replicated grid. The grid is not constrained to any one application server product or to any particular management unit, such as WebSphere Application Server Network Deployment cells. User sessions can be shared between any set of application servers, even across data centers, allowing a more reliable and fault-tolerant user session state. No application code change is required when using WebSphere eXtreme Scale to store and manage HTTP session data.

25.8.1 Enabling database persistence

We assume in this section that the following tasks are complete *before* enabling database persistence:

1. Create a session database.
2. (z/OS DB2) Create a table for the session data. Name the table *SESSIONS*. If you choose to use another name, update the web container custom property SessionTableName value to the new table name. Grant ALL authority for the server region user ID to the table. You can find an example of creating the table at the following website:
http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=/com.ibm.websphere.nd.multiplatform.doc/info/ae/ae/tprs_db2tzos.html

3. In distributed environments, the session table is created automatically when you define the data source for the database as the session management table; however, if you want to use a page (row) size greater than 4 KB, you need to create the table space manually. You can find an example of creating the table space at the following website:
http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=/com.ibm.websphere.nd.multiplatform.doc/info/ae/ae/tprs_db2t.html

4. Create a JDBC provider and data source for the database.

The data source should be non-XA enabled and must be a non-JTA enabled data source.

The JNDI name is used to specify the database for persistence (in this example, *jdbc/Sessions*).

A summary of the data source selections for a DB2 database on z/OS is shown at the end of the wizard, as shown in Figure 25-6.

Summary	
Summary of actions:	
Options	Values
Scope	cells:WTCell
Data source name	SessionDb
JNDI name	jdbc/Sessions
JDBC provider name	DB2 Universal JDBC Driver Provider
Description	One-phase commit DB2 JCC provider that supports JDBC 3.0. Data sources that use this provider support only 1-phase commit processing, unless you use driver type 2 with the application server for z/OS. If you use the application server for z/OS, driver type 2 uses RRS and supports 2-phase commit processing.
Class path	<code> \${DB2UNIVERSAL_JDBC_DRIVER_PATH}/db2jcc.jar \${UNIVERSAL_JDBC_DRIVER_PATH}/db2jcc_license_cu.jar \${DB2UNIVERSAL_JDBC_DRIVER_PATH}/db2jcc_license_cisuz.jar</code>
<code> \${DB2UNIVERSAL_JDBC_DRIVER_PATH}</code>	/pp/db2v8/UK39204/jcc/classes
<code> \${UNIVERSAL_JDBC_DRIVER_PATH}</code>	
Native path	<code> \${DB2UNIVERSAL_JDBC_DRIVER_NATIVEPATH}</code>
<code> \${DB2UNIVERSAL_JDBC_DRIVER_NATIVEPATH}</code>	/pp/db2v8/UK39204/jcc/lib
Implementation class name	com.ibm.db2.jcc.DB2ConnectionPoolDataSource
Driver type	2
Database name	DB8X
Server name	wtsc04.itso.ibm.com
Port number	33760
Use this data source in container managed persistence (CMP)	true
Component-managed authentication alias	WTDmNode/jdbc/db8x
Mapping-configuration alias	(none)
Container-managed authentication alias	WTDmNode/jdbc/db8x

Figure 25-6 Data source creation summary

Example: The following example illustrates the steps to enable database persistence at the application server level. Session management settings can also be performed at the enterprise application level and the web application level.

To enable database persistence, complete the following steps for each application server:

1. Click **Servers** → **Server Types** → **WebSphere application servers**. Then, select the server.
2. In the Container Settings section, click **Session management**.
3. Click **Distributed environment settings**.

4. Select **Database**, and click the **Database** link, as shown in Figure 25-7.

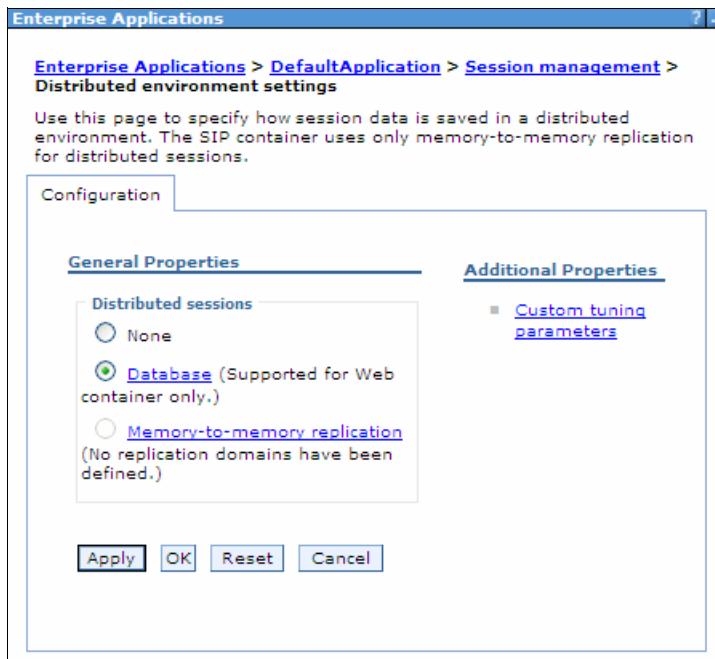


Figure 25-7 Distributed Environment Setting (database)

5. Enter the following database information:

- (z/OS) Enter the data source JNDI name (Figure 25-8), and click **OK**.

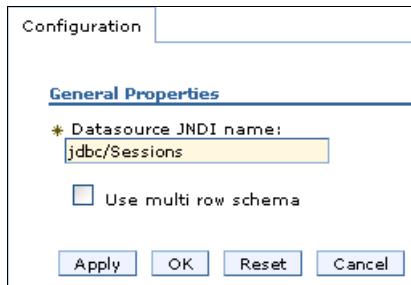


Figure 25-8 Data source name

- (Distributed platforms) Enter the information that is required to access the database (Figure 25-9).

If you are using DB2 and you anticipate requiring row sizes greater than 4 KB, select the appropriate value from the DB2 row size menu. If the DB2 row size is other than 4 KB, you are required to enter the name of the table space. See 25.8.4, “Larger DB2 page sizes and database persistence” on page 921 for more information.

If you intend to use a multi-row schema, select **Use Multi row schema**. See 25.8.5, “Single and multi-row schemas (database persistence)” on page 921 for more information.

Click **OK**.

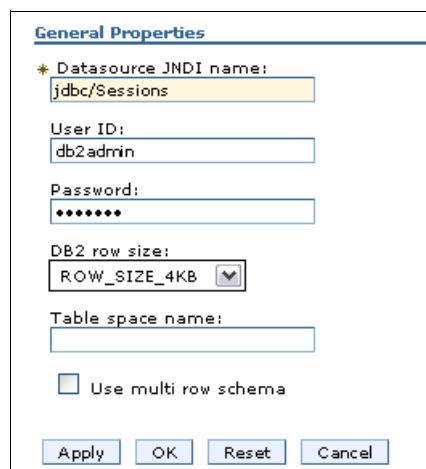


Figure 25-9 Database settings for session persistence

6. Click **OK** to accept the changes to the distribute session configuration.
7. Repeat these steps for each server in the cluster. Then, save the configuration changes, synchronize the changes with the servers, and restart the application servers.

25.8.2 Memory-to-memory replication

Memory-to-memory replication uses the data replication service to replicate data across many application servers in a cluster without using a database. Using this method, sessions are stored in the memory of an application server, providing the same functionality as a database for session persistence. Separate threads handle this functionality within an existing application server process.

The data replication service is an internal WebSphere Application Server component. In addition to its use by the session manager, it is also used to replicate dynamic cache data and stateful session beans across many application servers in a cluster.

Using memory-to-memory replication eliminates the impact and cost of setting up and maintaining a real-time production database. It also eliminates the single point of failure that can occur with a database. Session information between application servers is encrypted.

Memory-to-memory replication: Memory-to-memory replication requires the high availability (HA) manager to be active. HA manager is active by default but can be disabled. For more information, see the “When to use a high availability manager” topic at the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.nd.multiplatform.doc/info/ae/ae/crun_ha_ham_required.html

Data replication service modes

The memory-to-memory replication function is accomplished by the creation of a data replication service instance in an application server that communicates to other data replication service instances in remote application servers.

You can set up a replication service instance to run in any of the following modes:

- ▶ Server mode

In this mode, a server stores only backup copies of other application server sessions. It does not send copies of sessions that are created in that particular server.

- ▶ Client mode

In this mode, a server broadcasts or sends only copies of the sessions it owns. It does not receive backup copies of sessions from other servers.

- ▶ Both mode

In this mode, the server simultaneously sends copies of the sessions it owns and acts as a backup table for sessions that are owned by other application servers.

You can select the replication mode of server, client, or both when configuring the session management facility for memory-to-memory replication. The default is both modes.

With respect to mode, these are the primary examples of memory-to-memory replication configuration:

- ▶ Peer-to-peer replication
- ▶ Client/server replication

Although the administrative console allows flexibility and additional possibilities for memory-to-memory replication configuration, only these configurations are officially supported.

There is a single replica in a cluster by default. You can modify the number of replicas through the replication domain.

Peer-to-peer topology

Figure 25-10 shows an example of peer-to-peer topology. Each application server stores sessions in its own memory. It also stores sessions to and retrieves sessions from other application servers. Each application server acts as a client by retrieving sessions from other application servers. Each application server also acts as a server by providing sessions to other application servers.

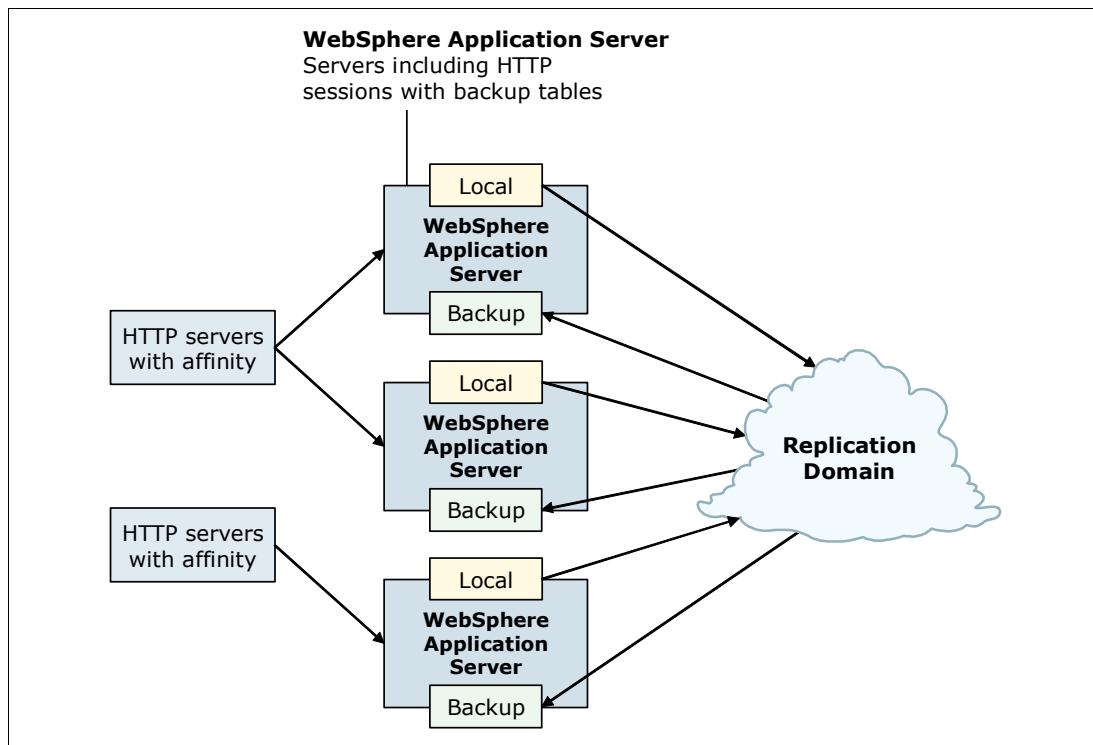


Figure 25-10 Example of peer-to-peer topology

The basic peer-to-peer (both mode) topology is the default configuration and has a single replica. However, you can also add additional replicas by configuring the replication domain.

In this basic peer-to-peer topology, each application server can:

- ▶ Host the web application using the HTTP session
- ▶ Send changes to the HTTP session that it owns
- ▶ Receive backup copies of the HTTP session from all of the other servers in the cluster

This configuration represents the most consolidated topology, where the various system parts are collocated and requires the fewest server processes. When using this configuration, the most stable implementation is achieved when each node has equal capabilities (CPU, memory, and so on) and when each handles the same amount of work.

The advantage of this topology is that no additional processes and products are required to avoid a single point of failure, thus reducing the time and cost that are required to configure and maintain additional processes or products.

One of the disadvantages of this topology is that it can consume large amounts of memory in networks with many users, because each server has a copy of all sessions. For example, assuming that a single session consumes 10 KB and one million users are logged in to the system, each application server consumes 10 GB of memory to keep all sessions in its own memory. Another disadvantage is that every change to a session must be replicated to all application servers. This replication can cause a performance impact.

Client/server topology

Figure 25-11 shows an example of client/server topology. In this setup, application servers act as either a replication client or a server. Those that act as replication servers store sessions in their own memory and provide session information to clients. They are dedicated replication servers that just store sessions but do not respond to the users' requests. Client application servers send session information to the replication servers and retrieve sessions from the servers. They respond to user requests and store only the sessions of the users with whom they interact.

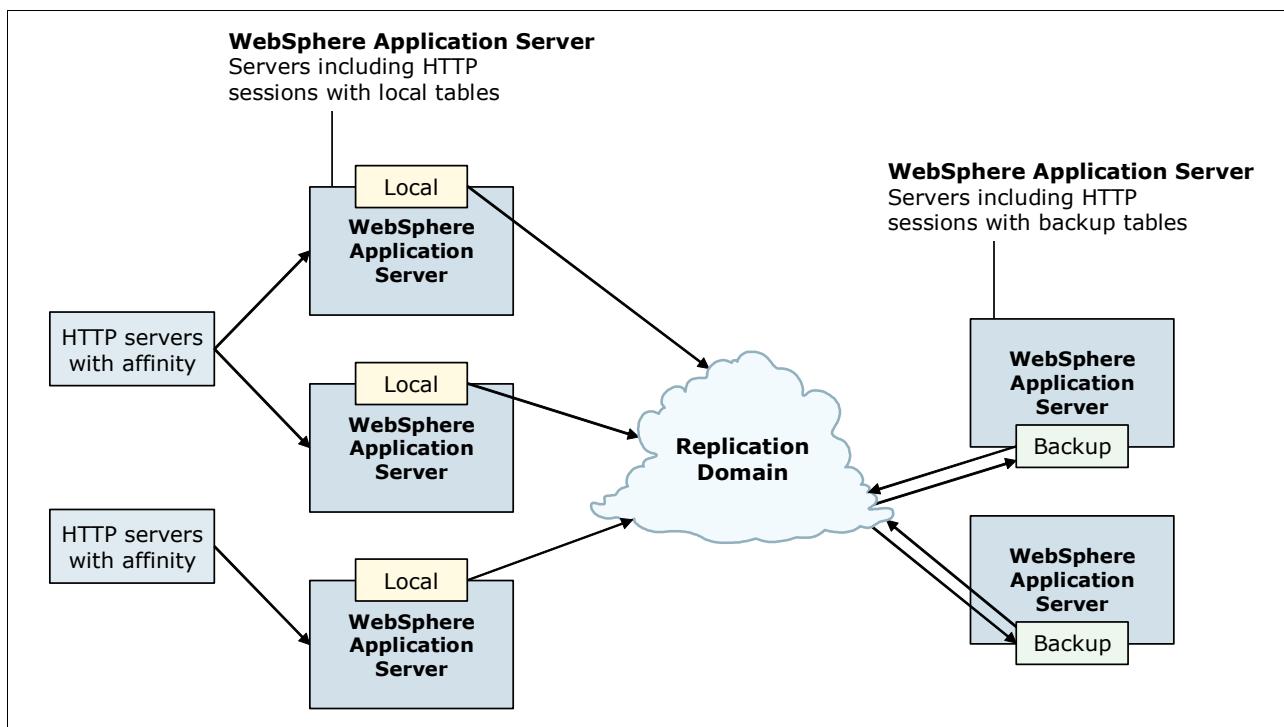


Figure 25-11 Example of client/server topology

The advantage of this topology is that it clearly distinguishes the role of client and server. Only replication servers keep all sessions in their memory and only the clients interact with users. This process reduces the consumption of memory on each application server and reduces the performance impact, because session information is sent only to the servers.

You can recycle a backup server without affecting the servers that are running the application. When there are two or more backups, failure recovery is possible. Conversely, you can recycle an application server without losing the backup data.

When running web applications on lower-end hardware, you can choose to have one or two more powerful computers that have the capacity to run a couple of session managers in replication server mode, allowing you to reduce the load on the web application hardware.

One of the disadvantages of this topology is that additional application servers have to be configured and maintained over and above those servers that interact with users. Have multiple replication servers configured to avoid a single point of failure.

Replication domain

The memory-to-memory replication function is accomplished by the creation of a data replication service instance in an application server that communicates to other data replication service instances in remote application servers. You must configure this data replication service instance as a part of a replication domain.

Data replication service instances on disparate application servers that replicate to one another must be configured as a part of the same domain. You must configure all session managers connected to a replication domain to have the same topology. If one session manager instance in a domain is configured to use the client/server topology, then the remainder of the session manager instances in that domain must be a combination of servers that are configured as client only and server only.

If one session manager instance is configured to use the peer-to-peer topology, then all session manager instances must be configured as both client and server. For example, a server-only data replication service instance and a both client and server data replication service instance cannot exist in the same replication domain. Multiple data replication service instances that exist on the same application server due to session manager memory-to-memory configuration at various levels that are configured to be part of the same domain must have the same mode.

Create a separate replication domain for each consumer. For example, create one replication domain for session manager and another replication domain for dynamic cache. Configure one replication domain only when you configure session manager replication and stateful session bean failover. Using one replication domain in this situation ensures that the backup state information of HTTP sessions and stateful session beans are on the same application server.

Enabling memory-to-memory replication

We assume in this section that the following tasks have been completed before enabling data for the replication service:

1. You have created a cluster consisting of at least two application servers.

In this example, we are working with a cluster called *MyCluster*. It has two servers, *server1* and *server2*.

2. You have installed applications to the cluster.

Note: This example illustrates setting up the replication domain and replicators after the cluster is created. You have the option of configuring the “http session memory-to-memory replication” option in the first step when you create the cluster. If you choose this option, the cluster members are created in both client and server replication mode.

To enable memory-to-memory replication, complete the following steps:

1. Create a replication domain to define the set of replicator processes that communicate with each other:
 - a. Click **Environment** → **Replication domains**.
 - b. Click **New**. Then, enter the following information, as shown in Figure 25-12:

- Name

At a minimum, you need to enter a name for the replication domain. The name must be unique within the cell. In this example, we used *MyClusterRepDomain* as the name and used the defaults for the other properties.

- Number of replicas

A single replica allows you to replicate a session to only one other server, which is the default. When you choose this option, a session manager picks another session manager that is connected to the same replication domain with which to replicate the HTTP session during session creation. All updates to the session are replicated to that single server. This option is set at the replication domain level.

When this option is set, every session manager that is connected to this replication domain creates a single backup copy of the HTTP session state information about a backup server.

Alternatively, you can replicate to every application server that is configured as a consumer of the replication domain using the “Entire Domain” option, or you can replicate to a specified number of replicas within the domain.

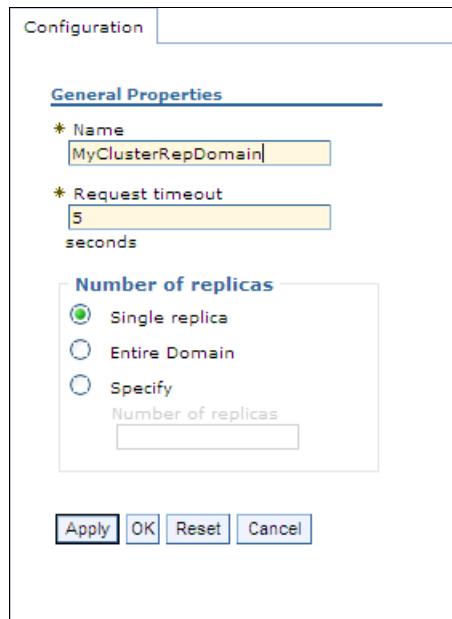


Figure 25-12 Create a replication domain

- c. Click **Apply**. Then, click **OK**.
 - d. Save the configuration changes.
2. Configure the cluster members:
- Complete the following steps for each application server:
- a. Click **Servers** → **Server Types** → **WebSphere application servers**.
 - b. Click the application server name. In this example, **server1** and **server2** are selected as application servers, respectively.
 - c. In the Container Settings section, click **Session management**.
 - d. Click **Distributed environment settings**.

- e. Click the **Memory-to-memory replication** property (Figure 25-13).

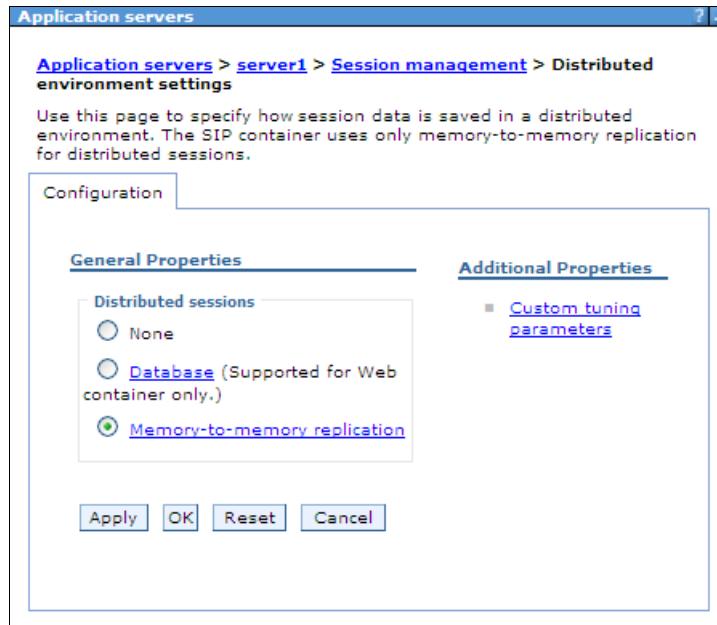


Figure 25-13 Distributed environment settings

- f. Choose a replication domain.

Select the replication topology by specifying the replication mode. Clicking **Both client and server** identifies this topology as a peer-to-peer topology. In a client/server topology, click **Client only** for application servers that will be responding to user requests. Click **Server only** for those servers that will be used as replication servers. See Figure 25-14.

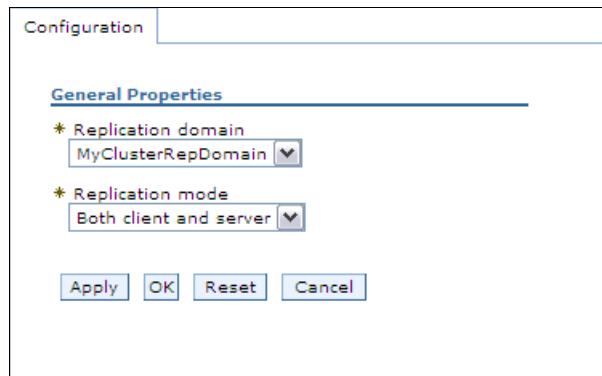


Figure 25-14 Data replication service settings

- g. Click **OK**.

3. Repeat steps 1 on page 912 to 2 on page 913 for the remainder of the application servers in the cluster.
4. Save the configuration, and restart the cluster. You can restart the cluster by clicking **Servers** → **Clusters** → **WebSphere application server clusters**. Select the cluster, and then click **Stop**. After the messages indicate that the cluster has stopped, click **Start**.

HTTP session replication in the z/OS controller

Websphere Application Server V8 for Z/OS can store replicated HTTP session data in the controller and can replicate data to other WebSphere Application Servers. HTTP session data stored in a controller is retrievable by any of the servants of that controller. HTTP session affinity is still associated to a particular servant; however, if that servant should fail, any of the other servants can retrieve the HTTP session data stored in the controller and establish a new affinity.

The capability of storing HTTP sessions in the controller can also be enabled in unmanaged application servers on z/OS. When this capability is enabled, servants store the HTTP session data in the controller for retrieval when a servant fails, which is similar to managed servers. HTTP session data stored in the controller of an unmanaged application server is not retrievable by other application servers and is not replicated to other application servers.

To store HTTP session data in the controller in an unmanaged application server, complete the following steps:

1. Click **Servers** → **Server Types** → **WebSphere application servers** → **server_name**.
2. Under Server Infrastructure, click **Java and Process Management** → **Process Definition** → **Servant** → **Java Virtual Machine** → **Custom Properties**.
3. Enter **HttpSessionEnableUnmanagedServerReplication** as the name, and enter true as the value, as shown in Figure 25-15.

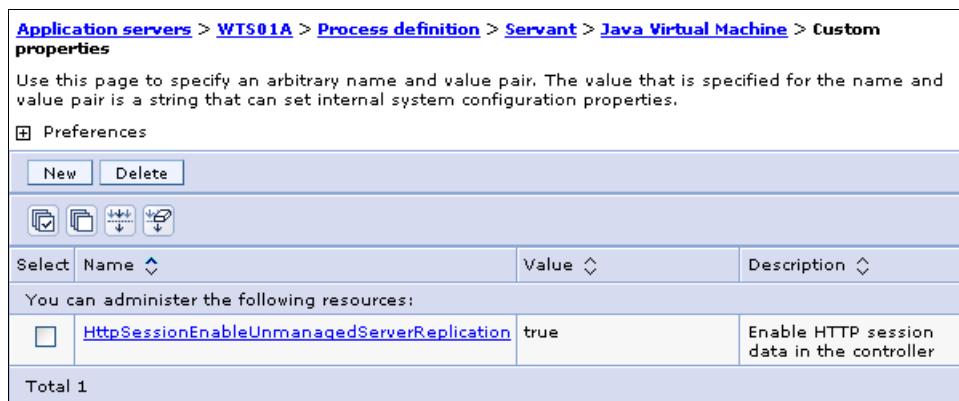


Figure 25-15 Enable HTTP session in the controller

4. Save and restart the application server.

25.8.3 Session management tuning

Performance tuning for session management persistence consists of defining the following conditions:

- ▶ How often session data is written (write frequency settings)
- ▶ How much data is written (write contents settings)
- ▶ When the invalid sessions are cleaned up (session cleanup settings)

These settings are set in the “Custom tuning parameters” option found under the Additional Properties section for session management settings. Several combinations of these settings are predefined and available for selection, or you can customize them. These options are available by clicking **Servers** → **Server Types** → **WebSphere application servers** → **server_name** → **Session management** → **Distributed environment settings** → **Custom tuning parameters**.

As shown in Figure 25-16, the options use the following default values:

- ▶ Very high (optimize for performance)

Write frequency	Time based
Write interval	300 seconds
Write contents	Only updated attributes
Schedule sessions cleanup	true
First time of day default	0
Second time of day default	2

- ▶ High

Write frequency	Time based
Write interval	300 seconds
Write contents	All session attributes
Schedule sessions cleanup	false

- ▶ Medium

Write frequency	End of servlet service
Write contents	Only updated attributes
Schedule sessions cleanup	false

- ▶ Low (optimize for failover)

Write frequency	End of servlet service
Write contents	All session attributes
Schedule sessions cleanup	false

- ▶ Custom settings

Write frequency default	Time based
Write interval default	10 seconds
Write contents default	All session attributes
Schedule sessions cleanup default	false

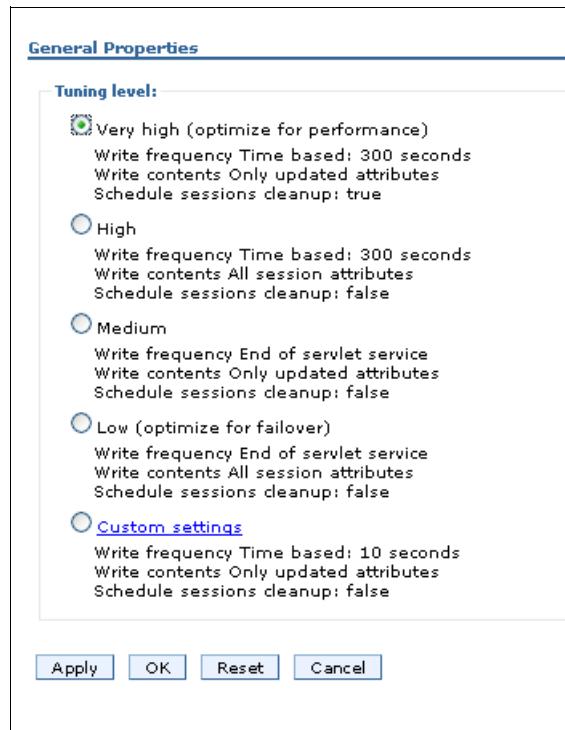


Figure 25-16 Distributed session management tuning levels

Custom settings

You can customize the settings by selecting **Custom settings**, as shown in Figure 25-17.

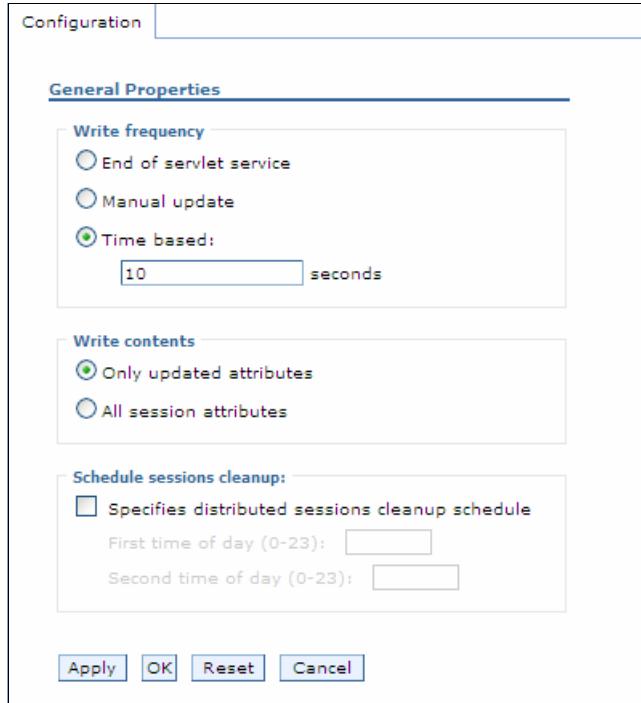


Figure 25-17 Session management tuning parameters

The following sections go into more detail about these custom settings.

Writing frequency settings

You can select from the following settings that determine how often session data is written to the persistent data store:

- ▶ End of servlet service

If the session data has changed, it is written to the persistent store after the servlet finishes processing an HTTP request.

- ▶ Manual update

The session data is written to the persistent store when the `sync()` method is called on the `IBMSession` object.

- ▶ Time-based

The session data is written to the persistent store based on the specified write interval value.

Fast access time attribute: The last access time attribute is updated each time the session is accessed by the servlet or JSP, whether or not the session is changed. This update is done to make sure the session does not time out:

- ▶ If you choose the End of servlet service option, each servlet or JSP access results in a corresponding persistent store update of the last access time.
- ▶ If you select the Manual update option, the update of the last access time in persistent store occurs on a `sync()` call or at later time.
- ▶ If you use time-based updates, the changes are accumulated and written in a single transaction. This option can significantly reduce the amount of I/O to the persistent store.

See 25.12.2, “Reducing persistent store I/O” on page 931 for options to change this database update behavior.

Consider an example where the web browser accesses the application once every 5 seconds:

- ▶ In “End of servlet service” mode, the session is written out every 5 seconds.
- ▶ In “Manual update” mode, the session is written out when the servlet issues `IBMSession.sync()`. It is the responsibility of the servlet writer to use the `IBMSession` interface instead of the `HttpSession` Interface, and the servlets or JSP pages must be updated to issue the `sync()` method.
- ▶ In “Time-based” mode, the servlet or JSP does not need to use the `IBMSession` class or issue the `IBMSession.sync()` method. If the write interval is set to 120 seconds, the session data is written out at most every 120 seconds.

End of servlet service

When the write frequency is set to the end of servlet service option, WebSphere writes the session data to the persistent store at the completion of the `HttpServlet.service()` method call. The write content settings determine output.

Manual update

In manual update mode, the session manager sends only changes to the persistent data store if the application explicitly requests a save of the session information.

Note: Manual updates use an IBM extension to `HttpSession` that is not part of the Servlet 2.5 API.

Manual update mode requires an application developer to use the `IBMSession` class for managing sessions. When the application invokes the `sync()` method, the session manager writes the modified session data and last access time to the persistent store. The session data that is written to the persistent store is controlled by the write contents option selected.

If the servlet or JSP terminates without invoking the `sync()` method, the session manager saves the contents of the session object into the session cache (if caching is enabled), but does not update the modified session data in the session database. The session manager updates only the last access time in the persistent store asynchronously, at a later time.

Example 25-5 shows how to use the `IBMSession` class to manually update the persistent store.

Example 25-5 Using the `IBMSession` class to manually update the persistent store

```
public void service (HttpServletRequest req, HttpServletResponse res)
    throws ServletException, IOException
{
    // Use the IBMSession to hold the session information
    // We need the IBMSession object because it has the manual update
    // method sync()
    com.ibm.websphere.servlet.session.IBMSession session =
        (com.ibm.websphere.servlet.session.IBMSession)req.getSession(true);

    Integer value = 1;

    //Update the in-memory session stored in the cache
    session.putValue("MyManualCount.COUNTER", value);

    //The servlet saves the session to the persistent store
    session.sync();
}
```

This interface gives the web application developer additional control over when and if session objects go to the persistent data store. If the application does not invoke the `sync()` method and if the manual update mode is specified, the session updates go only to the local session cache, not the persistent data store. Web developers use this interface to reduce unnecessary writes to the session database and, thereby, to improve overall application performance.

All servlets in the web application server must perform their own session management in manual update mode.

Time-based writes to the session database

Using the time-based write option writes session data to the persistent store at a defined write interval.

WebSphere provides the session affinity mechanism that assures an HTTP request is routed to the web application that handles its `HttpSession`. This assurance still holds in a WLM environment when using persistent `HttpSession`. Thus, the necessity to write the session data immediately to the persistent store can be relaxed somewhat in these environments as well as in non-clustered environments, because the persistent store is used only for failover and session cache full scenarios.

With this in mind, you can possibly gain potential performance improvements by reducing the frequency of persistent store writes.

Note: Time-based writes requires session affinity for session data integrity.

The following details apply to time-based writes:

- ▶ The expiration of the write interval does not necessitate a write to the persistent store unless the session has been touched (the `getAttribute()`, `setAttribute()`, or `removeAttribute()` method was called since the last write).

- ▶ If a session write interval has expired and the session has only been retrieved (that is, `request.getSession()` was called since the last write), then the last access time is written to the persistent store regardless of the write contents setting.
- ▶ If a session write interval has expired and the session properties have been either accessed or modified since the last write, then the session properties are written in addition to the last access time. The session properties that get written are dependent on the write contents settings.
- ▶ Time-based write allows the servlet or JSP to issue the `IBMSession.sync()` method to force the write of session data to the database.
- ▶ If the time between session servlet requests for a particular session is greater than the write interval, the session effectively gets written after each service method invocation.
- ▶ The session cache should be large enough to hold all of the active sessions. Otherwise, extra persistent store writes occur, because the receipt of a new session request can result in writing out the oldest cached session to the persistent store. To put it another way, if the session manager has to remove the least recently used HttpSession from the cache during a full cache scenario, the session manager will write that HttpSession using the write contents settings upon removal from the cache.
- ▶ The session invalidation time must be at least twice the write interval to ensure that a session is not inadvertently invalidated prior to being written to the persistent store.
- ▶ A newly created session is always written to the persistent store at the end of the service method.

Writing content settings

The writing content settings options control what is written. Refer to 25.8.6, “Contents written to the persistent store using a database” on page 923 before selecting one of the options, because there are several factors to decide. The following options are available:

- ▶ Only updated attributes are written to the persistent store
- ▶ All session attributes are written to the persistent store

Session cleanup settings

WebSphere allows the administrator to defer (to off-hours) the clearing of invalidated sessions from the persistent store. *Invalidate sessions* are sessions that are no longer in use and timed out. For more information, see 25.9, “Invalidate sessions” on page 925.

Select the **Specifies distributed sessions cleanup schedule** option to enable this feature.

The cleanup can be done either once or twice a day. The following fields are available:

- ▶ First time of day (0-23) is the first hour, during which the invalidated persistent sessions will be cleared from the persistent store. This value must be a positive integer between 0 and 23.
- ▶ Second time of day (0-23) is the second hour, during which the invalidated persistent sessions will be cleared from the persistent store. This value must be a positive integer between 0 and 23.

Consider using scheduled invalidation for intranet-style applications that have a somewhat fixed number of users wanting the same HTTP session for the whole business day.

25.8.4 Larger DB2 page sizes and database persistence

WebSphere supports 4 KB, 8 KB, 16 KB, and 32 KB page sizes and can have larger varchar for bit data columns of 7 KB, 15 KB, or 31 KB. Using this performance feature, we see faster persistence for HttpSession of sizes from 7 KB to 31 KB in the single-row case or attribute sizes from, 4 KB to 31 KB in the multi-row case.

Enabling the use of greater than 4 KB page size involves dropping any existing table that was created with a 4 KB buffer pool and table space. This concept also applies if you subsequently change between 4 KB, 8 KB, 16 KB, or 32 KB.

To use a page size other than the default 4 KB, complete the following steps:

1. If the SESSIONS table already exists, drop it from the DB2 database by running the following commands:

```
DB2 connect to session
DB2 drop table sessions
```
2. Create a new DB2 buffer pool and table space, specify the same page size (8 KB, 16 KB, or 32 KB) for both, and assign the new buffer pool to this table space.

Example 25-6 shows the steps for creating an 8 KB page.

Example 25-6 Creating an 8 KB page size

```
DB2 connect to session
DB2 CREATE BUFFERPOOL sessionBP SIZE 1000 PAGESIZE 8K
DB2 connect reset
DB2 connect to session
DB2 CREATE TABLESPACE sessionTS PAGESIZE 8K MANAGED BY SYSTEM USING
('D:\DB2\NODE0000\SQL00005\sessionTS.0') BUFFERPOOL sessionBP
DB2 connect reset
```

Refer to the DB2 product documentation for details.

3. Configure the correct table space name and page size and DB2 row size in the session management database configuration. (Refer to Figure 25-9 on page 908.)
4. Restart WebSphere. On start, the session manager creates a new SESSIONS table based on the page size and table space name that you specified.

25.8.5 Single and multi-row schemas (database persistence)

When using the single-row schema, each user session maps to a single database row, which is WebSphere's default configuration for persistent session management. This setup includes hard limits to the amount of user-defined, application-specific data that WebSphere Application Server can access.

When using the multi-row schema, each user session maps to multiple database rows, with each session attribute mapping to a database row.

In addition to allowing larger session records, using a multi-row schema can yield performance benefits, as described in 25.12.3, "Multirow persistent sessions: Database persistence" on page 931.

Switching from single-row to multi-row schema

To switch from a single-row to multi-row schema for sessions, complete the following steps:

1. Modify the session manager properties to switch from single to multi-row schema. Click **Servers** → **Server Types** → **WebSphere application servers** → **server_name** → **Session management** → **Distributed environment settings** → **Database settings**. Then, select the **Use multi row schema** option.
2. Manually drop the database table or delete all the rows in the session database table. To drop the table:
 - a. Determine which data source the session manager is using. This value is set in the session management distributed settings window. Refer to 25.8.1, “Enabling database persistence” on page 905 for more information.
 - b. Look up the database name in the data source settings. Find the JDBC provider and then the data source. The database name is in the custom settings.
 - c. Use the database facilities to connect to the database and drop it.
3. Restart the application server or cluster.

Design considerations

Consider configuring direct, single-row usage to one database and multi-row usage to another database while you verify which option suits your application's specific needs. You can achieve this configuration by switching the data source used and then monitoring the performance.

Table 25-2 provides an overview.

Table 25-2 Single versus multi-row schemas

Programming issue	Application scenario
Reasons to use single-row	You can read/write all values with just one record read/write, which takes up less space in a database, because you are guaranteed that each session is only one record long.
Reasons <i>not</i> to use single-row	There is a 2 MB limit of stored data per session. The sum of sizes of all session attributes is limited to 2 MB.
Reasons to use multi-row	The application can store an unlimited amount of data. You are limited only by the size of the database and a 2 MB-per-record limit. The size of each session attribute can be 2 MB. The application can read individual fields instead of the entire record. When large amounts of data are stored in the session but only small amounts are specifically accessed during a given servlet's processing of an HTTP request, multi-row sessions can improve performance by avoiding unneeded Java object serialization.
Reasons <i>not</i> to use multi-row	If data is small in size, you probably do not want the extra impact of multiple row reads when everything can be stored in one row.

In the case of multi-row usage, design your application data objects so they do not have references to each other. This prevents circular references.

For example, suppose you are storing two objects (A and B) in the session using `HttpSession.put(...)`, and object A contains a reference to object B. In the multi-row case, because objects are stored in different rows of the database, when objects A and B are retrieved later, the object graph between objects A and B is different from the object graph that is stored. Objects A and B behave as independent objects.

25.8.6 Contents written to the persistent store using a database

WebSphere supports the following modes for writing session contents to the persistent store:

- ▶ Only updated attributes

Write only the HttpSession properties that have been updated using `setAttribute()` and `removeAttribute()`.

- ▶ All session attributes

Write all the HttpSession properties to the database.

You can find these settings in the tuning parameters for distributed environment settings (select **Custom Settings**).

When a new session is initially created with either of these options, the entire session is written, including any Java objects that are bound to the session. When using database persistence, the behavior for subsequent servlet or JSP requests for this session varies depending on whether the single-row or multi-row database mode is in use as follows:

- ▶ In single-row mode, choose from the following options:

- Only updated attributes

If any session attribute has been updated, through `setAttribute()` or `removeAttribute()`, then all of the objects bound to the session will be written to the database.

- All session attributes

All bound session attributes are written to the database.

- ▶ In multi-row mode, choose from the following options:

- Only updated attributes

Only the session attributes that were specified using `setAttribute()` or `removeAttribute()` are written to the database.

- All session attributes

All of the session attributes that reside in the cache are written to the database. If the session has never left the cache, then this cache should contain all of the session attributes.

By using the “All session attributes” mode, servlets and JSP pages can change Java objects that are attributes of the HttpSession without having to call the `setAttribute()` method on the HttpSession for that Java object in order for the changes to be reflected in the database. Using the “All session attributes” mode provides some flexibility to the application programmer and protects against any possible side effects of moving from local sessions to persistent sessions. However, using this mode can potentially increase activity and be a performance drain.

Evaluate both the pros and cons for your installation. Note that the combination of the “All session attributes” mode with a time-based write can greatly reduce the performance penalty and essentially give you the best of both worlds.

Example 25-7 and Example 25-8 illustrate the initial session creation with a `setAttribute()` method. Subsequent requests for that session do not need to use the `setAttribute()` method.

Example 25-7 Initial servlet

```
HttpSession sess = request.getSession(true);
myClass myObject = new myClass();
myObject.someInt = 1;
sess.setAttribute("myObject", myObject); // Bind object to the session
```

Example 25-8 Subsequent servlet

```
HttpSession sess = request.getSession(false);
myObject = sess.getAttribute("myObject"); // get bound session object
myObject.someInt++; // change the session object
// setAttribute() not needed with write "All session attributes" specified
```

Example 25-9 and Example 25-10 show that the `setAttribute()` method is still required, even though the write all session attributes option is enabled.

Example 25-9 Initial servlet

```
HttpSession sess = request.getSession(true);
String myString = new String("Initial Binding of Session Object");
sess.setAttribute("myString", myString); // Bind object to the session
```

Example 25-10 Subsequent servlet

```
HttpSession sess = request.getSession(false);
String myString = sess.getAttribute("myString"); // get bound session object
...
myString = new String("A totally new String"); // get a new String object
sess.setAttribute("myString", myString); // Need to bind the object to the session since a NEW Object is used
```

Table 25-3 summarizes the action of the `HttpSession setAttribute()` and `removeAttribute()` methods for various combinations of the row type, write contents, and write frequency session persistence options.

Table 25-3 Write contents versus write frequency

Row type	Write contents	Write frequency	Action for setAttribute	Action for removeAttribute
Single-row	Only updated attributes	End of servlet service / <code>sync()</code> call with Manual update	If any of the session data has changed, then write all of this session's data from cache. ^a	If any of the session data has changed, then write all of this session's data from cache. ^a

Row type	Write contents	Write frequency	Action for setAttribute	Action for removeAttribute
Single-row	Only updated attributes	Time-based	If any of the session data has changed, then write all of this session's data from cache. ^a	If any of the session data has changed, then write all of this session's data from cache. ^a
	All session attributes	End of servlet service / <code>sync()</code> call with Manual update	Always write all of this session's data from cache.	Always write all of this session's data from cache.
		Time-based	Always write all of this session's data from cache.	Always write all of this session's data from cache.
Multi-row	Only updated attributes	End of servlet service / <code>sync()</code> call with Manual update	Write only thread-specific data that has changed.	Delete only thread-specific data that has been removed.
		Time-based	Write thread-specific data that has changed for <i>all</i> threads using this session.	Delete thread-specific data that has been removed for <i>all</i> threads using this session.
	All session attributes	End of servlet service / <code>sync()</code> call with Manual update	Write all session data from cache.	Delete thread-specific data that has been removed for <i>all</i> threads using this session.
		Time-based	Write all session data from cache.	Delete thread-specific data that has been removed for <i>all</i> threads using this session.

a. When a session is written to the database while using single-row mode, all of the session data is written.

Therefore, no database deletes are necessary for properties removed with the `removeAttribute()` method, because the write of the entire session does not include removed properties.

Multi-row mode has the notion of thread-specific data. *Thread-specific data* is defined as session data that was added or removed while executing under this thread. If you use the “End of servlet service” or “Manual update” modes and enable the “Only updated attributes” option, only the thread-specific data is written to the database.

25.9 Invalidating sessions

This section discusses how to invalidate sessions when the user no longer needs the session object, for example, when the user has logged off a site. Invalidating a session removes it from the session cache and from the persistent store.

WebSphere offers the following methods for invalidating session objects:

- ▶ Programmatically, you can use the `invalidate()` method on the session object. If the session object is accessed by multiple threads in a web application, be sure that none of the threads still have references to the session object.
- ▶ An invalidator thread scans for timed-out sessions every *n* seconds, where *n* is configurable from the administrative console. The session timeout setting is in the general properties of the session management settings.

- ▶ For persistent sessions, the administrator can specify times when the scan runs. This feature has the following benefits when used with a persistent session:
 - Persistent store scans can be scheduled during periods that normally have low demand. This method avoids slowing down online applications due to contention in the persistent store.
 - When this setting is used with the “End of servlet service write frequency” option, WebSphere does not have to write the last access time with every HTTP request. The reason is that WebSphere does not have to synchronize the invalidator thread’s deletion with the HTTP request access.

You can find the schedule sessions cleanup setting in the “Session management” settings under the “Custom tuning” parameters for distributed environments (select **Custom Settings**).

If you are going to use session cleanup, be aware of these considerations:

- HttpSession timeouts are not enforced. Instead, all invalidation processing is handled at the configured invalidation times.
- With listeners, processing is potentially delayed by this configuration. Do not use session cleanup scheduling if you use listeners. We discuss session listeners in 25.10, “Session listeners” on page 926.

25.10 Session listeners

Listener classes are defined to listen for state changes of a session and its attributes. This “listening” allows greater control over interactions with sessions, so that programmers can monitor creation, deletion, and modification of sessions. Programmers can perform initialization tasks when a session is created or can clean up tasks when a session is removed. It is also possible to perform some specific tasks for the attribute when an attribute is added, deleted, or modified.

The following listener interfaces are used to monitor the events that are associated with the HttpSession object:

- ▶ `javax.servlet.http.HttpSessionListener`
Use this interface to monitor creation and deletion, including session timeout, of a session.
- ▶ `javax.servlet.http.HttpSessionAttributeListener`
Use this interface to monitor changes of session attributes, such as add, delete, and replace.
- ▶ `javax.servlet.http.HttpSessionActivationListener`
Use this interface to monitor sessions that are made active or that are made passive. This interface is useful to monitor if the session exists, whether in memory or not, when persistent session is used.

Table 25-4 gives a summary of the interfaces and methods.

Table 25-4 Listener interfaces and their methods

Target	Event	Interface	Method
session	create	HttpSessionListener	sessionCreated()
	destroy	HttpSessionListener	sessionDestroyed()
	activate	HttpSessionActivationListener	sessionDidActivate()
	passivate	HttpSessionActivationListener	sessionWillPassivate()
attribute	add	HttpSessionAttributeListener	attributeAdded()
	remove	HttpSessionAttributeListener	attributeRemoved()
	replace	HttpSessionAttributeListener	attributeReplaced()

For more information, see *Java Platform Enterprise Edition, v 5.0 API Specifications* at the following website:

<http://java.sun.com/javaee/5/docs/api/>

25.11 Session security

Websphere Application Server V8 maintains the security of individual sessions. When session manager integration with WebSphere security is enabled, the session manager checks the user ID of the HTTP request against the user ID of the session held within WebSphere. This check is done as part of the processing of the `request.getSession()` function. If the check fails, WebSphere throws the following exception:

`com.ibm.websphere.servlet.session.UnauthorizedSessionRequestException`

If the check succeeds, the session data is returned to the calling servlet or JSP.

Session security checking works with the standard HttpSession. The identity or user name of a session can be accessed through the `com.ibm.websphere.servlet.session.IBMSession` interface. An unauthenticated identity is denoted by the user name *anonymous*.

The session manager uses WebSphere's security infrastructure to determine the authenticated identity associated with a client HTTP request that either retrieves or creates a session.

Session management security has the following rules:

- ▶ Sessions in unsecured pages are treated as accesses by the anonymous user.
- ▶ Sessions created in unsecured pages are created under the identity of that anonymous user.
- ▶ Sessions in secured pages are treated as accesses by the authenticated user.
- ▶ Sessions created in secured pages are created under the identity of the authenticated user. They can only be accessed in other secured pages by the same user. To protect these sessions from use by unauthorized users, they cannot be accessed from an unsecure page. Do not mix access to secure and unsecure pages.

Table 25-5 lists possible scenarios when security integration is enabled, where outcomes depend on whether the HTTP request was authenticated and whether a valid session ID and user name was passed to the session manager.

Table 25-5 HTTP session security

Request session ID / user name.	Unauthenticated HTTP request is used to retrieve the session.	Authenticated HTTP request is used to retrieve the session. The user ID in the HTTP request is FRED.
No session ID was passed in for this request, or the ID is for a session that is no longer valid.	A new session is created. The user name is anonymous.	A new session is created. The user name is FRED.
A valid session ID is received. The current session user name is anonymous.	The session is returned.	The session is returned. The session manager changes the user name to FRED.
A valid session ID is received. The current session user name is FRED.	The session is not returned. UnauthorizedSessionRequestException is thrown. ^a	The session is returned.
A valid session ID is received. The current session user name is BOB.	The session is not returned. UnauthorizedSessionRequestException is thrown. ^a	The session is not returned. UnauthorizedSessionRequestException is thrown. ^a

a. A com.ibm.websphere.servlet.session.UnauthorizedSessionRequestException is thrown to the servlet or JSP.

Refer to 25.5, “General properties for session management” on page 898 for more information about the security integration setting.

25.12 Session performance considerations

This section includes guidance for developing and administering scalable, high-performance web applications using Websphere Application Server V8 session support.

25.12.1 Session size

Large session objects pose several problems for a web application. If the site uses session caching, large sessions reduce the memory that is available in the WebSphere instance for other tasks, such as application execution.

For example, assume that a given application stores 1 MB of information for each user session object. If 100 users arrive over the course of 30 minutes, and assume the session timeout remains at 30 minutes, the application server instance must allocate 100 MB just to accommodate the newly arrived users in the session cache:

$$1 \text{ MB for each user session} * 100 \text{ users} = 100 \text{ MB}$$

Note this number does not include previously allocated sessions that have not timed out yet. The memory that is required by the session cache can be considerably higher than 100 MB.

Web developers and administrators have several options for improving the performance of session management:

- ▶ Reducing session object size
- ▶ Reducing the session cache size
- ▶ Creating additional application servers
- ▶ Invalidating unneeded sessions
- ▶ Increasing available memory
- ▶ Reducing the session timeout interval

Reducing session object size

Web developers must consider carefully information that is kept by the session object:

- ▶ Remove information that is obtained or derived easily to keep the session object small.
- ▶ Remove unnecessary, unneeded, or obsolete data from the session.
- ▶ Consider whether it is better to keep a certain piece of data in an application database rather than in the HTTP session. This option gives the developer full control over when the data is fetched or stored and how it is combined with other application data. Web developers can use the power of SQL if the data is in an application database.

Reducing object size becomes particularly important when persistent sessions are used. Serializing a large amount of data and writing it to the persistent store requires significant WebSphere performance impact. Even if the option to write only updated attributes is enabled, if the session object contains large Java objects or collections of objects that are updated regularly, there is a significant performance penalty in persisting these objects. This penalty can be reduced by using time-based writes.

Best performance with session objects: In general, you can obtain the best performance with session objects that are less than 2 KB in size. When the session object exceeds 4 to 5 KB, you can expect a significant decrease in performance.

Even if session persistence is not an issue, minimizing the session object size can help to protect your web application from scale-up disasters as user numbers increase. Large session objects require more and more JVM memory, leaving no room to run servlets.

Refer to 25.8.4, “Larger DB2 page sizes and database persistence” on page 921 to learn how WebSphere can provide faster persistence of larger session objects when using DB2.

Reducing the session cache size

The session manager allows administrators to change the session cache size to alter the cache’s memory footprint. By default, the session cache holds 1000 session objects. By lowering the number of session objects in the cache, the administrator reduces the memory required by the cache.

However, if the user’s session is not in the cache, WebSphere must retrieve it from either the overflow cache (local caching), or the session store (for persistent sessions). If the session manager must retrieve persistent sessions frequently, the retrievals can impact overall application performance.

WebSphere maintains overflowed local sessions in memory, as discussed in 25.4, “Local sessions” on page 897. Local session management with cache overflow enabled allows an unlimited number of sessions in memory. To limit the cache footprint to the number of entries specified in session manager, use persistent session management, or disable the overflow.

Important: When using local session management without specifying the “Allow overflow” property, a full cache results in the loss of user session objects.

Creating additional application servers

WebSphere also gives the administrator the option of creating additional application servers. Creating additional instances spreads the demand for memory across more JVMs, thus reducing the memory burden on any particular instance. Depending on the memory and CPU capacity of the machines involved, the administrator can add additional instances within the same machine. Alternatively, the administrator can add additional machines to form a hardware cluster, and spread the instances across this cluster.

User distribution within the cluster: When configuring a cluster, session affinity routing provides the most efficient strategy for user distribution within the cluster, even with session persistence enabled. With cluster members, the web server plug-in provides affinity routing among cluster member instances.

Invalidating unneeded sessions

Invalidate a session if the user no longer needs the session object, for example, when the user has logged out of the site. Invalidating a session removes it from the session cache and from the session database. For more information, refer to 25.9, “Invalidating sessions” on page 925.

Increasing available memory

WebSphere allows the administrator to increase an application server’s heap size. By default, WebSphere allocates 256 MB as the maximum heap size. Increasing this value allows the instance to obtain more memory from the system, and thus hold a larger session cache.

A practical limit exists, however, for an instance heap size. The machine memory containing the instance needs to support the heap size requested. Also, if the heap size grows too large, the length of the garbage collection cycle with the JVM might impact overall application performance. This impact has been reduced with the introduction of multi-threaded garbage collection.

Reducing the session timeout interval

By default, each user receives a 30 minute interval between requests before the session manager invalidates the user’s session. Not every site requires a session timeout interval this generous. By reducing this interval to match the requirements of the average site user, the session manager purges the session from the cache and the persistent store, if enabled, more quickly.

Avoid setting this parameter too low and frustrating users. In some cases where the persistent store contains a large number of entries, frequent execution of the timeout scanner reduces overall performance. In cases where the persistent store contains many session entries, avoid setting the session timeout so low that it triggers frequent, expensive scans of the persistent store for timed out sessions. Alternatively, the administrator should consider schedule-based invalidation, where scans for invalid object can be deferred to a time that normally has low demand. Refer to 25.9, “Invalidating sessions” on page 925 for more information.

25.12.2 Reducing persistent store I/O

From a performance point of view, the web developer should keep in mind the following considerations:

- ▶ Optimize the use of the HttpSession within a servlet. Only store the minimum amount of data required in HttpSession. Data that does not have to be recovered after a cluster member fails or is shut down can be best kept elsewhere, such as in a hash table. Recall that HttpSession is intended to be used as a *temporary* store for state information between browser invocations.
- ▶ Specify session=false in the JSP directive for JSP pages that do not need to access the session object.
- ▶ Use time-based write frequency mode to greatly reduce the amount of I/O, because the persistent store updates are deferred up to a configurable number of seconds. Using this mode, all of the outstanding updates for a web application are written periodically based on the configured write interval.
- ▶ Use the Schedule sessions cleanup option. When using the “End of servlet service write frequency” mode, WebSphere does not have to write out the last access time with every HTTP request. WebSphere does not have to synchronize the invalidator thread’s deletion with the HTTP request’s access.

25.12.3 Multirow persistent sessions: Database persistence

When a session contains multiple objects accessed by different servlets or JSP pages in the same web application, multi-row session support provides a mechanism for improving performance. Multi-row session support stores session data in the persistent session database by web application and value.

Table 25-6 shows a simplified representation of a multi-row database table.

Table 25-6 Simplified multi-row session representation

Session ID	Web application	Property	Small value	Large value
DA32242SSGE2	ShoeStore	ShoeStore.First.Name	Alice	
DA32242SSGE2	ShoeStore	ShoeStore.Last.Name	Smith	
DA32242SSGE2	ShoeStore	ShoeStore.Big.String		A big string....

In this example, if the user visits the ShoeStore application and if the servlet involved needs the user’s first name, the servlet retrieves this information through the session API. The session manager brings into the session cache only the value requested. The ShoeStore.Big.String item remains in the persistent session database until the servlet requests it. This method saves time by reducing both the data retrieved and the serialization impact for data the application does not use.

After the session manager retrieves the items from the persistent session database, these items remain in the in-memory session cache. The cache accumulates the values from the persistent session database over time as the various servlets within the web application request them. With WebSphere’s session affinity routing, the user returns to this same cached session instance repeatedly. This reduces the number of reads against the persistent session database, and gives the web application better performance.

How session data is written to the persistent session database is configurable in WebSphere. For information about session updates using single and multi-row session support, refer to 25.8.5, “Single and multi-row schemas (database persistence)” on page 921. Also, refer to 25.8.6, “Contents written to the persistent store using a database” on page 923.

Even with multi-row session support, web applications perform best if the overall contents of the session objects remain small. Large values in session objects require more time to retrieve from the persistent session database, generate more network traffic in transit, and occupy more space in the session cache after retrieval.

Multi-row session support provides a good compromise for web applications that require larger sessions. However, single-row persistent session management remains the best choice for web applications with small session objects. Single-row persistent session management requires less storage in the database and requires fewer database interactions to retrieve a session’s contents. (All of the values in the session are written or read in one operation.) This method keeps the session object’s memory footprint small and reduces the network traffic between WebSphere and the persistent session database.

Tip: Avoid circular references within sessions if using multi-row session support. The multi-row session support does not preserve circular references in retrieved sessions.

25.12.4 Managing your session database connection pool

When using persistent session management, the session manager interacts with the defined database through a WebSphere Application Server data source. Each data source controls a set of database connections known as a connection pool. By default, the data source opens a pool of no more than 10 connections. The maximum pool size represents the number of simultaneous accesses to the persistent session database available to the session manager.

For high-volume websites, the default settings for the persistent session data source might not be sufficient. If the number of concurrent session database accesses exceeds the connection pool size, the data source queues the excess requests until a connection becomes available. Data source queuing can impact the overall performance of the web application (sometimes dramatically).

For best performance, the impact of the connection pool used by the session manager needs to be balanced against the time that a client can spend waiting for an occupied connection to become available for use. By definition, a connection pool is a *shared* resource, so in general the best performance is realized typically with a connection pool that has significantly fewer connections than the number of simultaneous users.

A large connection pool does not necessarily improve application performance. Each connection represents memory impact. A large pool decreases the memory available for WebSphere to execute applications. Also, if database connections are limited because of database licensing issues, the administrator must share a limited number of connections among other web applications requiring database access as well. This is one area where performance tuning tests are required to determine the optimal setting for a given application.

As discussed previously, session affinity routing combined with session caching reduces database read activity for session persistence. Likewise, manual update write frequency, time-based write frequency, and multi-row persistent session management reduce unnecessary writes to the persistent database. Incorporating these techniques can also reduce the size of the connection pool required to support session persistence for a given web application.

Prepared statement caching is a connection pooling mechanism that can be used to further improve session database response times. A cache of previously prepared statements is available on a connection. When a new prepared statement is requested on a connection, the cached prepared statement is returned, if available. This caching reduces the number of costly prepared statements created, which improves response times.

In general, base the prepared statement cache size on the following considerations:

- ▶ The smaller of the number of concurrent users or the connection pool size
- ▶ The number of different prepared statements

With 50 concurrent users, a connection pool size of 10, and each user using two statements, a select and an insert, the prepared statement cache size should be at least $10 \times 2 = 20$ statements. Consult the Information Center at the following website for more details:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=/com.ibm.websphere.nd.multiplatform.doc/info/ae/ae/udat_jdbcdatasorprops.html

25.12.5 Session database tuning

Although the session manager implementation in WebSphere provides for a number of parameters that you can tune to improve performance of applications that use HTTP sessions, maximizing performance requires tuning the underlying session persistence table. WebSphere provides a first step by creating an index for the sessions table when creating the table. The index is composed of the session ID, the property ID for multi-row sessions, and the web application name.

Most database managers provide a great deal of capability in tuning at the table or table space level. However, creating a separate database or instance provides the most flexibility in tuning. Proper tuning of the instance and database can improve performance by 5% or more over improvements that can be achieved by simply tuning the table or table space.

The specifics can vary, depending on the database and operating system. In general, tune and configure the database as appropriate for a database that experiences a great deal of I/O. The database administrator (DBA) should monitor and tune the database buffer pools, database log size, and write frequency. Additionally, maximizing performance requires striping the database or instance across multiple disk drives and disk controllers and using any hardware or operating system buffering that is available to reduce disk contention.

25.13 Stateful session bean failover

Stateful session bean uses the functions of the data replication service and workload management. Each EJB container provides a method for stateful session beans to fail over to other servers. This action enables you to specify whether failover occurs for the stateful session beans at the EJB module level or container level. You can also override the parent object's stateful session bean replication settings from the module level.

25.13.1 Enabling stateful session bean failover

Depending on the requirement, you might not want to enable failover for every single stateful session bean that is installed in the EJB container. You can set or override the EJB container settings at either the application or EJB module level. You can either enable or disable failover at each of these levels.

For example, consider the following situations:

- ▶ You want to enable failover for all applications except for a single application. Enable failover at the EJB container level and override the setting at the application level to disable failover on the single application.
- ▶ You want to enable failover for a single, installed application. Disable failover at the EJB container level and then override the setting at the application level to enable failover on the single application.
- ▶ You want to enable failover for all applications except for a single module of an application. Enable failover at the EJB container level, then override the setting at the module application level to disable failover on the single module.
- ▶ You want to enable failover for a single, installed EJB module. Disable failover at the EJB container level and then override the setting at the EJB module level to enable failover on the single EJB module.

EJB container stateful session bean failover properties

To access stateful session bean failover properties at the EJB container level from the administrative console, complete the following steps:

1. Click **Servers** → **Server Types** → **WebSphere application servers**.
2. Click the application server.
3. In the Container Settings section of the Configuration tab, click **EJB Container Settings** → **EJB container**.
4. In the General Properties section, select the **Enable stateful session bean failover using memory-to-memory replication** option, and click **Apply**.

This option is disabled until you define a replication domain. This selection has a hyperlink to help you configure the replication settings. If no replication domains are configured, the link takes you to a window where you can create one. If at least one domain is configured, the link takes you to a window where you can select the replication settings to be used by the EJB container. Refer to Figure 25-18.

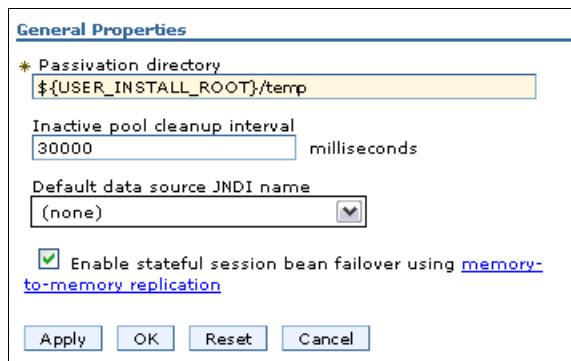


Figure 25-18 Stateful session bean failover settings at the container level

EJB module stateful session bean failover properties

To access stateful session bean failover properties at the EJB module level from the administrative console, complete the following steps:

1. Click **Applications** → **Application Types** → **WebSphere enterprise applications**.
2. Click the application.

3. In the Enterprise Java Bean Properties section of the Configuration tab, select the following options as appropriate for your setup:
 - Enable stateful session bean failover

This option enables stateful session bean failover. If you want to disable the failover, clear this option.
 - Use replication settings from EJB container

If you select this option, any replication settings that are defined for this application are ignored.

Important: If you use this option, then you must configure memory-to-memory replication at the EJB container level. Otherwise, the settings on this window are ignored by the EJB container during server start, and the EJB container logs a message that indicates that stateful session bean failover is not enabled for this application.

- Use application replication settings

If you select this option, you override the EJB container settings. This option is disabled until you define a replication domain. This selection has a hyperlink to help you configure the replication settings. If no replication domains are configured, the link takes you to a window to create one. If at least one domain is configured, the link takes you to a window where you can select the replication settings to be used by the application.
4. Select your choice of replication settings, as shown in Figure 25-19:
 - Use replication settings from EJB container
 - Use application replication settings

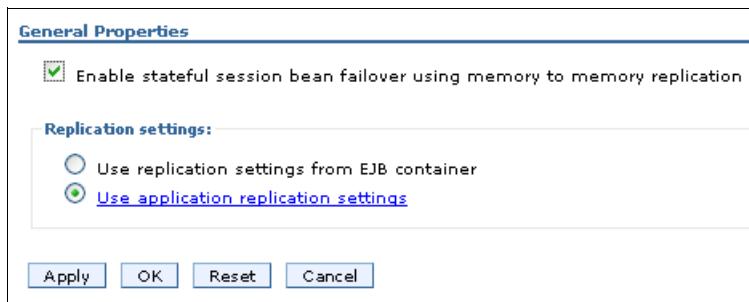


Figure 25-19 Stateful session bean failover settings at the module level

5. Click **OK**.

With Websphere Application Server V8 for Z/OS, you can enable the stateful session bean failover among servants. Failover occurs only between the servants of a given unmanaged server. If an unmanaged z/OS server has only one servant, enabling failover has no effect. An unmanaged z/OS server that has failover enabled does not fail over to another unmanaged z/OS server.

For information about how to enable this feature, consult the “Enabling failover of servants in an unmanaged z/OS server” topic in the Information Center at the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=/com.ibm.websphere.nd.multiplatform.doc/info/ae/ae/tejb_sfsbfzos.html

25.13.2 Stateful session bean failover consideration

In the following sections, we present a few considerations when using the stateful session bean failover feature.

Stateful session bean activation policy with failover enabled

At application assembly, you can specify an activation policy to use for stateful session beans. Remember that the only time that the EJB container prepares for failover by replicating the stateful session bean data using DRS is when the stateful session bean is passivated. If you configure the bean with an activate once policy, the bean is essentially never passivated. If you configure the activate at transaction boundary policy, the bean is passivated whenever the transaction that the bean is enlisted in completes.

For stateful session bean failover to be useful, the “Activate at transaction boundary” policy is required. Rather than forcing you to edit the deployment descriptor of every stateful session bean and reinstall the bean, the EJB container simply ignores the configured activation policy for the bean when you enable failover. The container uses the activate automatically at transaction boundary policy.

Container or bean-managed units of work

The relevant units of work in this case are transactions and activity sections. WebSphere Application Server supports stateful session bean failover for the following components:

- ▶ Container-managed transactions (CMT)
- ▶ Bean-managed transactions (BMT)
- ▶ Container-managed activity sessions (CMAS)
- ▶ Bean-managed activity sessions (BMAS)

In the container-managed cases, preparation for failover occurs only if a request for an enterprise bean method invocation fails to connect to the server. Failover does not take place if the server fails after a request is sent to it and has been acknowledged.

When a failure occurs in the middle of a request or unit of work, WLM cannot safely fail over to another server without some compensation code being executed by the application. When that happens, the application receives a Common Object Request Broker Architecture (CORBA) exception and minor code telling it that transparent failover could not occur because the failure happened during execution of a unit of work.

The application should be written to check for the CORBA exception and minor code, and compensate for the failure. After the compensation code executes, the application can retry the requests and, if a path exists to a backup server, WLM routes the new request to a new primary server for the stateful session bean.

The same is true for bean-managed units of work, transactions, or activity sessions. However, bean-managed work introduces a new possibility that needs to be considered.

For bean-managed units of work, the failover process is not always able to detect that a BMT or BMAS started by a stateful session bean method has not completed. Thus, it is possible that failover to a new server can occur despite the unit of work failing during the middle of a transaction or session.

Because the unit of work is implicitly rolled back, WLM behaves as though it is safe to fail over transparently to another server, when in fact some compensation code might be required. When this situation occurs, the EJB container detects this occurrence on the new server and initiates an exception. This exception occurs under the following scenario:

1. A method of a stateful session bean using bean-managed transaction or activity session calls begin on a UserTransaction that it obtained from the SessionContext. The method does some work in the started unit of work, but does not complete the transaction or session before returning to the caller of the method.
2. During post invocation of the method started in step 1, the EJB container suspends the work started by the method. This action is the action that is required by EJB specification for bean managed units of work when the bean is a stateful session bean.
3. The client starts several other methods on the stateful session bean. Each invocation causes the EJB container to resume the suspended transaction or activity session, to dispatch the method invocation, and then to suspend the work again before returning to the caller.
4. The client calls a method on the stateful session bean that completes the transaction or session started in step 1.

This scenario depicts a *sticky* bean-managed unit of work. The transaction or activity session sticks around for more than a single stateful session bean method. If an application uses a sticky BMT or BMAS, and the server fails after a sticky unit of work completes and before another sticky unit of work starts, failover is successful. However, if the server fails before a sticky transaction or activity session completes, the failover is not successful. Instead, when the failover process routes the stateful session bean request to a new server, the EJB container detects that the failure occurred during an active, sticky transaction or activity session. At that time, the EJB container initiates an exception.

Essentially, failover for both container-managed and bean-managed units of work is not successful if the transaction or activity session is still active. The only real difference is the exception that occurs.

Application design considerations

Consider the following considerations when designing applications that use the stateful session bean failover process:

- ▶ To avoid the server failing with unsuccessful failover described previously, write the application to configure stateful session beans to use CMT rather than BMT.
- ▶ If you want immediate failover and if your application creates either an HTTP session or a stateful session bean that stores a reference to another stateful session bean, the administrator must ensure the HTTP session and stateful session bean are configured to use the same replication domain.
- ▶ Do not use a local and a remote reference to the same stateful session bean.

Normally a stateful session bean instance with a given primary key can only exist on a single server at any given moment in time. Failover might cause the bean to be moved from one server to another, but it never exists on more than one server at a time.

However, there are some unlikely scenarios that can result in the same bean instance, the same primary key, existing on more than one server concurrently. When that happens, each copy of the bean is unaware of the other, and no synchronization occurs between the two instances to ensure they have the same state data. Thus, your application receives unpredictable results.

Important: To avoid the same primary key existing on more than one server concurrently, you must remember that with failover enabled, your application should never get both a local (EJBLocalObject) and remote (EJBObject) reference to the same stateful session bean instance.



Part 6

Maintenance



Managing your environment with the centralized installation manager

WebSphere Application Server V7 introduce the centralized installation manager. In WebSphere Application Server V8, the centralized installation manager evolves with new capabilities and is now integrated with the Installation Manager and the job manager.

The centralized installation manager can reduce the number of steps that are required to install and maintain the WebSphere Application Server environment. New features allow the centralized installation manager to work outside the WebSphere Application Server cell to schedule the installation tasks.

The process for managing Version 7 and previous versions is different than the process for managing Version 8. CIM Version 8 uses the Installation Manager to install the product on remote machines, where Versions 6.1 and 7 uses ISMP and Update Installer. Refer to 1.6, “Centralized installation manager” on page 27, for the processing changes to the centralized installation manager from Version 7 to Version 8.

This chapter includes the following topics:

- ▶ The centralized installation manager prerequisites
- ▶ Planning considerations
- ▶ Working with the centralized installation manager and WebSphere Application Server Version 8
- ▶ Working with the centralized installation manager and WebSphere Application Server V6.1 and V7
- ▶ Managing WebSphere Application Server V8 environment with the centralized installation manager
- ▶ Managing WebSphere Application Server V6.1 and V7 with the centralized installation manager

For additional information about working with the centralized installation manager, refer to the Information Center at the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.installation.nd.doc/info/ae/ae/tins_the_centralized_installation_manager_overview.html

26.1 The centralized installation manager prerequisites

To avoid issues when using the centralized installation manager, ensure that your platform satisfies the requirements that we list in this section.

26.1.1 Linux and UNIX target requirements

The centralized installation manager, through Remote Execution and Access (RXA), uses Secure Shell V2 (SSH2) to access UNIX and Linux target workstations. This use requires at least either OpenSSH V3.6.1 or, if accessing AIX targets, OpenSSH V4.7 or Sun SSH 1.1 on the target hosts.

Using OpenSSH: OpenSSH V4.7.0.5302 for IBM AIX 5L™ V5.3 is not compatible with RXA V2.3. If your target systems are running AIX 5L V5.3 with OpenSSH V4.7.0.5302 installed, the file transfer might stop in the middle of the transfer. To avoid this problem, revert the OpenSSH version from Version 4.7.0.5302 to Version 4.7.0.5301.

Using the SSH protocol

RXA does not supply SSH code for UNIX operating systems. You must ensure that SSH is installed and enabled on any target that you want to access using centralized installation manager.

In AIX and Linux environments, the Bourne shell (`sh`) is used as the target shell. To communicate with Linux and other SSH targets using password authentication, you must edit the `/etc/ssh/sshd_config` file on the targets and set the following property:

`PasswordAuthentication yes`

The default value for the `PasswordAuthentication` property is `no`.

After changing this setting, stop and restart the SSH daemon using either of the following commands sequences:

- ▶ `stopsrc -s sshd`
`startsrc -s sshd`
- ▶ `/etc/init.d/sshd stop`
`/etc/init.d/sshd start`

Installing a secure shell public key to access remote targets

UNIX platforms generally support the use of SSH protocol. To use the SSH public/private key as an authentication method for accessing remote workstations, SSH must be installed and enabled on the installation target system. On AIX and Linux systems, issue the following command to ensure that SSH is enabled on the installation target:

`ps -e | grep sshd`

You can generate an RSA private key and its corresponding public key using the **ssh-keygen** command, as shown in the following example:

```
ssh-keygen -t rsa
```

Take the default location for storing the private key and make note of it. If you specify a non-empty string for the passphrase prompt, make sure that you remember the string, because you will need it when you want to use the generated private key.

Additionally, you must know the location of the SSH public key file on the deployment manager and the administrative ID and password for the installation target. This information is the same administrative ID and password that you use later to install or uninstall software packages on the same installation target.

26.1.2 Windows target requirements

Many RXA operations require access to resources that are not generally accessible by standard user accounts. Therefore, the account names that you use to log in to remote Windows system targets must have administrative privileges.

To ensure that your Windows system targets cooperate with the centralized installation manager, configure the following components:

- ▶ Simple file sharing
- ▶ Firewall
- ▶ Administrative sharing, if the target is Windows Vista 7 or 2008

For more informations about Windows system targets configuration, refer to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.installation.nd.doc/info/ae/ae/cins_cim_rxa_requirements.html

26.1.3 IBM i targets

Use of SSH public/private key authentication is not supported on IBM i platforms.

For information about the IBM i targets pre configuration, refer to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.installation.nd.doc/info/ae/ae/tins_cim_ibmi.html

26.1.4 Additional requirements

Before using the centralized installation manager to install or uninstall maintenance on IBM AIX operating systems as a non-root user, you must install and configure **sudo**, an open-source tool, on the target AIX operating systems.

To install and configure sudo, download **sudo** from the IBM AIX Toolbox for Linux Applications official download website:

<http://www-03.ibm.com/systems/power/software/aix/linux/toolbox/date.html>

After it is downloaded, log on to the target system as root and issue the following command to install **sudo**:

```
rpm -i sudo-*.rpm
```

If your AIX system does not already have **rpm** installed, you can download an AIX **installp** image for the **rpm** package manager for IBM POWER® from the following website:

<http://www-03.ibm.com/systems/power/software/aix/linux/toolbox/download.html>

Authorize a non-root user ID that you specify to run the **slibclean** command as a root user without providing a password. Then, issue the **visudo** command to add the following entry to the /etc/sudoers configuration file (replace *userid* with the real user ID that you will be using):

```
userid ALL = NOPASSWD: /usr/sbin/slibclean
```

Log in with the specified user ID, and then issue the **sudo -l** command. If successful, a message that is similar to the following example displays:

```
User userid may run the following commands on this host:  
(root) NOPASSWD: /usr/sbin/slibclean
```

If you do not have **sudo** installed or if **sudo** is installed but not configured correctly for the specified user ID, error messages display.

26.2 Planning considerations

This section lists planning considerations for WebSphere Application Server.

Note: Ensure that your environment meets the centralized installation manager prerequisites for your platform before you attempt to work with the centralized installation manager. For details, refer to 26.1, “The centralized installation manager prerequisites” on page 942.

26.2.1 WebSphere Application Server V8

For WebSphere Application Server V8, you access the centralized installation manager functions through the job manager or deployment manager. Using the job manager or deployment manager, you can perform the following functions:

- ▶ Install, update, and uninstall Installation Manager on remote machines.
- ▶ Install, update, and uninstall WebSphere Application Server V8 offerings on remote machines.
- ▶ Collect, distribute, and delete files on remote hosts.
- ▶ Manage WebSphere Application Server V8 profiles on remote hosts.
- ▶ Run scripts on remote hosts.
- ▶ Support for z/OS operating system targets.

Important: With the centralized installation manager for Version 8, you cannot install or update Installation Manager on a z/OS target. Prior to working with z/OS targets with the centralized installation manager, ensure that the Installation Manager is already installed.

- ▶ Support to add targets outside the cell.
- ▶ Job scheduling.

When working with WebSphere Application Server V8 and the centralized installation manager, you will need an additional 120 MB of disk space for the Installation Manager installation kit. If your environment includes different operating systems, additional disk space is required for other Installation Manager versions.

26.2.2 WebSphere Application Server V6.1 and V7

For WebSphere Application Server V6.1 and V7, you access the centralized installation manager functions using only the deployment manager. The centralized installation manager for Version 6.1 and Version 7 does not support z/OS operating system targets. Using the centralized installation manager for Version 7, you can perform the following functions:

- ▶ Install, update, and uninstall WebSphere Application Server Network Deployment V7 on remote machines
- ▶ Install and uninstall WebSphere Application Server V6.1 and V7 refresh packs, fix packs, and interim fixes on remote machines
- ▶ Install and uninstall customized installation packages (CIPs)

If you plan to use the centralized installation manager for Version 6.1 or Version 7, the disk space that is required increases significantly and varies depending on how many binaries there will be in the repository.

The centralized installation manager relies on current information regarding the versions of WebSphere Application Server that are installed on each node. This information is kept current on the deployment manager configuration by the node agent that is running on each node. The deployment manager is aware of the correct versions of WebSphere Application Servers that are installed on each node, if the node agent of each node is started at least once after an update is applied. To ensure that the deployment manager receives this information, the centralized installation manager starts the node agent automatically after each installation or uninstallation of maintenance.

The centralized installation manager relies on the node agent to effectively stop the server processes on the target node. If the node agent is not running, it is up to the administrator to ensure that all the server processes are stopped on the target node before initiating any maintenance update operations on the node.

26.3 Working with the centralized installation manager and WebSphere Application Server Version 8

In WebSphere Application Server V8, the centralized installation manager uses the Installation Manager to manage targets. You can access the centralized installation manager using several methods, including the job manager console or the command line. This section provides more information about using aspects of the centralized installation manager with WebSphere Application Server V8 environments.

26.3.1 Installation Manager

Important: WebSphere Application Server V8 uses Installation Manager. Be sure to use an up-to-date version of Installation Manager when working with the centralized installation manager. The minimum required version of the Installation Manager is Version 1.4.3.

Installation Manager is integrated with the centralized installation manager, deployment manager, and job manager to provide a single, central place of administration for the following tasks:

- ▶ Installing and uninstalling products
- ▶ Updating and rolling back fix packs and interim fixes
- ▶ Installing and uninstalling feature packs

Installation Manager provides a common installation technology, which makes it easier and faster to manage your software. Installation Manager can install a desired level of maintenance in single pass, based on GUI commands or response files. One instance of Installation Manager can manage WebSphere and other products family, such as Rational.

Installation Manager also allows you to record response files when invoked with the **-record** and **-skipInstall** options. This method makes it easier to prepare a valid response file for your environment. To learn more about Installation Manager options, refer to 2.2, “Installation Manager installation” on page 35 and the Installation Manager Information Center at the following website:

http://publib.boulder.ibm.com/infocenter/install/v1r4/topic/com.ibm.silentinstall12.doc/topics/t_silent_create_response_files_IM.html

The job manager or deployment manager stores the Installation Manager installation kit in their local directory. By default, the kit will reside at the job manager or deployment manager profile_home/IMKit directory. To change the location of the kit, in the job manager or deployment manager console, click **Jobs** → **Installation Manager installation kits**.

You can add multiple Installation Manager installation kits for many platforms. Figure 26-1 illustrates an example of Installation Manager installation kit V1.4.4 for AIX operating systems, which runs on a POWER platform.

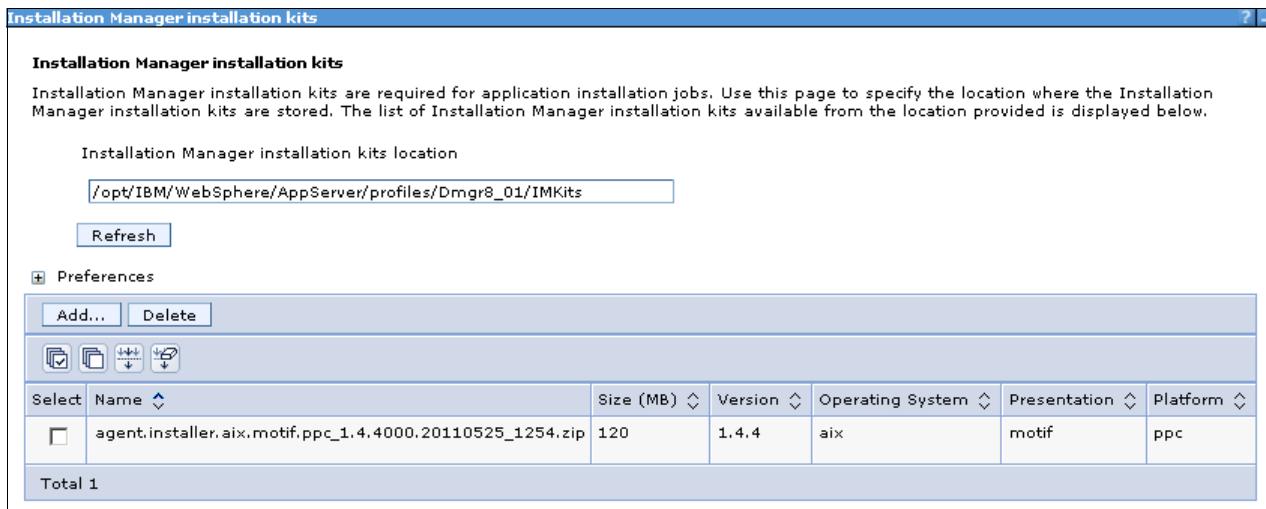


Figure 26-1 Configuration of the Installation Manager installation kit V1.4.4 in the job manager

Note: To install software on your targets using the Installation Manager, you must add a proper version of the Installation Manager installation kit that is valid for your target operating systems.

26.3.2 Accessing the centralized installation manager

You can access the centralized installation manager in WebSphere Application Server V8 from the following interfaces:

- ▶ The job manager console
- ▶ The deployment manager console
- ▶ The **AdminTask** object from **wsadmin**

The console for the centralized installation manager tools is available under the Jobs menu and is accessible from the job manager or deployment manager for WebSphere Application Server V8. Figure 26-2 shows the following menu options in the job manager or deployment manager console:

- ▶ Submit
Prepares and submits jobs to one or more targets.
- ▶ Status
Displays a current status of submitted jobs.
- ▶ Targets
Lists defined targets and allows you to create new targets.
- ▶ Target resources
Lists target resources, which is useful when working with target groups.
- ▶ Target groups
Lists and defines target groups.
- ▶ Installation Manager installation kits
Configures the Installation Manager installation kit repository.



Figure 26-2 Centralized installation manager Jobs menu

You can also work with the centralized installation manager using the command line. You can use the **AdminTask** object by invoking it from the **wsadmin** script from the job manager or the deployment manager host. Example 26-1 presents a Jython example of the **AdminTask** object usage. In this example, a **testConnection** job type invokes the target to verify that the connection between job manager or deployment manager and `yourhost.com` is active.

Example 26-1 Jython AdminTask command used for connection test on WebSphere Application Server V8

```
AdminTask.submitJob('-jobType testConnection -targetList [yourhost.com]' -username admin -password admin)
```

26.4 Working with the centralized installation manager and WebSphere Application Server V6.1 and V7

WebSphere Application Server V8 does not require IBM Update Installer or IBM Installation Factory. However, it supports these products in case you plan to use Version 6.1 or Version 7 together with WebSphere Application Server V8.

The following sections explain additional components that are used when managing Version 6.1 or Version 7 installations using WebSphere Application Server V8.

26.4.1 IBM Update Installer

The centralized installation manager for Version 6.1 and Version 7 installs Update Installer on the target systems. Update Installer installs fix packs and other maintenance on Version 6.1 and Version 7 targets for the central installation manager.

Version note: To avoid issues when using the centralized installation manager, be sure you use Update Installer V7.0.0.15 or later and Installation Factory V7.0.0.15 or later.

If you previously installed Update Installer on any of the target hosts in a directory location other than the *install_root/UpdateInstaller* directory, you might want to consider uninstalling Update Installer with its uninstall process, because it will not be used by the centralized installation manager. You do not need to uninstall the previous copy of Update Installer for centralized installation manager to work properly.

When you install fix packs or other maintenance on the target systems, the centralized installation manager installs Update Installer if the option is selected. If the version of Update Installer located in the *install_root/UpdateInstaller* directory does not meet the minimum version that is required by the interim fix or fix pack, the centralized installation manager automatically installs a newer version on the target, if the newer version has been downloaded to the centralized installation manager repository.

You can use the centralized installation manager to install Update Installer on nodes that are federated to the deployment manager cell.

26.4.2 The centralized installation manager repository structure

The centralized installation manager for WebSphere Application Server V6.1 or V7 works with the additional repository that is illustrated in Figure 26-1 on page 946. This repository consists of directories that contain the installation images for product files, the maintenance files, and the Update Installer files. These directories use the following directory names and file types:

- ▶ UPDI70

This directory holds the *7.0.0.*-WS-UPDI-* .zip* files that contains the Update Installer installation images for the operating systems that you want in your repository. For example, the following file might be copied into this directory:

7.0.0.17-WS-UPDI-AixPPC64.zip

- ▶ WAS70Updates

This directory holds the *.pak files that contain interim fixes for WebSphere Server Network Deployment V7. You can remove these files when they are no longer required. The following file might be in this directory:

7.0.0.1-WS-WAS-IFPK75887.pak

- ▶ WAS70FP*n*

This directory contains the .pak files that make up a specific fix pack for WebSphere Server Network Deployment V7. The following lists shows examples of the files that might be copied into the WAS70FP1 directory. Refer to the WebSphere Application Server V7 support website for the list of files that are required for each fix pack.

For example, for WebSphere Application Server Network Deployment V7 Fix Pack 17, the following .pak files are copied to the WAS70FP17 directory:

7.0.0-WS-WAS-AixPPC64-FP0000017.pak

7.0.0-WS-WASSDK-AixPPC64-FP0000017.pak

- ▶ ND61Updates

This directory holds the .pak files that contain interim fixes for WebSphere Server network Deployment V6.1.

- ▶ ND61FP*n*

This directory contains the .pak files that make up a specific fix pack for WebSphere Application Server V6.1. Refer to the WebSphere Application Server V6.1 support website for the list of files that are required for each fix pack. For example, for WebSphere Application Server Network Deployment V6.1 Fix Pack 37, the following .pak files are copied to the ND61FP37 directory:

6.1.0-WS-WAS-*platform_architecture*-FP0000037.pak

6.1.0-WS-WASSDK-*platform_architecture*-FP0000037.pak

6.1.0-WS-WASWebSvc-*platform_architecture*-FP0000037.pak

6.1.0-WS-WASEJB3-*platform_architecture*-FP0000037.pak

- ▶ WAS70

This directory is created during the installation of the centralized installation manager repository. It contains the product installation files for the operating systems in your environment:

jdk.7000.aix.ppc64.zip

was.nd.7000.aix.ppc64.zip

- ▶ Descriptors

This directory is created when the centralized installation manager is installed and contains the following descriptor files:

InstallPackageND61FP37.xml

InstallPackageND70FP17.xml

26.4.3 Package types

In this section, we describe the packages types that the centralized installation manager supports.

Product installation

The descriptor and binary files are included in product packages and are loaded when the product is loaded into the repository. During the product installation, the following descriptors are included:

- ▶ Maintenance for WebSphere Application Server Network Deployment V6.1 descriptor files that are provided by the product installation
- ▶ Maintenance for WebSphere Application Server Network Deployment V7
- ▶ Update Installer for WebSphere Application Server V7
- ▶ WebSphere Application Server Network Deployment V7

Maintenance tool

This package contains Update Installer, which is the tool used to apply maintenance to a WebSphere Application Server V6.1 or V7 environment. Before using the centralized installation manager to apply maintenance on remote systems, you must download the latest level of Update Installer. You must first install fix packs locally on the deployment manager system using Update Installer.

Refresh and fix packs

With this package type, you can download binary files based on a specific platform. When a refresh or fix pack for IBM WebSphere Application Server is released, it usually comes with a fix pack for Java SDK. The centralized installation manager requires both fix packs in the repository and installs both fix packs to the selected targets.

Interim fixes

This package type is used to apply a small update to the WebSphere Application Server run time, usual provided by WebSphere support.

Refer to 26.6, “Managing WebSphere Application Server V6.1 and V7 with the centralized installation manager” on page 976 for more information about adding additional packages or registering a repository.

26.4.4 Accessing the central installation manager

You can manage WebSphere Application Server V6.1 or V7 from the centralized installation manager in WebSphere Application Server V8 using the following interfaces:

- ▶ The deployment manager console
- ▶ The **AdminTask** object from **wsadmin**

In the deployment manager console, the centralized installation manager jobs are also available from the Jobs menu, but they are reserved only for WebSphere Application Server V8. Additional tools for WebSphere Application Server V6.1 and V7 are available by clicking **System administration** → **Centralized Installation Manager**, as shown in Figure 26-3 on page 951. Shown are the two centralized installation manager sections available from the deployment manager V8 console. The marked section is only for managing the WebSphere Application Server V6.1 and V7.

Available Installations Installs installation packages to the targets.

Installation Packages Displays all available descriptors and packages in the centralized installation manager repository.

Installations in Progress Displays the status of installations that are in progress.

Installation History	Displays the history log for installations done with the centralized installation manager.
Installation Targets	Lists and defines installation targets for the centralized installation manager.



Figure 26-3 Centralized installation manager sections from the deployment manager V8 console

To work with centralized installation manager through the command line, you can use the **AdminTask** object by invoking it from the **wsadmin** script from the deployment manager host. Example 26-2 presents a Jython example of **AdminTask** object usage. In this example, a connection test is made with target host `yourhost.com` from deployment manager to check if the connection is active.

Example 26-2 Jython AdminTask command used for connection test on WebSphere Application Server V7

```
AdminTask.testConnectionToHost ('[-hostName yourhost.com  
-platformType linux -adminName root -adminPassword password]')
```

Note: The AdminTask object uses different methods and parameters when working with the centralized installation manager for WebSphere Application Server V8 than for Version 6.1 or Version 7. For more information about the available methods, refer to 26.6.11, “The centralized installation manager AdminTask commands” on page 993.

To learn more about using the centralized installation manager, refer to the Chapter 26, “Managing your environment with the centralized installation manager” on page 941.

26.5 Managing WebSphere Application Server V8 environment with the centralized installation manager

Managing the full life cycle of WebSphere Application Server V8 environment using the centralized installation manager consists of the following steps:

1. Define your targets.
2. Install Installation Managers on targets.
3. Install the product.
4. Create profiles.
5. Register with job manager.
6. Work with environment.

Note: Not every step is required when working with the centralized installation manager. It depends on your WebSphere Application Server environment and if you work with existing installations or will create a new environment.

26.5.1 Adding new targets

To register a new target, start the job manager or deployment manager and the targets. In the web console, click **Jobs** → **Targets** → **New Host**.

Supply the form presented on Figure 26-4 on page 953 with the host name of your target and its operation type. Specify the user that will be used for product installation and management or select to use a public/private key pair to authenticate to the target. If you select a non-root user, be sure that this account will allow you to install the product. Refer to 26.1.4, “Additional requirements” on page 943 for more information about this topic.

You can also select the **Save security information** check box to save the credentials together with the host. This action frees you from supplying the credentials each time you submit a job to that host.

In the “Installation Manager data location path(s)” field, you can also supply paths on the target host to the Installation Manager installation directory. If there is no Installation Manager on that host, leave this field blank. If the Installation Manager was not installed in the standard location (during installation, a custom path was given), then you should use it in this field. This action helps the centralized installation manager in obtaining which Installation Manager it should use on the target, because the inventory job that the centralized installation manager uses to detect the installed WebSphere software, might not be able to locate the Installation Manager product.

Click **OK** to save the target.

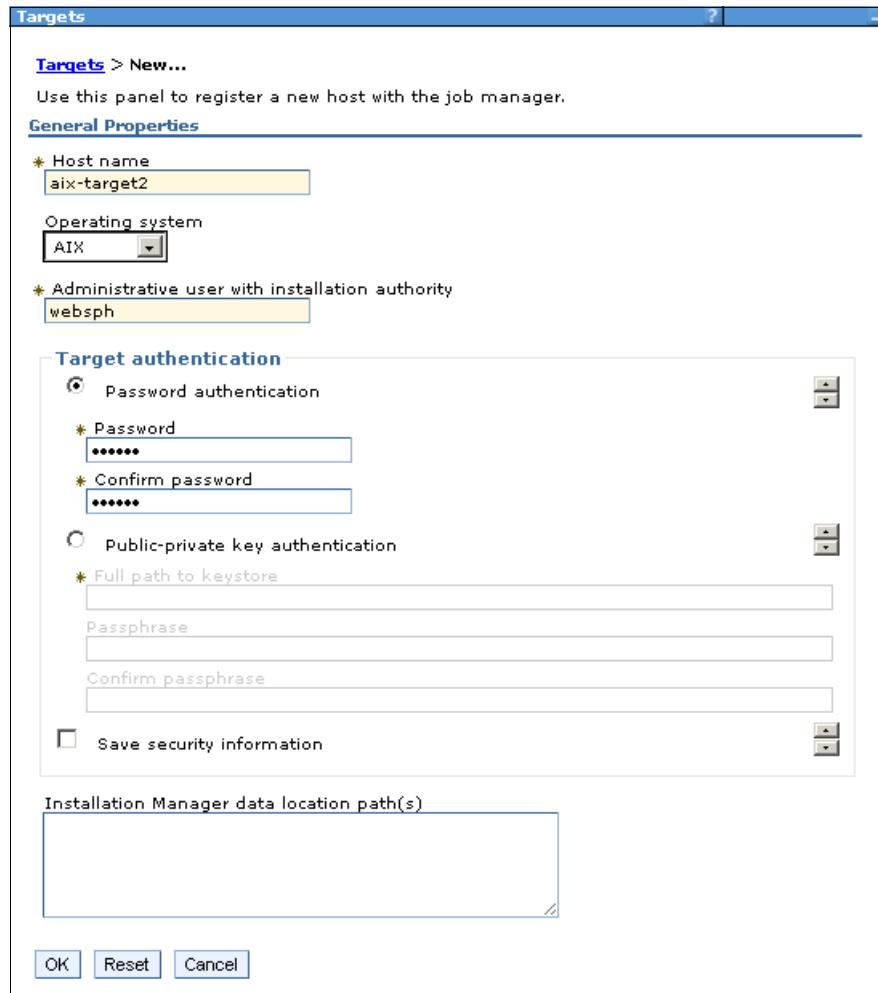


Figure 26-4 New target definition

Complete the following steps to list the available resources discovered by the centralized installation manager on the target:

1. Click **Jobs** → **Targets**.
2. Select the check boxes next to your targets and click the **Display Resources** button.

3. In the pop-up menu, select **All** to see all available resources. See Figure 26-5 for more details. Note that not all targets contain the version information. Targets, which have versions, are managed nodes registered in the job manager, such as deployment manager or administrative agent servers, and the version is their product version level. Targets, which are just added as new targets (as in the step above), will not contain this information.

Targets	
Targets	
Use this panel to find targets for jobs. Either select a saved search, create a new search, or find targets by name.	
+ Find	
<input type="button" value="New Host..."/>	<input type="button" value="Display Resources"/> All
<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	Application
<input type="checkbox"/>	Server
Select	Installation Manager
<input type="checkbox"/>	AAMan
<input type="checkbox"/>	Package
<input checked="" type="checkbox"/>	Profile
<input checked="" type="checkbox"/>	WAS8N Package Group
	Version ▽
	ND 8.0.0.0
	ND 8.0.0.0
	aix-target2
<input checked="" type="checkbox"/>	alias_v8dmqr
	ND 8.0.0.0
	saw116-sys2
Total	5

Figure 26-5 Listing resources on targets

Discovered resources are displayed (Figure 26-6 on page 955). If your target does not contain any WebSphere Application Server or Installation Manager products, the list will be empty. You can also filter the resources you want to see by selecting a different **Display Resources** option, for example, limited only to Applications.

Targets		
Targets > Target resources		
Target resources are applications, servers, and clusters that can be used in jobs submitted to the job manager. Set the find parameters to limit the search results for target resources. Use the target resources panels to explore the resources that are available.		
<input type="button" value="Find"/> <input type="button" value="Preferences"/>		
Resources	Quantity	Target name
InstallationManager/InstallationManager	1	saw116-sys1
InstallationManager/InstallationManager/PackageGroup/IBM WebSphere Application Server Network Deployment V8.0	1	saw116-sys1
InstallationManager/InstallationManager/PackageGroup/IBM WebSphere Application Server Network Deployment V8.0/Package/IBM WebSphere Application Server Network Deployment	1	saw116-sys1
InstallationManager/InstallationManager/PackageGroup/IBM WebSphere Application Server Network Deployment V8.0/Package/IBM WebSphere Application Server Network Deployment/Profile/AppSrv8_01	1	saw116-sys1
InstallationManager/InstallationManager/PackageGroup/IBM WebSphere Application Server Network Deployment V8.0/Package/IBM WebSphere Application Server Network Deployment/Profile/Dmqr8_01	1	saw116-sys1
InstallationManager/InstallationManager/PackageGroup/IBM WebSphere Application Server Network Deployment V8.0/Package/IBM WebSphere Application Server Network Deployment/Profile/V8JobMar01	1	saw116-sys1
Total 6		

Figure 26-6 Resources discovered by the centralized installation manager

Note that the target resources include information such as the package or package group that was used to install a given product with the Installation Manager. In Figure 26-6, the package group *IBM WebSphere Application Server Network Deployment V8.0* was used as the base for each WebSphere Application Server resource. You can also see that there are three profiles resources derived from the *IBM WebSphere Application Server Network Deployment* package.

26.5.2 Installing Installation Manager on remote targets

With defined targets, you are able to install the Installation Manager on that target. Complete the following steps to prepare your Installation Manager installation kit to use it on your targets.

Tip: WebSphere Application Server V8 comes with Installation Manager V1.4.3. You can use this version to work with the centralized installation manager, but consider using the newest Installation Manager version available. To download a current version of IBM Installation Manager, go to the following website:

http://www-947.ibm.com/support/entry/portal/All_download_links/Software/Rational/IBM_Installation_Manager

1. Download the compressed Installation Manager copy valid for your target operating system to your local machine.
2. Click **Jobs** → **Installation Manager installation kits**.

3. Optionally, if you want to change the default local directory where the job manager or deployment manager will keep binaries, enter the path under **Installation Manager installation kits location**. By default, it is the `profile_home/IMKits` directory, where `profile_home` is your job manager or deployment manager profile directory.
4. Click the **Add** button and point to the Installation Manager compressed file on your local machine. The Installation Manager installation kit will be transferred to the job manager or deployment manager and registered in its repository.

Tip: If you do not want to transfer the Installation Manager installation kit using your local computer and HTTP transfer, you can also copy this file directly to the Installation Manager installation kit location directory on the deployment manager or job manager.

After you complete these steps, you should see a view similar to Figure 26-7. On this window, Installation Manager for AIX systems, running on a POWER platform, is used.

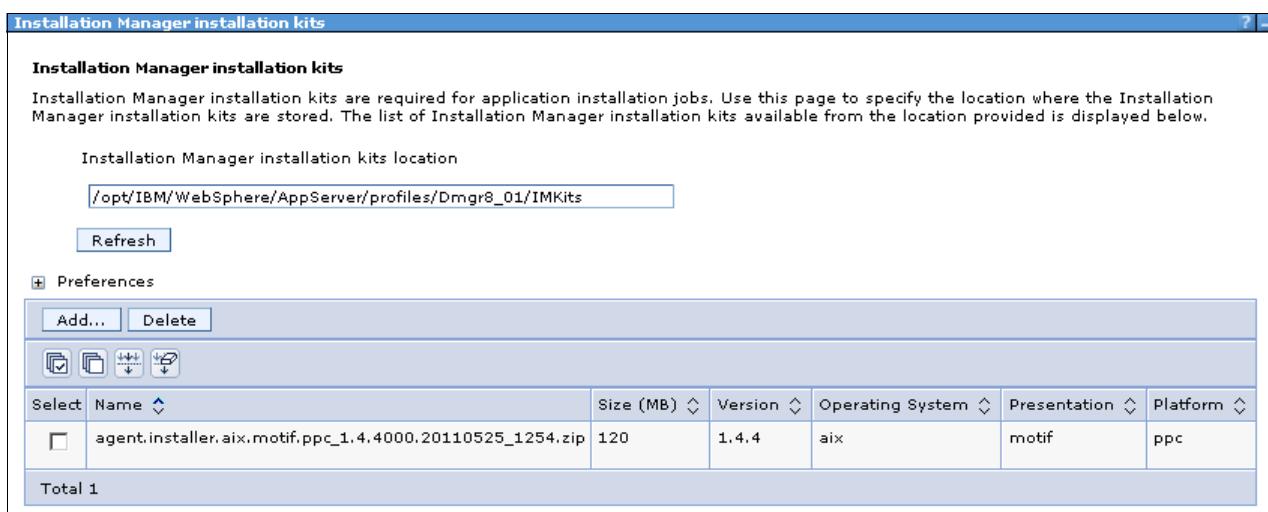


Figure 26-7 Installation Manager installation kit definition

To delete the Installation Manager installation kit, select it from the list shown in Figure 26-7 and click the **Delete** button. The compressed file with the Installation Manager will also be deleted from the server.

You can install additional versions of Installation Manager for other platforms of your choice that are supported by the Installation Manager. Each of them will be kept in the Installation Manager installation kit directory.

Note: The centralized installation manager repository location is configured in the `CIMJMMetadata.xml` file. This file is located in the job manager or deployment manager `profile_home/properties/cimjm` directory.

Complete the following steps to proceed with the installation of Installation Manager on remote targets:

1. Click **Jobs** → **Submit** from the Jobs navigation section.
2. From the drop-down menu, choose the **Install IBM Installation Manager** job and click **Next**.
3. Choose the job targets.

- a. Select a group of targets from the list, or select **Target names**.
- b. If you selected **Target names**, specify a target name and click **Add**, or click **Find** and specify the chosen targets on the Find targets window.
- c. If user authentication is required, specify a user name, password, or any other authentication values as needed (Figure 26-8).

Submit a job to the job manager

Choose the targets to use. Targets can be either a target group or a list of one or more target names. If security is enabled on a target, then the saved authentication information for the target is used. Otherwise, you can optionally provide authentication parameters. If a user name is not provided, then the current console user ID is used. When authentication parameters are entered, they are used for all targets and take precedence over the saved authentication associated with the host.

Step 1: Choose a job type

→ Step 2: Choose job targets

Step 3: Specify job parameters

Step 4: Schedule the job

Step 5: Review the summary and submit the job

Choose job targets

Job type: Manage offerings

Target groups
-- No groups --

Target names
 Add
aix-target2

Target authentication

User name

Password authentication
* Password
* Confirm password

Public-private key authentication
* Full path to keystore
Passphrase
Confirm passphrase

Previous **Next** **Cancel**

Figure 26-8 Configuring job authentication method for target

- d. Click **Next**.

- Because you specified the Installation Manager installation kit in the previous step, you can leave the “The path and file name of Installation Manager kit” field empty, as shown in Figure 26-9. Configuring the Installation Manager installation kit before submitting the job is also useful, because when you install the Installation Manager on group of targets (consisting of even different platforms), the Installation Manager automatically uses an appropriate Installation Manager installation kit for each target (if it has a valid copy for the given platform), so providing a static directory will only work for single targets.

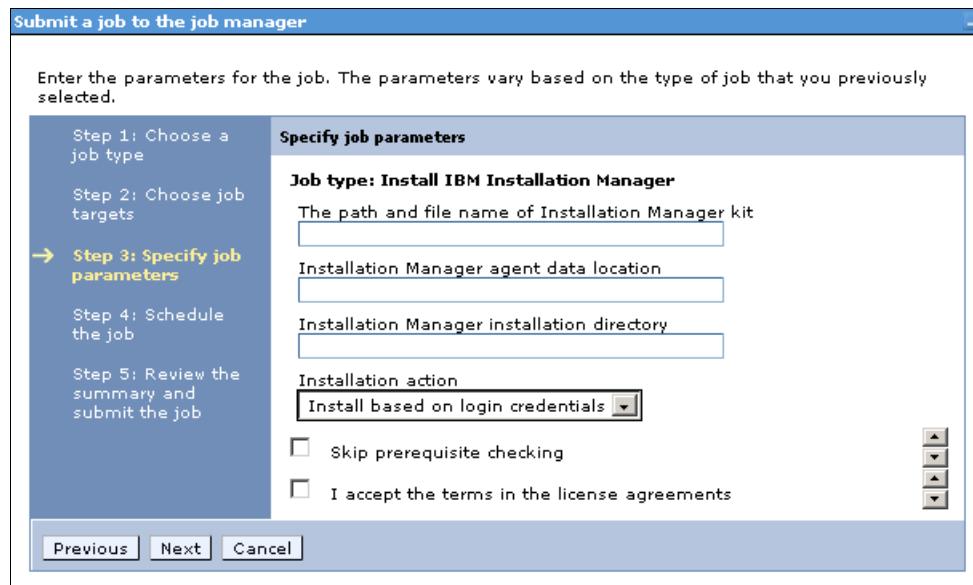


Figure 26-9 Installing the Installation Manager on the target

You can also leave the Installation Manager agent data location and Installation Manager installation directory empty if you want to use the default Installation Manager install directory.

To proceed to the next step, accept the terms in the license agreements by selecting **I accept the terms in the license agreements**. To review the license, extract the Installation Manager installation kit and from that location run the `install.exe` command for Windows systems or `install` command for AIX, Linux or Solaris operating systems. You can also run `installc -c` to review the license in text mode.

Click **Next** to continue.

- If you want to schedule this job to run at a specified time, you can use this step to define specific dates and time. You can also specify the job to run on a given basis, such as daily or weekly. To just install the product, leave the form with its defaults with Availability/interval selected to **Run once** and click **Next**.

Figure 26-10 shows the available configuration options of this step.

Submit a job to the job manager

Schedule the job by specifying when the job available, when the job expires, if the job is to rerun after a period of time, and what email address is to receive notification when the job is done. Jobs are scheduled for availability and expiration relative to the time of the machine on which the job manager resides.

Step 1: Choose a job type
Step 2: Choose job targets
Step 3: Specify job parameters
→ Step 4: Schedule the job
Step 5: Review the summary and submit the job

Schedule the job

Job type: Manage offerings
Notification
Email addresses

Initial Availability
Specify when this job is first available.
 Make the job available now.
 Schedule availability
Date (MM/dd/yyyy) / / Time (HH:mm:ss) : :

Expiration
Specify when this job is no longer available.
 Use default expiration - 1 days.
 Expire the job based on a date
Date (MM/dd/yyyy) / / Time (HH:mm:ss) : :
 Expire the job based on a duration
Expire after minutes

Job Availability Interval
Jobs can run repeatedly based on an interval. Specify the interval that the job is available.
Availability interval

Previous **Next** **Cancel**

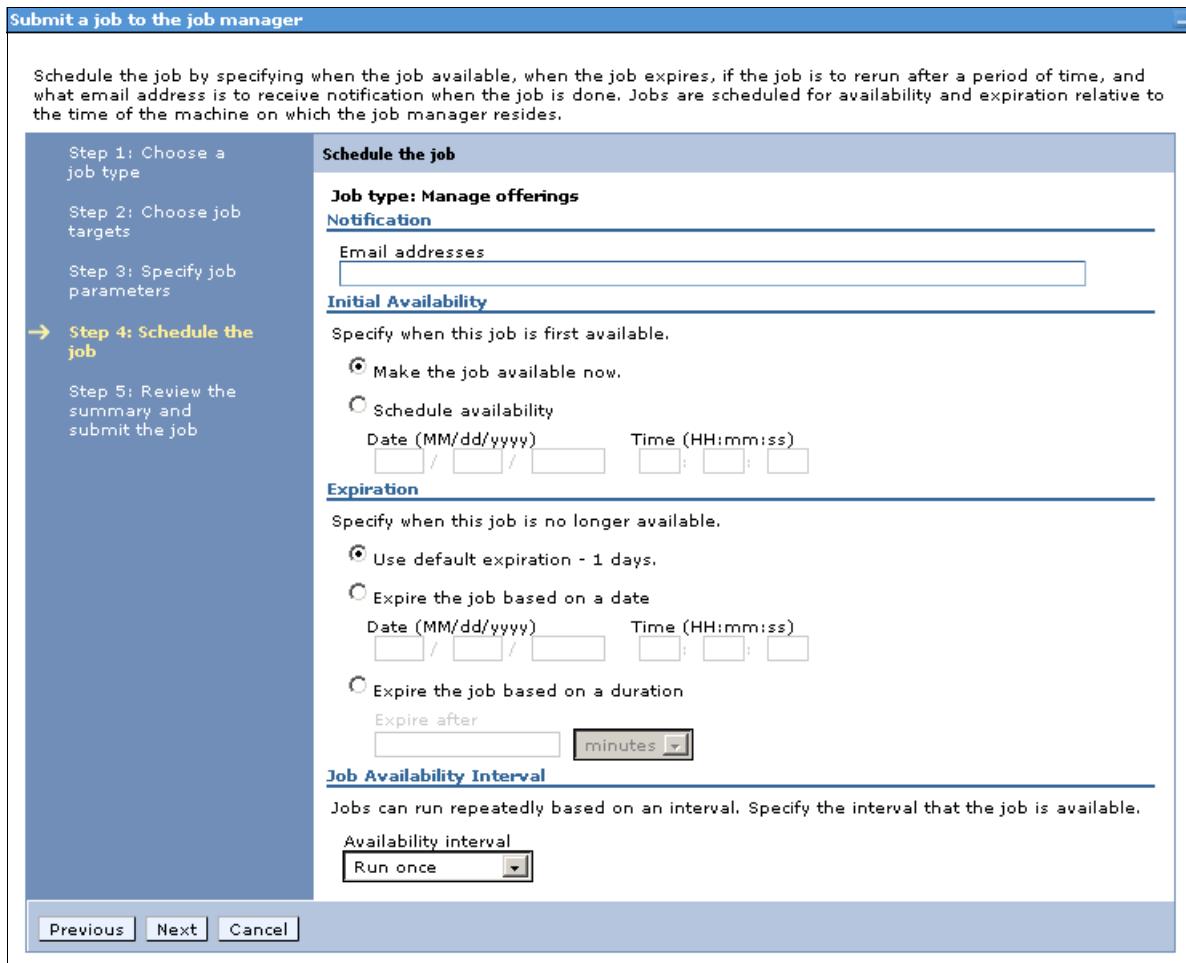


Figure 26-10 Scheduling a job

6. Review the summary and click **Finish** to submit the job.

After you submit the job, a unique identifier will be allocated to it to track its status. The administrator can always return to the **Jobs** → **Status** view to track the job progress (Figure 26-11).

The screenshot shows the 'Job status' interface. At the top, there's a status summary key with four categories: Succeeded (green), Partially succeeded (yellow), Failed (red), and Incomplete (light blue). Below this are 'Find' and 'Preferences' buttons. A toolbar with icons for Suspend, Resume, Delete, and other actions is present. The main area is a table with columns: Select, Job ID, Description, State, Activation Time, Expiration Time, and Status Summary. The table contains four rows of job data:

Select	Job ID	Description	State	Activation Time	Expiration Time	Status Summary
<input type="checkbox"/>	130998509206961514	installSSHPublicKey	Active	07/06/2011 14:44:52	07/07/2011 14:44:52	<div style="width: 100%;">1</div>
<input type="checkbox"/>	130998541017261519	testConnection	Active	07/06/2011 14:50:10	07/07/2011 14:50:10	<div style="width: 10%;">1</div>
<input type="checkbox"/>	130998552445661524	testConnection	Active	07/06/2011 14:52:04	07/07/2011 14:52:04	<div style="width: 100%;">1</div>
<input type="checkbox"/>	130998560640761529	testConnection	Active	07/06/2011 14:53:26	07/07/2011 14:53:26	<div style="width: 10%;">1</div>

Total 4

Figure 26-11 Monitoring jobs status

If the job is successful or it fails, additional messages and files might be shown (Figure 26-12 on page 961). If files are present, you can click them to see the full path to the file on the target. This path can be used to locate it and obtain it using a Collect file job. In case of an error, investigate the error message or file, correct the condition that caused the error, and submit the job again.

The screenshot shows the 'Job status' interface with the following details:

- Job ID:** 130886432165293312
- Description:** manageOfferings
- Activation Time:** 06/23/2011 15:25:21
- Expiration Time:** 06/24/2011 15:25:21
- Target Names:** saw116-sys2
- Status:** Failed
- Output Files:** stdErr.txt, stdOut.txt, logFilename.txt

A 'Back' button is visible at the bottom left.

Figure 26-12 Additional jobs messages located in files accessible from the console

Updating the Installation Manager on remote targets

To update the Installation Manager, select the **Update IBM Installation Manager** job.

For additional information about this procedure, refer to the Information Center at the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.installation.nd.doc/info/ae/ae/tagt_jobmgr_update_im.html

Uninstalling the Installation Manager on remote targets

Select **Uninstall IBM Installation Manager** to proceed with this procedure. Note that to uninstall the Installation Manager, all products installed using it must be uninstalled first. For additional information about this topic, refer to the Information Center at the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.installation.nd.doc/info/ae/ae/tagt_jobmgr_uninstall_im.html

26.5.3 Installing a Secure Shell (SSH) public key

Installing a secure shell public key is not a required step, but can be done using the job manager or the deployment manager console.

Complete the following steps to install a secure shell public key:

1. Click **Jobs** → **Submit** from the navigation tree of the administrative console.
2. Select the **Install SSH Public Key** job and click **Next**.
3. Select the job targets and supply the administrative user name and password. Click **Next**.

4. On the Specify the job parameters window, specify the location of the public key file that you want to install on the selected target and click **Next** (Figure 26-13).

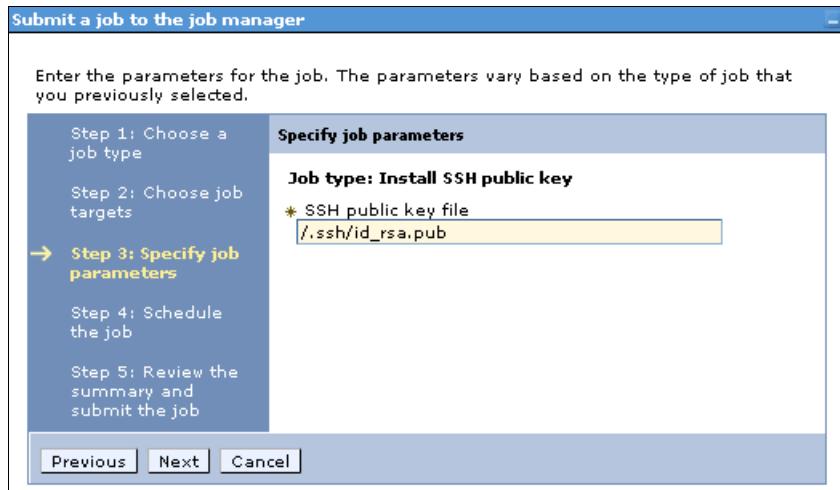


Figure 26-13 Installing a public key on target

To learn more about how to obtain the keys, refer to 26.1, “The centralized installation manager prerequisites” on page 942.

5. Schedule the job and click **Next**.
6. Review the summary and click **Finish** to submit the job.

If the job is successful, you will be able to use public/private key pair method to authenticate from the job manager or network manager host to the target host.

26.5.4 Installing WebSphere Application Server binaries on remote host

With the targets defined and the Installation Manager installed on the targets, complete the following steps to install the WebSphere Application Server binaries on remote hosts:

1. Consider running an inventory job to refresh the job manager or deployment manager targets repository:
 - In the administrative console, click **Job** → **Submit**.
 - In the Job type menu list, select the **Inventory** job and click **Next**.
 - Specify the target names and target authentication and click **Next**.
 - Schedule and submit the job.
2. Use the **Manage offerings** job to install the product:
 - In the administrative console, click **Job** → **Submit**.
 - In the Job type menu list, select the **Manage offerings** job and click **Next**.
 - Specify the target names and target authentication and click **Next**.
 - Specify the required parameter:

Response file path name The full path name to the Installation Manager response file. The path must point to a file located on job manager or deployment manager machine.

Tip: You can obtain the Installation Manager response file using a **IBMIM.exe** command for Windows or **IBMIM** command for UNIX, executed from `installation_manager_home/eclipse` directory:

```
IBMIM -record <path_to_your_file>.xml -skipInstall  
<path_to_empty_directory>
```

Example 26-3 shows a sample response file that could be used for WebSphere Application Server V8 product installation. Note that you will need to edit the highlighted properties to tell Installation Manager which repository it should use and where it should install the product on the target. There are many more ways to edit this file to instruct Installation Manager. For more information about this topic, refer to the Information Center at the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.nd.doc/info/ae/ae/tagt_job_install_was.html

Example 26-3 Response file used by Installation Manager to install WebSphere Application Server Network Deployment V8 product

```
<?xml version="1.0" encoding="UTF-8"?>  
<!--The "acceptLicense" attribute has been deprecated. Use "-acceptLicense"  
command line option to accept license agreements.-->  
<agent-input acceptLicense='true'>  
<server>  
<repository location='<MY REPOSITORY LOCATION>' />  
</server>  
<profile id='IBM WebSphere Application Server Network Deployment V8.0'  
installLocation='<LOCATION TO INSTALL PRODUCT ON TARGET MACHINE>'>  
<data key='eclipseLocation' value='<LOCATION TO INSTALL PRODUCT ON TARGET  
MACHINE>' />  
<data key='user.import.profile' value='false' />  
<data key='cic.selector.os' value='aix' />  
<data key='cic.selector.ws' value='motif' />  
<data key='cic.selector.arch' value='ppc' />  
<data key='cic.selector.nl' value='en' />  
</profile>  
<install modify='false'>  
<offering id='com.ibm.websphere.ND.v80' version='8.0.0.20110503_0200'  
profile='IBM WebSphere Application Server Network Deployment V8.0'  
features='core.feature,ejbdeploy,thinclient,embeddablecontainer,com.ibm.sdk.  
6_64bit,samples' installFixes='none' />  
</install>  
<preference name='com.ibm.cic.common.core.preferences.eclipseCache'  
value='/opt/IBM/IMShared' />  
<preference name='com.ibm.cic.common.core.preferences.connectTimeout'  
value='30' />  
<preference name='com.ibm.cic.common.core.preferences.readTimeout'  
value='45' />  
<preference  
name='com.ibm.cic.common.core.preferences.downloadAutoRetryCount'  
value='0' />  
<preference name='offering.service.repositories.areUsed' value='true' />  
<preference name='com.ibm.cic.common.core.preferences.ssl.nonsecureMode'  
value='false' />
```

```

<preference
name='com.ibm.cic.common.core.preferences.http.disablePreemptiveAuthenticati
on' value='false'/>
<preference name='http.ntlm.auth.kind' value='NTLM' />
<preference name='http.ntlm.auth.enableIntegrated.win32' value='true' />
<preference
name='com.ibm.cic.common.core.preferences.preserveDownloadedArtifacts'
value='true' />
<preference name='com.ibm.cic.common.core.preferences.keepFetchedFiles'
value='false' />
<preference name='PassportAdvantageEnabled' value='false' />
<preference name='com.ibm.cic.common.core.preferences.searchForUpdates'
value='false' />
<preference name='com.ibm.cic.agent.ui.displayInternalVersion'
value='false' />
</agent-input>

```

- e. Specify optional parameters:

IBM Installation Manager Path

Specify the path to install Installation Manager on the remote machine. If this parameter is blank, then Installation Manager is considered to be installed in the default location.

IBM Installation Manager agent data location

Specify an IBM Installation Manager data location that is not the default location for the manageOfferings job.

IBM Installation Manager key ring file

If the package repository requires a key ring file for authentication, specify the full path name of the key ring file on the job manager machine.

Key ring file password

If the key ring file is password protected, specify the key ring password.

Tip: To avoid trouble, if you installed the Installation Manager using defaults, you should also use defaults during this task. Do not use a non-default data location unless you are familiar with IBM Installation Manager.

- f. Select **I accept the terms in the license agreements** and click **Next**.
- g. Schedule the job or click **Next** to proceed with the next step.
- h. Review the summary of the job and click **Finish** to submit it.

At this point, your job is submitted and has a unique identifier. You can track its status by clicking **Jobs → Status**.

To learn more about installation of the WebSphere Application Server product using jobs, refer to the Information Center at the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.installation.nd.doc/info/ae/ae/tagt_job_install_was_gui.html

26.5.5 Creating a WebSphere Application Server profile on a remote target

In this example, two profiles are created: a network deployment profile and a default profile, which will be federated into the network manager cell.

To create profiles with centralized installation manager, you need response files. Consider preparing your own response files on your local, experimental environment. Two response files that will be used in this section are shown in Example 26-4 and in Example 26-5.

Example 26-4 Response file for a Deployment Manager cell

```
create
profileName=Dmgr8_01
profilePath=/opt/IBM/WebSphere/AppServer/profiles/Dmgr8_01
templatePath=/opt/IBM/WebSphere/AppServer/profileTemplates/cell/dmgr
nodeName=aix-target2CellManager01
cellName=aix-target2Cell01
hostName=aix-target2
adminUserName=admin
adminPassword=admin
enableAdminSecurity=true
samplesPassword=admin
appServerNodeName=aix-target2Node01
nodeProfilePath=/opt/IBM/WebSphere/AppServer/profiles/AppSrv8_01
```

Example 26-5 Response file for a default server profile federated to the deployment manager cell

```
create
profileName=AppSrv8_01
profilePath=/opt/IBM/WebSphere/AppServer/profiles/AppSrv8_01
templatePath=/opt/IBM/WebSphere/AppServer/profileTemplates/cell/default
nodeName=aix-target2CellManager01
cellName=aix-target2Cell01
hostName=aix-target2
enableAdminSecurity=true
adminUserName=admin
adminPassword=admin
samplesPassword=admin
appServerNodeName=aix-target2Node01
dmgrProfilePath=/opt/IBM/WebSphere/AppServer/profiles/Dmgr8_01
nodePortsFile=/opt/IBM/WebSphere/AppServer/profiles/Dmgr8_01/properties/nodeportdef.props
portsFile=/opt/IBM/WebSphere/AppServer/profiles/Dmgr8_01/properties/portdef.props
```

Complete the following steps to use the centralized installation manager to create those profiles.

1. Click **Jobs** → **Submit**.
2. From the drop-down menu, select **Manage profiles**.
3. Select your targets and authentication method and click **Next**.
4. Specify the installation home directory of WebSphere Application Server product. You should specify the path from Example 26-3 on page 963, where you defined it during base product installation.

Specify the path to the response file that will be used to create the profile. In this case, copy the response file from Example 26-4 on page 965 on the job manager or deployment manager machine and type the path to that file (Figure 26-14). Click **Next**.

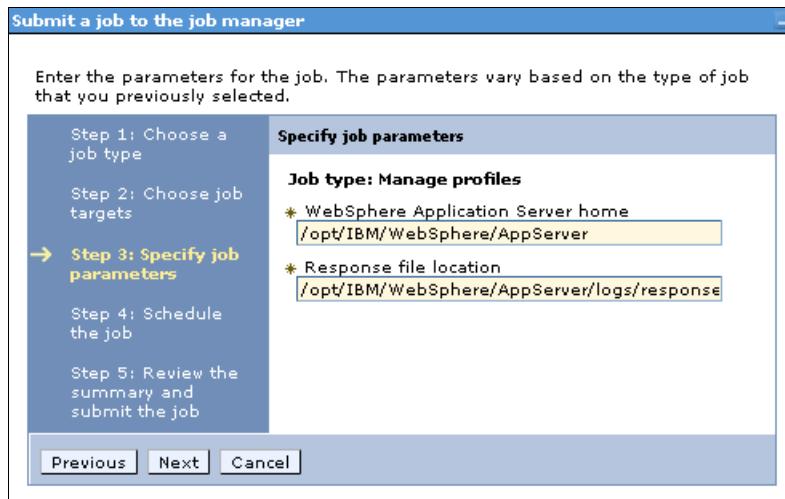


Figure 26-14 Creation of profile using the Manage profiles job

5. Schedule the job or submit it to run once and click **Next**.
6. Review your job details and submit it by clicking **Finish** or change its parameters by clicking **Previous**.

If the job completes successfully, proceed with the creation of the next profile. If any errors occur during installation, inspect them, correct them, and run the manage profiles job again.

To create a second profile, complete steps 1 on page 965 to 6 again, but in step 4, enter the path to the response file shown in Example 26-5 on page 965.

When the second job has completed successfully, the profiles will be ready, but they will be stopped. To use them, the administrator has to start the deployment manager and the node agent of the server profile. You can either log into the targets and invoke the `./startManager` and `./startNode` commands or use the job manager or deployment manager.

Complete the following steps to use the job manager or deployment manager console to start the deployment manager and node agent:

1. Click **Jobs** → **Submit**.
2. Select **Run command job on remote host** from the drop-down menu.
3. Add your target on which you installed the profiles.
4. Specify the user name that you used to install the product, supply the password or use the public/private key authentication method, and click **Next**.
5. Specify the script name you want to run in the Command or script field, and run `startManager.sh` to start the deployment manager or `startNode.sh` to start the node agent.

In the field under Working directory, specify the path to your command. If you want to start the deployment manager server, use:

```
/opt/IBM/WebSphere/AppServer/profiles/Dmgr8_01/bin
```

Or, you can use your own directory where you installed the deployment manager profiles. If you want to start the server profile, use:

/opt/IBM/WebSphere/AppServer/profiles/AppSrv8_01/bin

Or, you can use your own directory where you installed the federated server profile (Figure 26-15).

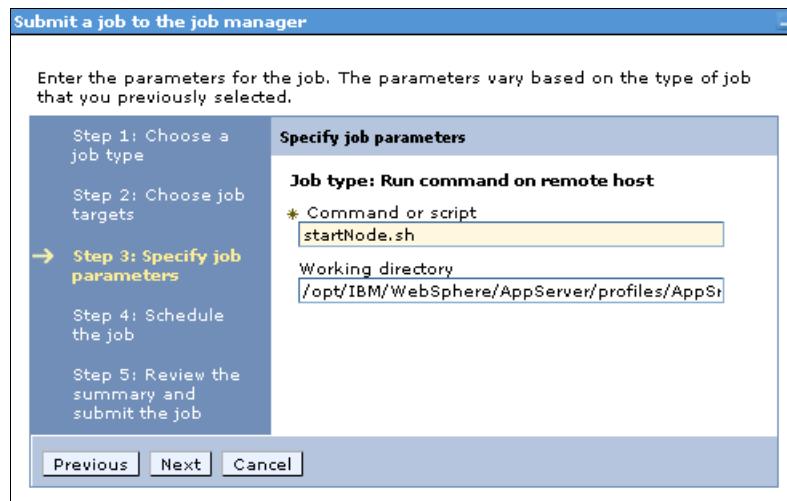


Figure 26-15 Running a remote script using job manager

6. Schedule your job and click **Next**.
7. Review the job details and submit it by clicking **OK**.

26.5.6 Registering and unregistering the profile in the Job Manager console

To fully control target WebSphere Application Server profiles, you have to register the newly created deployment manager.

Note: Apart from deployment manager servers, you can also register administrative agent type servers to manage even more complex WebSphere Application Server environments.

Complete the following steps to register deployment manager:

1. In the target deployment manager administrative console, click **System administration** → **Deployment manager**.
2. Click the **Job managers** link under the Additional Properties section.
3. Click the **Register with Job Manager** button.

- Supply the managed node name for your job manager profile (Figure 26-16).
Specify the job manager host name, secure http port, and the alias name that will be used in the job manager. Specify the user name and password of the job manager user.

The screenshot shows a 'Deployment manager' dialog box with the title 'Deployment manager > Job managers > Register with Job Manager'. The main content area contains a detailed description of the registration process. Below this, a 'General Properties' section is displayed with the following fields:

* Managed node name	saix-target2CellManager01
Alias	WAS8ND_on_aix-target2
Host name	saw116-sys1
Port	9943
User name	jmgadmin
Password	[REDACTED]
Confirm password	[REDACTED]

At the bottom of the dialog are three buttons: 'OK', 'Reset', and 'Cancel'.

Figure 26-16 Registering the deployment manager in the job manager

Complete the following steps to view your deployment manager or administrative agent with job manager:

- Click **System administration** → **Deployment manager**.
- Click the **Job managers** link under the Additional Properties section.

You should see a list similar to Figure 26-17. It lists every job manager instance in which your deployment manager server is registered.



Figure 26-17 Registered deployment manager ID

Complete the following steps to unregister your deployment manager from the job manager:

1. Click **System administration** → **Deployment manager**.
2. Click the **Job managers** link under the Additional Properties section.
3. Click the **Unregister from a Job Manager** button.
4. Supply the form from Figure 26-16 on page 968 with all of the information about your job manager profile. WebSphere Application Server uses this information to unregister your deployment manager.

Note that although only the Managed node name is the required field, you have to specify all information about your job manager to unregister it.

If some information is missing or is not correct, an error message similar to the one shown in Figure 26-18 is displayed.

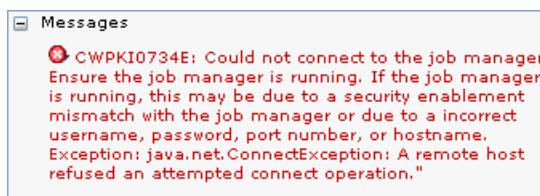


Figure 26-18 Error message when trying to unregister a host providing wrong data

26.5.7 Working with remote targets

In this section, the job manager is used to deploy applications on the target. We assume that you followed the previous steps and your environment is prepared to deploy applications.

You can download the sample applications from the WebSphere Application Server V8 samples website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=/com.ibm.websphere.nd.multiplatform.doc/info/ae/ae/welcome_ndmp.html

A JavaServer Faces 2.0 JSFSample example is used in this chapter.

Complete the following steps to install a new application on a remote target:

1. Copy the application ear file (JSFSample.ear) into the job manager directory profile_home/config/temp/JobManager.
2. Log on to the job manager console.
3. Click **Jobs** → **Submit**.
4. From the Job type box, select the **Distribute file** job and click **Next**.
5. Select your target and authentication method you want to use against the operating system and click **Next**.
6. Specify the source as JSFSample.ear. It is a relative path to the job manager directory from step 1.

Specify the destination where the file will be uploaded. In this case, use the /opt/IBM/WebSphere/AppServer/profiles/Dmgr8_01/downloadedContent path. Click **Next**.

Note: Your user must have access to the directory where you upload the application.

Figure 26-19 illustrates this step.



Figure 26-19 Copying the application to the target

7. Schedule the job and click **Next**.
8. Review the job details and click **Finish** to submit it.

When the job finishes successfully, the file is transferred to the directory of your choice on the target. Complete the following steps to install it remotely on the WebSphere Application Server target:

1. Click **Jobs** → **Submit**.
2. From the Job type box, select the **Install application** job and click **Next**.
3. Select your target and the authentication method you want to use against the deployment manager.
4. Specify the job parameters. Enter the application file name without the .ear suffix in the Application name field. For example, if you copied the application to the dmgr_profile_home/downloadedContent directory on remote deployment manager, you can simply enter the application full name (JSFSample.ear) in the Application location.

To specify on which server the application will be deployed, click the **Find** button next to the Server name. Select the server1 instance and click **OK**. See Figure 26-20 for details.

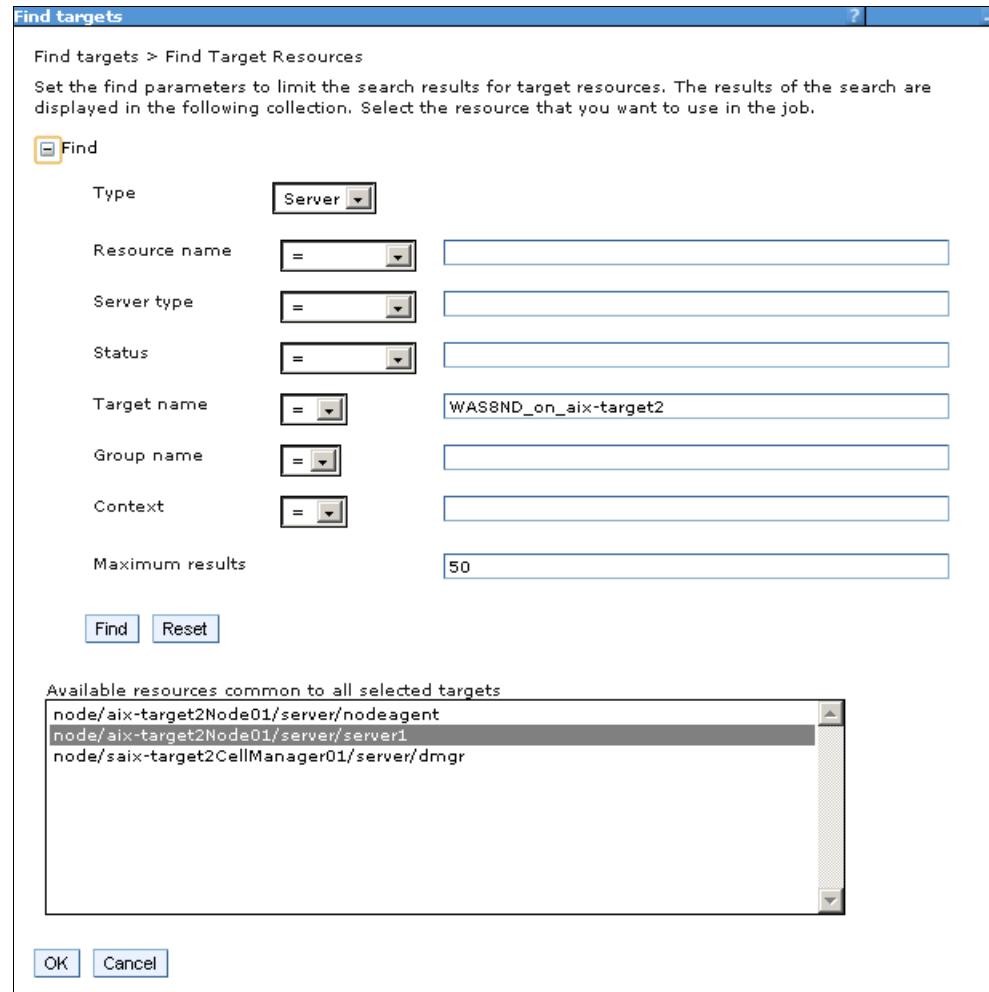


Figure 26-20 Specifying a target server for an application

After you specify the server instance, the Node name field should be automatically set with the correct server1 node name. Your form at this point should look similar to Figure 26-21.

The screenshot shows a dialog box titled "Submit a job to the job manager". On the left, a vertical navigation bar lists steps: Step 1: Choose a job type, Step 2: Choose job targets, Step 3: Specify job parameters (which is highlighted in yellow), Step 4: Schedule the job, and Step 5: Review the summary and submit the job. The main area is titled "Specify job parameters" and "Job type: Install application". It contains fields for Application name (JSFSample), Application location (JSFSample.ear), Server name (server1), Node name (aix-target2Node01), and Cluster name (empty). Each field has a "Find..." button to its right. At the bottom of the dialog are "Previous", "Next", and "Cancel" buttons.

Figure 26-21 Deploying an application using the job manager

If your topology uses application clusters, you should specify the cluster name on which the application will be deployed. In this case, we run it only on a single server, so you can omit this field and click **Next**.

5. Schedule the job and click **Next**.
6. Review the job information and click **Finish** to submit it.

When the job finishes successfully, the application is deployed on the remote target, but it is stopped. You need to start the server1 instance and then start the application.

Complete the following steps to start the server using job manager:

1. Click **Jobs** → **Submit**.
2. From the Job type box, select the **Start server** job and click **Next**.
3. Select the target and authentication method you want to use against the deployment manager and click **Next**.
4. Specify the **Server name** you want to start and use the **Find** button to select it from the list shown in Figure 26-20 on page 971. After you select the target server, the Node name field should automatically be set with the correct value. Click **Next**.
5. Schedule the job and click **Next**.
6. Review the job information and click **Finish** to submit it.

When server1 is available, you can start the JSFSample application using the Start application job by completing the following steps:

1. Click **Jobs** → **Submit**.
2. From the Job type box, select the **Start application** job and click **Next**.
3. Select the target and authentication method you want to use against the deployment manager and click **Next**.

- Specify the application name to start (Figure 26-22). Note that to use this job the application must already be deployed on the server. Use the **Find** button to select it and click **Next**.

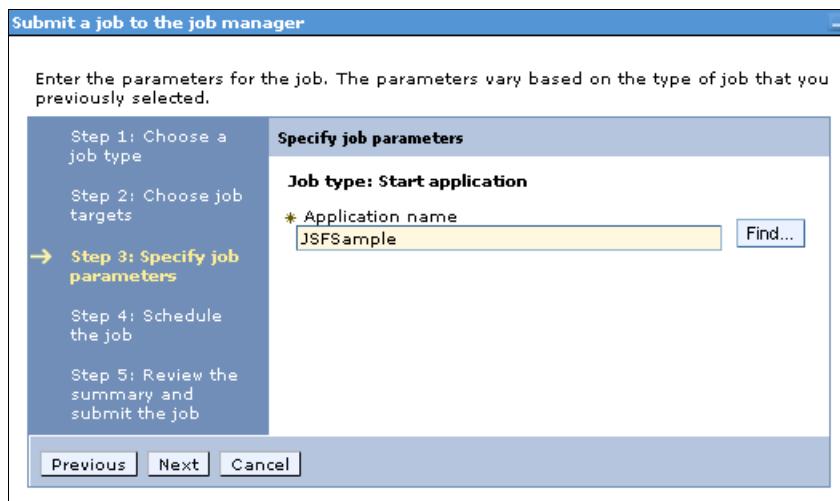


Figure 26-22 Starting the application on the target

- Schedule the job and click **Next**.
- Review the job information and click **Finish** to submit it.

When the job finishes successfully, you can try to invoke the application on the remote target. If you used the JSFSample application, use this URL to access application:

`http://<your target host>:<your port>/SampleTemplating/page1.faces`

You should see a window similar to the one shown in Figure 26-23.



Figure 26-23 JSFSample application output

26.5.8 Installing maintenance to remote targets

To install maintenance packages on remote targets, use the Manage offerings job with the appropriate response file. Before installation, make sure that all Java processes on the target run time are stopped.

- From the list, select the **manageOfferings** job to install maintenance:
 - In the administrative console, click **Job** → **Submit**.
 - In the Job type menu list, select the **Manage offerings** job and click **Next**.

c. Specify the target names and target authentication and click **Next**.

d. Specify this required parameter:

Response file path name

The full path name to the Installation Manager response file in which you specify the maintenance to install, repository, and target WebSphere Application Server run time. The path must point to a file located on the job manager or the deployment manager machine. You can prepare your response file using Installation Manager parameters shown in example in 26.5.4, “Installing WebSphere Application Server binaries on remote host” on page 962.

e. Specify these optional parameters:

IBM Installation Manager Path

Specify the path to install Installation Manager on the remote machine. If this parameter is blank, then Installation Manager is considered to be installed in the default location.

IBM Installation Manager agent data location

Specify an IBM Installation Manager data location that is not the default location for the manageOfferings job.

IBM Installation Manager key ring file

If the package repository requires a key ring file for authentication, specify the full path name of the key ring file on the job manager machine.

Key ring file password

If the key ring file is password protected, specify the key ring password.

f. Select **I accept the terms in the license agreements** and click **Next**.

g. Schedule the job or click **Next** to proceed with the next step.

h. Review the summary of the job and click **Finish** to submit it.

At this point, the job is submitted and has a unique identifier. You can track its status under the **Jobs → Status** link.

After a successful installation, you should see a message in stdOut.txt log similar to this one:

Updated to com.ibm.websphere.ND.v80_8.0.1.20110620_2048 in the /opt/IBM/WebSphere/AppServer directory

The change is also reflected in the console of the target. Log on to the updated server to see the new maintenance level. The current version will be displayed on the welcome window, as shown in Figure 26-24.

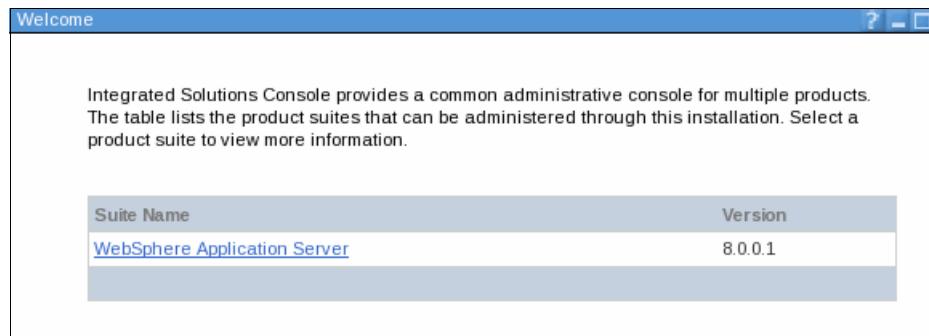


Figure 26-24 WebSphere Application Server binaries updated with Fix Pack 8.0.0.1

For more information about installing maintenance, refer to the Information Center at the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.installation.nd.doc/info/ae/ae/tagt_job_install_was_gui.html

26.5.9 Using the centralized installation manager with a command line

Every centralized installation manager operation that can be run from the job manager or deployment manager can also be run from the command-line interface. This interface allows you to create an even more automated WebSphere Application Server environment.

You can use the following list of jobs to work with your environment:

distributeFile	Distribute file.
findDataLocation	Add or search for Installation Manager agent data locations.
installIM	Install IBM Installation Manager.
installSSHPublicKey	Install SSH public key.
inventory	Inventory.
manageOfferings	Manage offerings.
manageprofiles	Manage profiles.
removeFile	Remove file.
runCommand	Run command on remote host.
testConnection	Test connection.
uninstallIM	Uninstall IBM Installation Manager.
updateIM	Update IBM Installation Manager.

To discover more about each job type, refer to the Information Center at the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.nd.multiplatform.doc/info/ae/ae/rxml_7jobtypes.html

26.6 Managing WebSphere Application Server V6.1 and V7 with the centralized installation manager

To use the centralized installation manager in WebSphere Application Server V8 with previous Version 6.1 or Version 7 products, the following additional steps must be completed:

1. Installation of the IBM Installation Factory
2. Configuration of the centralized installation manager repository for WebSphere Application Server V6.1 and V7 products
3. Installation of additional product packages into the repository
4. Creation of additional targets
5. Working with the environment

Note: The centralized installation manager for WebSphere Application Server V6.1 and V7 has limited functionality compared to Version 8. Refer to 26.2, “Planning considerations” on page 944 for more details.

26.6.1 Installing the IBM Installation Factory

You should use Installation Factory V7.0.0.15 or newer. You can download the current version from the IBM Installation Factory for WebSphere Application Server website:

<http://www-01.ibm.com/support/docview.wss?rs=180&uid=swg24020213>

After it is downloaded, extract the binaries to any directory for which you have the appropriate permissions. You can also install Installation Factory on your system using the code from the product media. Copy the Installation Factory on to your operating system. Use the `setupif` command provided on the Installation Factory disc, or as part of the package download:

- ▶ UNIX: Run the `setupif.sh` command or `setupif.sh target_location`.
- ▶ Windows: Run the `setupif.bat` command or `setupif.bat target_location`.

This command copies the Installation Factory to `user_home/InstallationFactory` by default. You can specify the target location by using the target location parameter.

26.6.2 Creating the centralized installation manager repository

Before you can populate the centralized installation repository, ensure that you have write access to the directories you will be using by completing the following steps:

1. Download the product images and expand the file (.tar or .zip) to a temporary directory or ensure access to the product CD.

2. Use the **ifcli.bat** command for Windows or **ifcli.sh** for UNIX to populate the repository. Example 26-6 shows a command to set up the centralized installation manager repository in the /opt/IBM/CIMRepo directory. The centralized installation manager will be configured for the WebSphere Application Server product installed under /opt/IBM/WebSphere/AppServer location and populated with WebSphere Application Server V7 binaries that are downloaded to the /tmp/installls/WAS7OND directory.

Example 26-6 Command for creating the central installation manager repository for WebSphere Application Server V6.1 and V7

```
installation_factory_install_root/bin/ifcli.sh -wasPath
/opt/IBM/WebSphere/AppServer -repositoryPath /opt/IBM/CIMRepo
-installationPackagePath /tmp/installls/WAS7OND
```

You can also use GUI based tools to create repositories. To learn more about using Installation Factory, go to Information Center at the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.installation.nd.doc/info/ae/ae/tins_the_centralized_installation_manager_package_add_if.html

Successful creation of the repository should result in the message shown in Example 26-7.

Example 26-7 Listing a successful repository creation

```
*****
The ifcli command started at Jun 27, 2011 3:12:39 PM with options: '-launch
-wasPath /opt/IBM/WebSphere/AppServer -repositoryPath /opt/IBM/the centralized
installation managerRepo -installationPackagePath /tmp/installls/WAS7OND '.
*****
The specified installation was successfully configured to associate with the
repository. Centralized Installation Manager must be restarted in order to use the
repository.
Adding installation package to the repository...
The installation package was successfully added to the repository.
*****
he ifcli command ended at Jun 27, 2011 3:13:52 PM with return code:
INSTCONFSUCCESS.
*****
```

You can review the structure of the repository in the /opt/IBM/CIMRepo directory. Refer to 26.4.2, “The centralized installation manager repository structure” on page 948 for more information about the repository structure.

26.6.3 Adding packages to the centralized installation manger repository when the deployment manager is connected to the Internet

Complete the following steps to download the latest version of the Update Installer to your centralized installation manager repository. Update Installer will be required to apply maintenance on the target systems.

1. Click **System Administration** → **Centralized Installation Manager** → **Installation Packages** → **Update Installer for WebSphere Application Server**.

2. Select one or more operating systems that you will use, then click **Download**. See Figure 26-25 for more details.

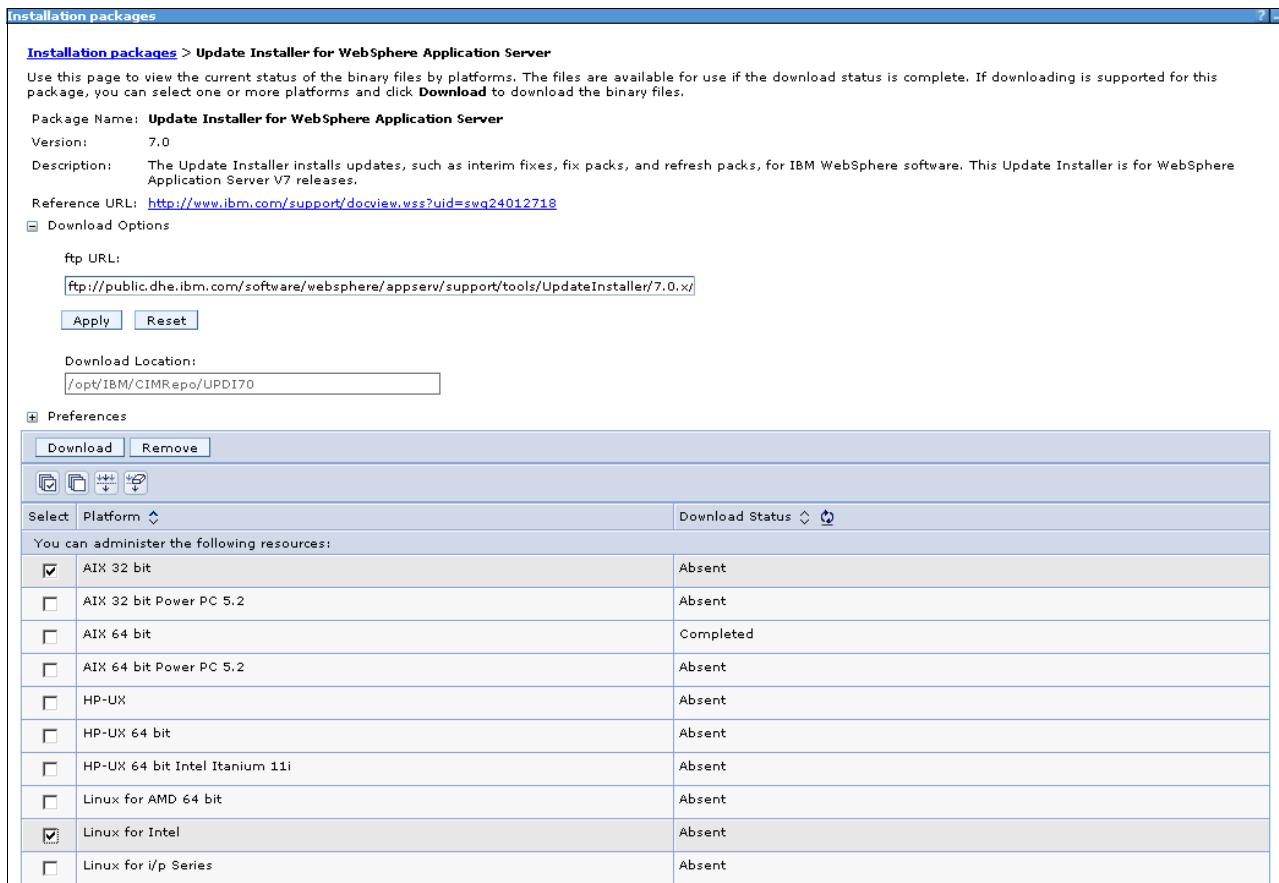


Figure 26-25 Downloading packages for centralized installation manager from the Internet

3. Review the summary, and select **Download** to start downloading the packages.
4. On the Installation packages window, after the download starts, you can monitor the status by clicking the **Refresh** button. If you receive errors, refer to the following file for more detailed information:
`<dmgr_profile_home>/logs/dmgr/systemOut.log`

Downloading descriptors

The centralized installation manager supports the installation of Network Deployment V6.1 and V7 Fix Packs on remote nodes that are within the Network Deployment cell. This configuration is known as a mixed-version cell, where the deployment manager node is at Version 7 or higher and the other nodes within the cell are either at the same level as the deployment manager node or at the Version 6.1 level. The centralized installation manager does not support maintenance levels below Version 6.1.

Download additional installation packages and maintenance files to your centralized installation manager repository by clicking **System Administration** → **Centralized Installation Manager** → **Installation Packages** → **add package**. On the next window, select one or more descriptor files (Figure 26-26) and click **Download** to proceed.

The screenshot shows a web-based interface titled 'Installation packages'. At the top, there's a navigation bar with a magnifying glass icon and a help icon. Below it, a sub-navigation bar shows 'Installation packages > Download descriptors'. A descriptive text block explains how to download descriptors from the IBM support Web site, mentioning the 'Download Options' and 'Preferences' sections. A large 'Download' button is prominently displayed. Below the button is a toolbar with icons for search, refresh, and other functions. A table lists several XML files, each with a checkbox column, a file name, type, and size. The first file, 'InstallPackageND70FP17.xml', has its checkbox checked. The table has columns for 'Select', 'File Name', 'Type', and 'Size (KB)'. The data in the table is as follows:

Select	File Name	Type	Size (KB)
<input checked="" type="checkbox"/>	InstallPackageND70FP17.xml	File	18
<input type="checkbox"/>	InstallPackageND70FP15.xml	File	19
<input checked="" type="checkbox"/>	InstallPackageND61FP35.xml	File	23
<input type="checkbox"/>	InstallPackageND61FP33.xml	File	23
<input type="checkbox"/>	InstallPackageND61FP31.xml	File	23

Figure 26-26 Downloading maintenance descriptors to the centralized installation manager repository

You can monitor the progress of the download by selecting the **Download Status** button.

Downloading fix pack binaries

After the descriptor for the required Network Deployment V6.1 Fix Pack has been downloaded using the method described here, download the *.pak files for that fix pack to the centralized installation manager repository.

The centralized installation manager uses the Update Installer for WebSphere Application Server V7 to install and uninstall the centralized installation manager defined Network Deployment V6.1 and V7 Fix Packs.

To download the binary files for a refresh pack, fix pack, or maintenance tool package type, which includes the Update Installer, complete the following steps:

1. Click **System Administration** → **Centralized Installation Manager** → **Installation Packages**, as shown in Figure 26-27. Select the package name in the table.

Installation packages			
Installation packages			
Use this page to add or remove installation packages in this cell.			
<input type="checkbox"/> Preferences			
	Add Packages	Remove Packages	
Select	Package Name	Version	Package Type
	Maintenance for WebSphere Application Server Network Deployment /opt/IBM/WebSphere/AppServer/properties /cim/InstallPackageND61Maintenance.xml	6.1	Interim fix
	Maintenance for WebSphere Application Server Network Deployment /opt/IBM/WebSphere/AppServer/properties /cim/InstallPackageND70Maintenance.xml	7.0	Interim fix
	Update Installer for WebSphere Application Server /opt/IBM/WebSphere/AppServer/properties /cim/InstallPackageUpdi70X.xml	7.0	Maintenance tool
	WebSphere Application Server Network Deployment /opt/IBM/WebSphere/AppServer/properties /cim/InstallPackageND70X.xml	7.0	Product install
<input type="checkbox"/>	WebSphere Application Server Network Deployment Fix Pack /opt/IBM/CIMRepo/descriptors/InstallPackageND70FP17.xml	7.0.0.17	Fix pack
Total 5			

Figure 26-27 Browsing available packages in the centralized installation manager

2. Figure 26-28 shows the next window. Select one or more platforms, then select **Download** to proceed.

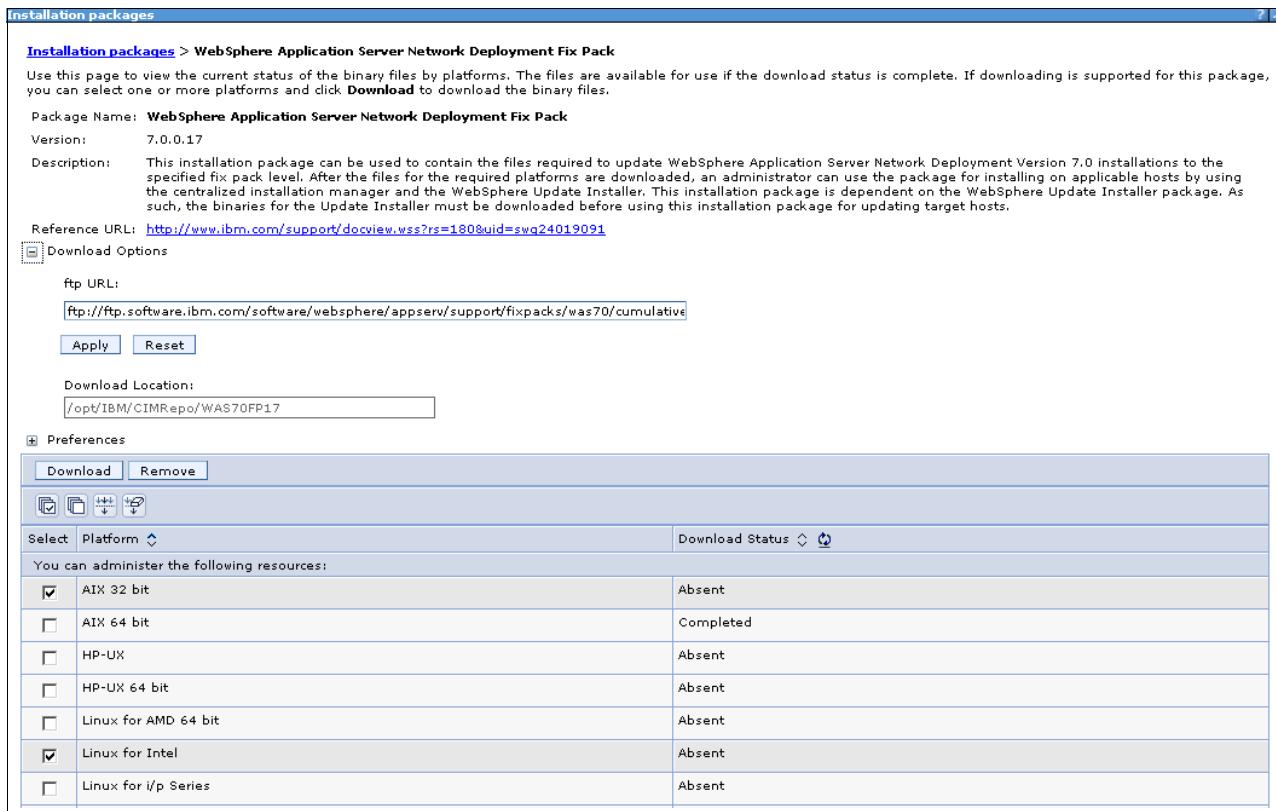


Figure 26-28 Downloading fix packs to the centralized installation manager repository

3. When all the required files have been downloaded, the Status column will display Complete. If one or more files are missing, the Download Status column displays an Incomplete status. In this case, you can try to download again. If your status is Incomplete, check for error messages in the *profile_root/logs/dmgr/SystemOut.log* file, where *profile_root* is the profile location of the deployment manager.

Downloading interim fixes

To download interim fixes, complete the following steps:

1. To download a specific APAR, click **System Administration** → **Centralized Installation Manager** → **Installation Packages**. Click the name of the package file in the table. A new window opens.
2. Select **Add files** to go to the Download Files window.

3. Enter the APAR number, and then select **Search**, as shown in Figure 26-29.

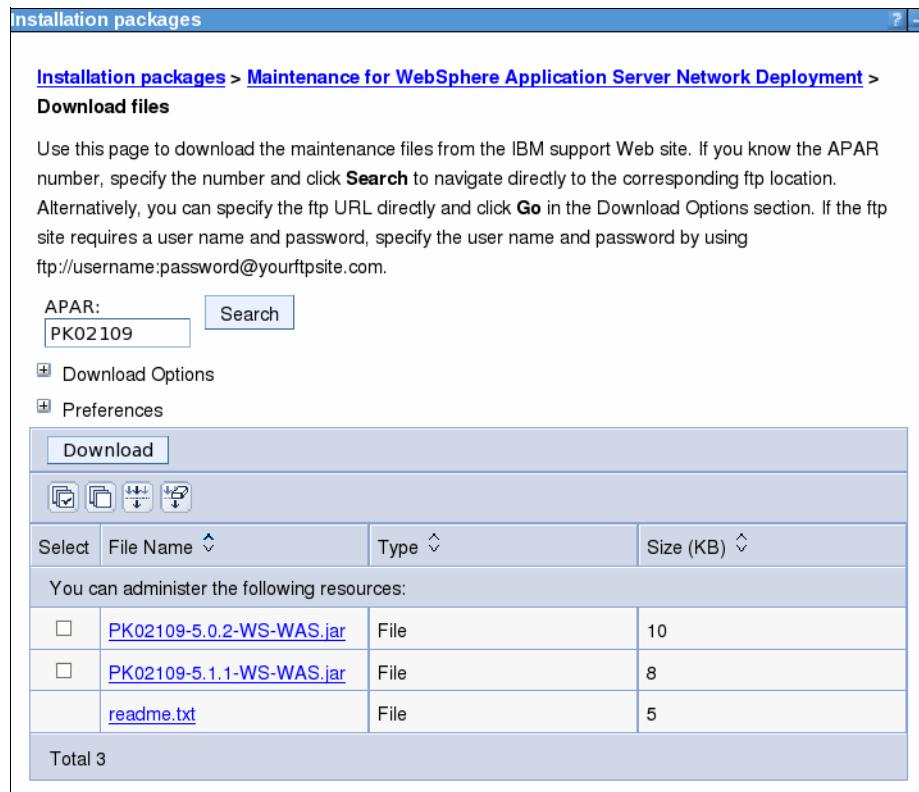


Figure 26-29 Searching for an interim fix using an Internet connection

4. Select the APAR from the list provided, then select **Download**, verify the information, and click **Download** to proceed.

26.6.4 When the deployment manager is not connected to the Internet

If your deployment manager system does not have Internet access, files will have to be manually downloaded and transferred to the centralized installation manager repository.

Before you can copy the downloaded files into the repository, ensure that you have set up the directory structure described in 26.6.2, “Creating the centralized installation manager repository” on page 976.

To obtain the address of the FTP site to manually download the required file, click **Administration console** → **System Administrator** → **Centralized Installation manager** → **Installation Packages**. The FTP URL format is:

`ftp://ftp.software.ibm.com/software/websphere/appserv/support/the centralized installation manager/the centralized installation manager70_yyyymmdd`

If the deployment manager does not have Internet access, an error message is displayed that the FTP URL is not known. Write down the FTP URLs, because they will be used on the system that has Internet access. See Figure 26-30.

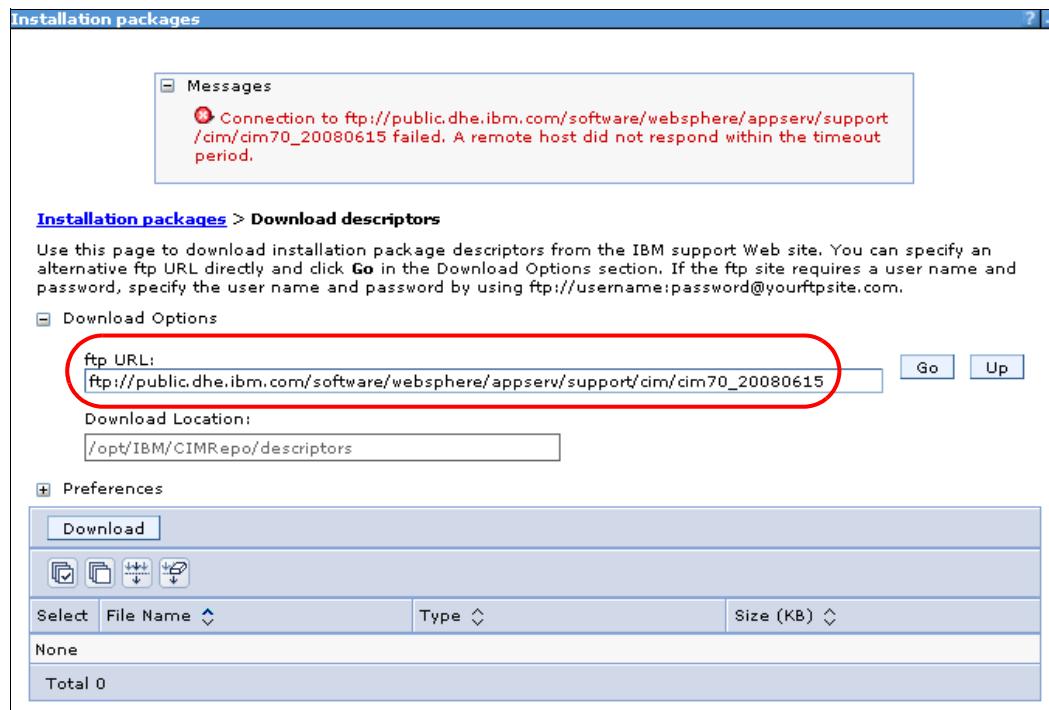


Figure 26-30 Obtaining the location of the fix packs download URL

You can find the address of the FTP site for the following items as follows:

- ▶ Descriptor files: Click **System Administrator** → **Centralized Installation manager** → **Installation Packages** and click the **Add packages** button. To determine the location of the FTP server, expand the **Download Options**, which gives you the FTP URL location used by the centralized installation manager for downloading the Descriptor files.
- ▶ Update Installer files: Click **System Administrator** → **Centralized Installation manager** → **Installation Packages** and click **Update Installer for WebSphere Application Server**. Expand the **Download Options**. This gives you the FTP URL location used by the centralized installation manager for downloading the Update Installer for the various operating systems.
- ▶ Fix packs: Copy descriptor files to the descriptor directory of your centralized installation manager, click **System Administrator** → **Centralized Installation manager** → **Installation Packages**, click the name representing the particular fix pack on the table (such as WebSphere Application Server Network Deployment Fix Pack 7.0.0.17), and then expand **Download Options**. This action gives you the FTP URL location used by the centralized installation manager for downloading cumulative fix packs and individual fixes. Choose the FTP URL for the type of fix you will be downloading.
- ▶ Interim Fixes: Click **System Administrator** → **Centralized Installation manager** → **Installation Packages**, click **Maintenance for WebSphere Application Server Network Deployment 6.1** or **Maintenance for WebSphere Application Server Network Deployment 7.0**, and then click **Add Files**. The FTP URL will be available under Download Options section.

With the FTP URL, you can go to any system with Internet access and download the required files. After you have the files downloaded, copy them to the correct directory structure in the centralized installation manager repository. For more information about this topic, refer to 26.4.2, “The centralized installation manager repository structure” on page 948 and the Information Center at the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=/com.ibm.websphere.installation.nd.doc/info/ae/ae/tins_cim_files_manual_add.html

Another option for obtaining the latest Update Installer, fix packs, and interim fixes would be to set up an FTP gateway on a system that has Internet access. Refer to the IBM Information Center at the following website for the steps to set up an FTP Gateway:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.installation.nd.doc/info/ae/ae/tins_cim_files_download_no_internet.html

26.6.5 Adding and removing additional installation targets

To install products or maintenance on remote hosts, you need to specify the targets.

Click **System Administration** → **Centralized Installation Manager** → **Installation Targets** → **Add Installation Target**.

Enter the host name, user name, and password of the Platform type of the system that you would like to add (Figure 26-31). Consider using a host name rather than an IP address because this name will be used in the configuration of the node.

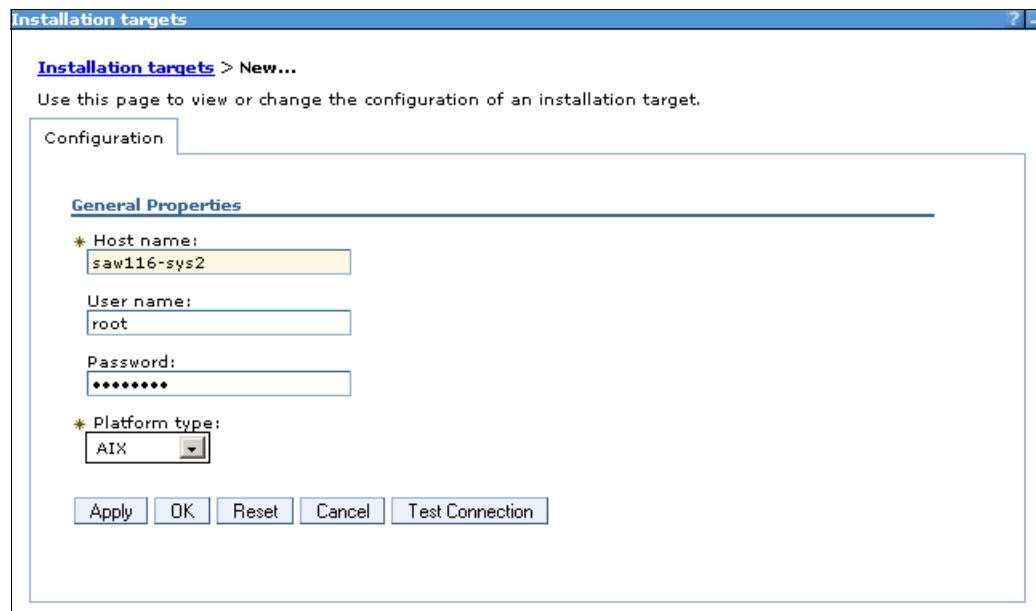


Figure 26-31 Defining targets for the centralized installation manager Version 6.1 and Version 7 targets

Click **OK**. You should now see the target in the list of target systems (Figure 26-32).



Figure 26-32 Listing the centralized installation manager Version 6.1 and Version 7 targets

To remove a target, click **System Administration** → **Centralized Installation Manager** → **Installation Targets** (Figure 26-32), select a target, and click **Remove Installation Target** button.

26.6.6 Installing a Secure Shell public key

Complete the following steps to install a Secure Shell (SSH) public key on specific installation targets:

1. Click **System administration** → **Installation Targets**, select one or more targets from the table, and click the **Install SSH Public Key** button.
2. On the next window, enter the user name and password and click **Next**.
3. You will be prompted for the specific SSH public key location (Figure 26-33). Enter the file location, then click **Next** to continue.

To learn more about how to obtain the keys, refer to 26.1, “The centralized installation manager prerequisites” on page 942.

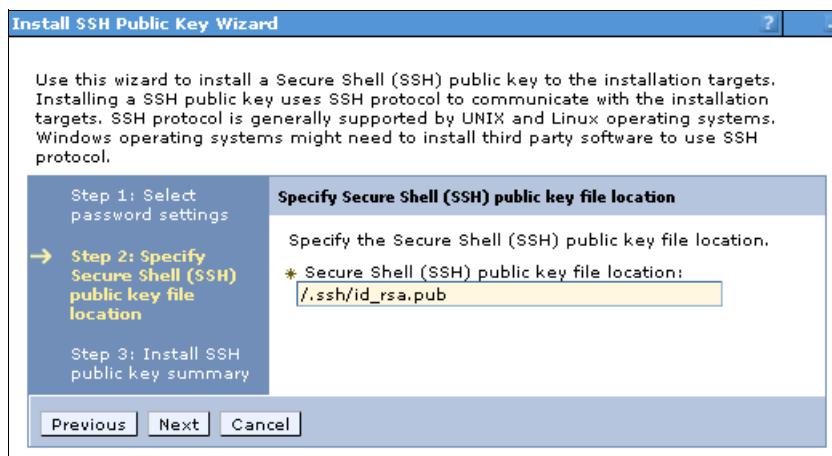


Figure 26-33 Specifying the public key location

- Verify the summary and click **Finish** to finish the task.

After this task finishes, you should see a window similar to Figure 26-34.

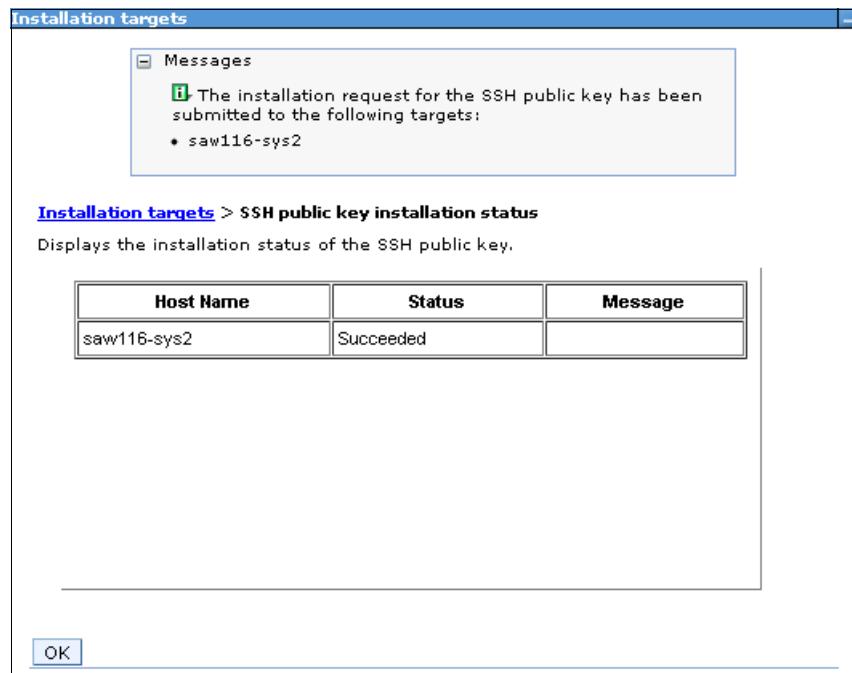


Figure 26-34 Successful installation of the public key on target message

You also see a key icon in the target list, as shown in Figure 26-35. This icon means that your deployment manager host can authenticate with that target using the public/private key pair method.

Add Installation Target	Remove Installation Target	Install SSH Public Key
Select	Host Name	SSH Public Key Installed
You can administer the following resources:		
<input type="checkbox"/>	saw116-sys1	
<input type="checkbox"/>	saw116-sys2	
Total 2		

Figure 26-35 Listing the centralized installation manager Version 6.1 and Version 7 targets

For more information about accessing your remote workstations by using the SSH public/private key pair authentication method, refer to “Installing a secure shell public key to access remote targets” on page 942 or go to the Information Center at the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.installation.nd.doc/info/ae/ae/tins_cim_targets_ssh.html

26.6.7 Installing packages to the target systems

The centralized installation manager relies heavily on remote node information maintained locally on the deployment manager node. This remote node information (namely the `node-metadata.properties` file) for each node is refreshed every time the node agent on the remote node starts and provides the centralized installation manager with up to date information regarding the WebSphere products and versions that are installed on the target nodes.

One example of how the `node-metadata.properties` file is used by the centralized installation manager is in the filtering of nodes that might be selected for the installation of an interim fix.

The `node-metadata.properties` file is also used by the centralized installation manager to determine only the applicable nodes during a maintenance installation. This process allows the cell administrator to see which nodes are potential candidates for this update and then initiate the installation of the interim fix on one or all the candidate nodes. Because of the availability of the `node-metadata.properties` file on the deployment manager node, you could use the centralized installation manager to perform this filtering without accessing the target nodes. The node agent process that runs on each node ensures that the `node-metadata.properties` files of the nodes on the deployment manager are kept up to date.

For this reason, if you apply maintenance to the node or install new WebSphere products (such as the Feature Pack for Web Services) outside of the centralized installation manager on the remote node, you must restart the node agent process after the installation to get the deployment manager copy of the `node-metadata.properties` file of the node up to date.

26.6.8 Product installation

Complete the following steps to install a software package:

1. Click **System Administrator** → **Centralized Installation Manager** → **Available Installations**.
2. Select the package type **Product Install** and select the installation package **WebSphere Application Server Network Deployment - 7.0**. When selecting a product install, you are required to select **Optional features**. You can choose from the following features:
 - Install the sample applications for learning and demonstration environments.
 - Install the non-English language files for using the administrative console from machines with non-English locales. If you do not select this option, then only the English language pack is installed.
 - Install the non-English language files that support the application server runtime environment, such as the `wsadmin` tool and logging. If you do not select this option, then only the English language pack is installed.

- Click the **Show Installation Targets** button to list defined targets (Figure 26-36) and select the targets on which you want to install the product.

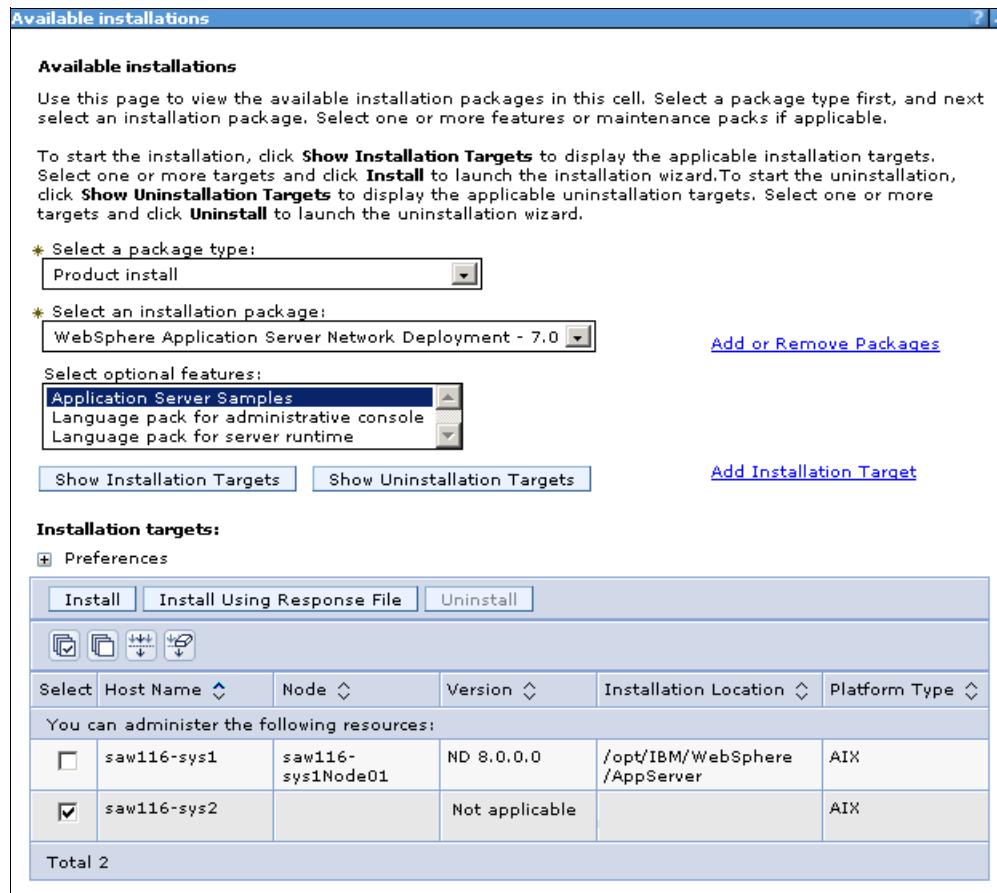


Figure 26-36 Installation of WebSphere Application Server V7 using the centralized installation manager

- Click **Install** → **Accept License Agreement** → **Next**.
- Select **Authentication method** to access the target. You can choose a user name and password or the use of Secure Shell (SSH) Public/Private key authentication.
- In the next window, provide the installation directory and a working (temporary) directory, then click **Next**.
- The next two windows give you the option to disable prerequisite checking and to accept limitations to allow installing as a non-root user.
- By default, the centralized installation manager uses 64-bit installation binaries on 64-bit operating systems. The next window allows you to override this setting. Click **Next**.
- Review the Summary window, then select **Finish** to start the installation.
- To watch the progress of the installation, click **System Administrator** → **Centralized Installation Manager** → **Installation Progress**.

11. When the installation is complete, click **System Administrator** → **Centralized Installation Manager** → **Installation History** for details about the installation (Figure 26-37).

Select	Host Name	Operation	Package and Features	Creation Time	Completion Time	Completion Status
You can administer the following resources:						
<input type="checkbox"/>	saw116-sys2	Install	WebSphere Application Server Network Deployment Fix Pack - 7.0.0.17 ● 7.0.0-WS-WAS-AixPPC64-FP0000017.pak - Applied ● 7.0.0-WS-WASSDK-AixPPC64-FP0000017.pak - Applied	2011-06-28 12:06:09	2011-06-28 12:25:56	Succeeded View Details
<input type="checkbox"/>	saw116-sys2	Install	WebSphere Application Server Network Deployment - 7.0 ● Language pack for administrative console ● Language pack for server runtime	2011-06-27 15:19:17	2011-06-27 15:37:04	Succeeded View Details
Total 2						

Figure 26-37 The centralized installation manager installation history for Version 6.1 and Version 7

26.6.9 Installing maintenance to a target system

Before you can install a fix pack on WebSphere Application Server V6.1 or V7 target systems, you must install it locally first on the deployment manager host using the Update Installer for WebSphere software. The centralized installation manager cannot be used to install maintenance to the deployment manager.

All node agents in the cell must be stopped on the target systems. If the node agents or any other server processes are running, it is up to the administrator to make sure that they all have been stopped.

Note: There is no need to explicitly install the Update Installer on the targets first before initiating the installation of the fix packs or interim fixes, because the centralized installation manager automatically installs the latest version of the Update Installer on the target if needed.

After the centralized installation manager successfully completes the installation process on a remote node, it then deletes the installation image files that are located in the temporary location that you specify during the installation process. If the installation is unsuccessful, the files remain in the temporary location for you to use to determine what caused the installation error. However, you can safely delete these files.

Important: Fix packs that include updates to the Software Development Kit (SDK) might overwrite unrestricted policy files. Back up unrestricted policy files before you apply a fix pack and reapply these files after the fix pack is applied.

Using the centralized installation manager to install refresh or fix packs

To use the centralized installation manager to install refresh or fix packs, complete the following steps:

1. Click **System Administrator** → **Centralized Installation Manager** → **Available Installations**. For the package type, select refresh pack, fix pack, or maintenance tool.
2. Next, from the drop-down list of available installation packages, choose the installation package that contains the refresh pack or fix pack that you want to install on your remote systems. These are the packages that you previously downloaded to your centralized installation manager repository.
3. Click **Show installation targets** to get a list of target systems available for install. Select your target systems. To continue, click **Install**.
4. The next window shows the license agreement. Review the agreement, select **Agree**, and then click **Next**.
5. On the next window, you can specify the authentication method you want to use, as well as your user name and password or Secure Shell (SSH). Choose your method, and then click **Next** to proceed.
6. The next window that opens depends on the type of authentication you choose. You see a window prompting you to enter a user name and password or a window prompting you for the location of the SSH private key file and keystore password.
7. Verify the installation and the working location of the installation targets. The installation location is the remote location of each installation target in which the package is to be installed. The working location specifies the directory on the remote target where the files are sent before the package is installed in the specified location. Make sure that you have enough disk space in both the installation location and the working location. Click **Next** to continue.
8. A summary window opens. Review the information entered, then to start the installation, select **Finish**.

Note: Any interim fixes that you previously installed on the remote targets are uninstalled by the Update Installer prior to installing the refresh pack or fix pack. If the refresh pack or fix pack does not include the official fixes that were included in the removed interim fixes, you must reinstall the interim fixes after you install the refresh pack or fix pack.

9. You can click **System Administrator** → **Centralized Installation Manager** → **Installation Progress** to check the progress of the installation. When complete, click **System Administrator** → **Centralized Installation Manager** → **Installation History** for details about the installation. Figure 26-37 on page 989 shows a history of installation for the WebSphere Application Server Network Deployment V7 product with language packs and also shows the installation of Fix Pack V7.0.0.17 on the same product.

Using the centralized installation manager to install interim fixes

Complete the following steps to install an interim fix on remote target:

1. Click **System Administrator** → **Centralized Installation Manager** → **Available Installations**. From the drop-down menu under “Select a package type”, select **Interim fix**.

On the drop-down menu under “Select an Installation package”, select either **Maintenance for WebSphere Application Server Network Deployment 7.0 or 6.1**. If the interim fixes have been previously downloaded to the centralized installation manager repository, they will be displayed under “Select one or more maintenance packs”. Select the maintenance pack you want to install on your target systems, as illustrated in Figure 26-38.

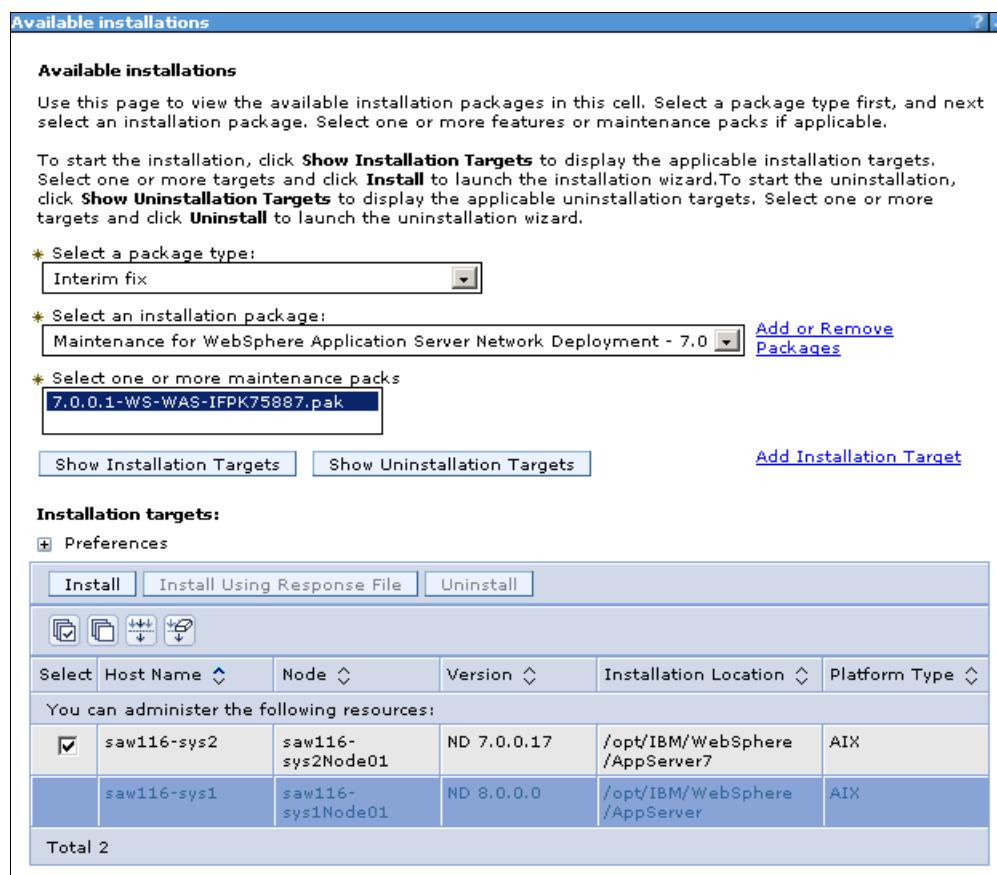


Figure 26-38 Installing an interim fix using the centralized installation manager for Version 6.1 and Version 7

2. Click the **Show Installation Targets** button. A list of your target systems will be displayed. From this list, select your targets, then click **Install**.

Note: Only the first target is available to select on the window shown in Figure 26-38, because it is the only one that is applicable for that interim fix. On the second target, there is an installation of WebSphere Application Server V8, for which this fix pack is not valid.

3. Click **View License Agreement** to read the agreement and accept the terms. Click **Next** to continue.

4. The next window is where you choose the authentication method you want to use, user name, and password or Secure Shell (SSH). Choose your method, then click **Next** to proceed.
5. The next window that opens depends on the type of authentication you choose. A window that prompts you to enter a user name and password opens or a window that prompts you for the location of the SSH private key file and keystore password opens.
6. Verify the installation and the working location of the installation targets. The installation location is the remote location of each installation target in which the package is to be installed. The working location specifies the directory on the remote target where the files are sent before the package is installed in the specified location. Make sure you have enough disk space in both the installation location and the working location. Click **Next** to continue.
7. Verify the Summary window and select **Finish** to start the installation request. To check the status of your request, select **Installations in Progress** on the administrative console. After the installation is completed, click **Installation history** in the administrative console to review the log files for each of the installation requests that you submit. From the Installation History window, the administrator can click **View Details** to open a window with additional details about the results. Links to logs on the remote targets are included. However, those logs can be moved, replaced, or deleted by other users or administrator if they are not viewed immediately after an installation operation.

If you attempt to install an interim fix without having a copy of the IBM Update Installer for WebSphere Software in your centralized installation manager repository, you receive the following error message:

The installation binary files required for the `install_package_name` or its dependent package Update Installer for WebSphere Application Server for `workstation_platform` do not exist.

26.6.10 Uninstalling packages

Use the following steps to use the centralized installation manager to uninstall previously installed packages on your installation targets. The tasks available for uninstall will depend on your environment.

1. Click **System Administrator** → **Centralized Installation Manager** → **Available Installations**. On this window, select the package type you would like to uninstall. Then select the installation package to be uninstalled. Click **Show Uninstallation Targets** → **Target system** and then click the **Uninstall** button. See Figure 26-39.

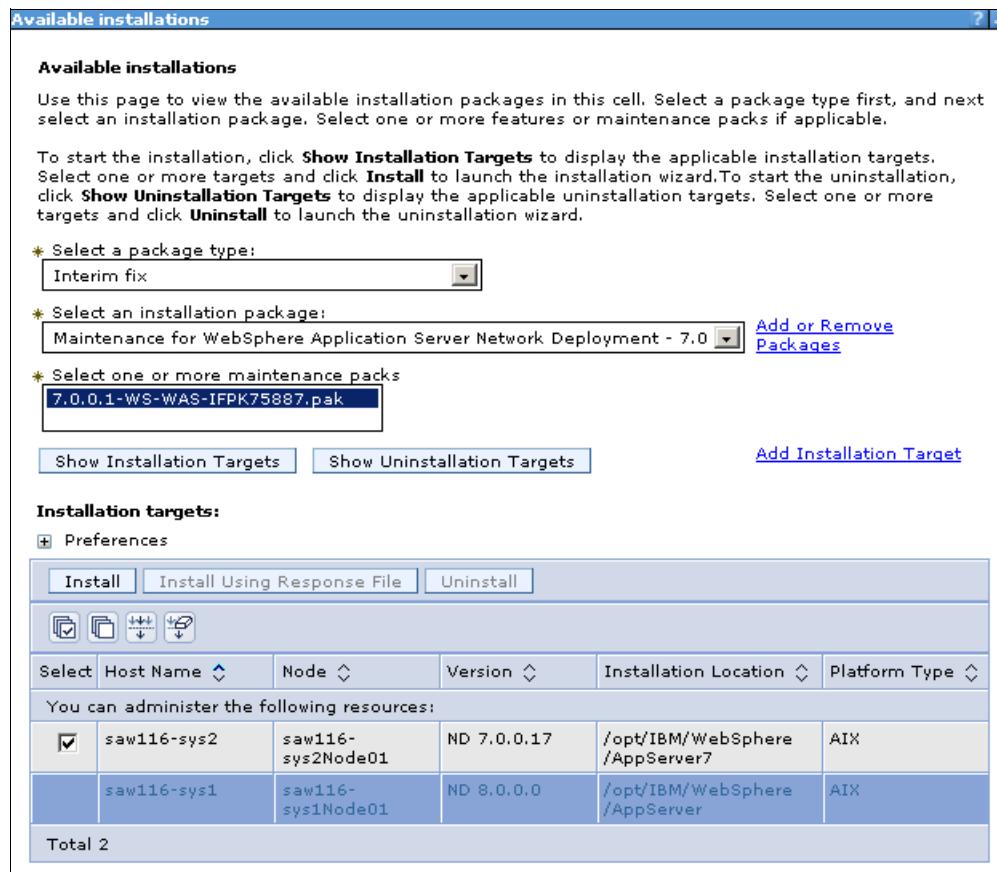


Figure 26-39 Uninstalling the fix pack using the centralized installation manager

2. You are prompted for an authentication method. Enter either a user name and password or the location of the SSH key file. Next, verify the Installation target directory. Review the summary page, then click **Finish** to proceed with the uninstallation.

Refer to the installation history for the results of the uninstallation.

26.6.11 The centralized installation manager AdminTask commands

You can use the centralized installation manager **AdminTask** commands with Jacl or Jython scripting language. These commands and parameters can be used to install, uninstall, and manage various software packages and maintenance files in the centralized installation manager environment. Here is a list of **AdminTask** commands available for WebSphere Application Server V6.1 and V7 products:

installWASExtension	Installs a server extension package.
installSoftware	Installs a specified software package.
installWithResponseFile	Installs a specified software package using parameters from a response file.
installMaintenance	Installs maintenance.

listPackagesForInstall	Lists packages from the centralized installation repository.
listFeaturesForInstall	Lists the features of software packages from the centralized installation repository.
showPackageInfo	Displays information about a specific software package.
showLicenseAgreement	Displays the license agreement for a specified installation package.
getManagedNodesOnHostByInstallLoc	Lists the names of managed nodes defined in the deployment manager cell.
listManagedNodesOnHost	Lists the names of the managed nodes located on targets in the deployment manager cell.
testConnectionToHost	Verifies if a connection from the deployment manager to a remote host can be established using a user name and password.
testConnectionToHostUsingSSHKey	Verifies if a connection from the deployment manager to a remote host can be established using an SSH private key.
installSSHPublicKeyOnHost	Installs an SSH public key on the remote target.
listKeyInstallationRecords	Lists the centralized installation manager SSH public key records of target host names.
updateKeyInstallationRecords	Updates the centralized installation manager SSH public key records.
listPendingRequests	Lists submitted installation or uninstallation requests that are not started yet.
listInProgressRequests	Lists submitted installation or uninstallation requests that are in progress.
listRequestsForTarget	Lists submitted installation or uninstallation requests for given target.
showLatestInstallStatus	Displays the status of the most recently submitted installation request.
showLatestUninstallStatus	Displays the status of the most recently submitted uninstallation request.
uninstallSoftware	Uninstalls the software package from the target.
uninstallMaintenance	Uninstalls the maintenance package from the target.

Refer to the Information Center at the following website for detailed information about these commands:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.installation.nd.doc/info/ae/ae/rxml_cim_commands.html



System recovery

This chapter provides information about using WebSphere Application Server V8 recovery enhancement mechanisms. WebSphere Application Server V8 comes with several features that can be used to recover its environment. System recovery might be essential after a hardware or software malfunction. In this chapter, you find information about:

- ▶ Using backups and configuration archives to restore the system
- ▶ Restoring transactions
- ▶ Recovering a failed node
- ▶ Moving a node to a different system
- ▶ Recreating a cell from a template

Important: To keep your WebSphere Application System environment healthy, monitor the state of the system and deployed applications to discover possible hardware and software malfunctions. For more information about WebSphere Application Server monitoring, refer to Chapter 15, “Monitoring distributed systems” on page 541.

27.1 Backups

This section summarizes some of the most common system management backup tasks with regard to configuration files:

- ▶ Backing up a profile
- ▶ Restoring a profile
- ▶ Exporting and importing servers or profiles

27.1.1 Backing up a profile

Use the **backupConfig** command to back up a profile configuration. Run this command from the *was_home/bin* directory, using the **-profileName** option to specify the profile to back up. Or, execute the command from the *profile_home/bin* directory to back up only this particular profile.

The **backupConfig** command compresses the configuration files and stores the compressed file in the current directory or a specified path. The compressed file can be restored using the **restoreConfig** command. By default, **backupConfig** stops all servers in the configuration before performing the backup. The syntax is shown in Example 27-1.

Example 27-1 backupConfig command

```
Usage: backupConfig [backup_file] [-nostop] [-quiet] [-logfile <filename>]
[-replacetolog] [-trace] [-username <username>] [-password <password>] [-profileName
<profile>] [-help]
```

The **backup_file** parameter specifies the file where the backup is to be written. If you do not specify a backup file name, a unique name is generated, and the file is stored in the working directory. If you specify a backup file name in a directory other than the current directory, the specified directory must exist.

For more information about these command options and usage, refer to the Information Center at the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.nd.multiplatform.doc/info/ae/ae/rxml_backupconfig.html

Example 27-2 shows the use of the **backupConfig** command to back up the profile configuration, AppSrv8_02. If you run the command without specifying the name of the backup file or log file, these files are generated by the WebSphere Application Server automatically.

Example 27-2 backupConfig example

```
./backupConfig.sh -profileName AppSrv8_02
ADMU0116I: Tool information is being logged in file
            /opt/IBM/WebSphere/AppServer/profiles/AppSrv8_02/logs/backupConfig.log
ADMU0128I: Starting tool with the AppSrv8_02 profile
ADMU5001I: Backing up config directory
            /opt/IBM/WebSphere/AppServer/profiles/AppSrv8_02/config to file
            /opt/IBM/WebSphere/AppServer/bin/WebSphereConfig_2011-07-10.zip
ADMU0505I: Servers found in configuration:
ADMU0506I: Server name: nodeagent
ADMU0506I: Server name: server1
ADMU2010I: Stopping all server processes for node aix-target2Node01
Realm/Cell Name: <default>
```

```
Username: admin
Password:
ADMU0510I: Server server1 is now STOPPED
Realm/Cell Name: <default>
Username: admin
Password:
ADMU0510I: Server nodeagent is now STOPPED
.....
.....
ADMU5002I: 1,164 files successfully backed up
```

Note: Consider using the **backupConfig** command before making any significant configuration changes to your WebSphere Application Server environment. If for any reason a new configuration causes the server to fail or behave in an unstable way, the fastest way to restore it is to use the backup configuration file.

27.1.2 Restoring a profile

Use the **restoreConfig** command to restore a profile configuration from an archive that was previously generated using **backupConfig** for that profile. If the configuration to be restored exists, the configuration directory is renamed to config.old (then config.old_1, and so on) before the restore begins. The command then restores the entire contents of the *profile_root/config* directory. By default, all servers on the node stop before the configuration restores so that node synchronization does not occur during the restoration.

The **restoreConfig** command syntax is shown in Example 27-3.

Example 27-3 restoreConfig command

```
Usage: restoreConfig backup_file [-location restore_location] [-quiet] [-nostop]
[-nowait] [-logfile <filename>] [-replacelog] [-trace] [-username <username>]
[-password <password>] [-profileName <profile>] [-help]
```

For more information about the command, refer to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.nd.multiplatform.doc/info/ae/ae/rxml_restoreconfig.html

Example 27-4 shows the results of a restore procedure for an application server profile. If you do not supply the optional parameter for the log file, WebSphere Application Server generates it automatically.

Example 27-4 restoreConfig command usage example

```
./restoreConfig.sh WebSphereConfig_2011-07-10.zip -profileName AppSrv8_02
ADMU0116I: Tool information is being logged in file
    /opt/IBM/WebSphere/AppServer/profiles/AppSrv8_02/logs/restoreConfig.log
ADMU0128I: Starting tool with the AppSrv8_02 profile
ADMU0505I: Servers found in configuration:
ADMU0506I: Server name: nodeagent
ADMU0506I: Server name: server1
ADMU2010I: Stopping all server processes for node aix-target2Node01
ADMU0512I: Server server1 cannot be reached. It appears to be stopped.
ADMU0512I: Server nodeagent cannot be reached. It appears to be stopped.
ADMU5502I: The directory
    /opt/IBM/WebSphere/AppServer/profiles/AppSrv8_02/config already
```

```
exists; renaming to  
/opt/IBM/WebSphere/AppServer/profiles/AppSrv8_02/config.old  
ADMU5504I: Restore location successfully renamed  
ADMU5505I: Restoring file WebSphereConfig_2011-07-10.zip to location  
/opt/IBM/WebSphere/AppServer/profiles/AppSrv8_02/config  
.....  
.....ADMU5506I: 1,164 files successfully restored  
ADMU6001I: Begin App Preparation -  
ADMU6009I: Processing complete.  
ADMU6002I: Begin Asset Preparation -  
ADMU6009I: Processing complete.
```

27.1.3 Exporting and importing profiles

You can also use the export and import capabilities of the WebSphere Application Server configuration. This procedure uses a compressed archive (CAR) file.

The general procedure to use an archived file is as follows:

1. Export a WebSphere configuration, which creates a compressed archived file containing the server configuration.
2. Optionally, extract the files for browsing or updating for use on other systems. For example, you might need to update resource references.
3. Upload the archive file to the target system.
4. Import the archive file. The import process requires that you identify the object in the configuration you want to import and the target object in the existing configuration. The target can be the same object type as the archive or its parent. Note that:
 - If you import a server archive to a server configuration, the configurations are merged.
 - If you import a server archive to a node, the server is added to the node.

Note: You can use this capability between two application server profiles under the same or different product installations, on the same or different host environments. In general, you can also use this capability to replicate a profile configuration across different platforms, as long as you do not add operating system-specific system properties to your configuration. An exception to this is that configurations of z/OS and distributed platform profiles are not compatible, so you cannot replicate configurations between these platforms.

Server archives

The following JACL command, executed from `wsadmin` from a `profile_home/bin` directory of the application server, can be used to create a CAR archive of this server:

```
$AdminTask importServer {-nodeInArchive saw116-sys2Node02 -serverInArchive server1  
-archive /tmp/archive/server1_archive.car}
```

This process removes applications from the server that you specify, and breaks the relationship between the server that you specify and the core group of the server, cluster, or bus membership. If you export a single server of a cluster, the relation to the cluster is eliminated in the archive file.

The following command imports an archived server template from the previous example as server2 into node saw116-sys1Node01:

```
$AdminTask importServer {-nodeName saw116-sys1Node01 -serverName server2 -archive /tmp/archive/server1_archive.car}
```

After importing, save your changes to the WebSphere Application Server run time by running the Jacl **save** command:

```
$AdminConfig save
```

When you use the **importServer** command, you select a configuration object in the archive as the source and select a configuration object on the system as the target. The target object can match the source object or be its parent. If the source and target are the same, the configurations are merged. Figure 27-1 shows the Application Server view in the application server console after importing server2. As you see, server2 is added to the existing configuration.

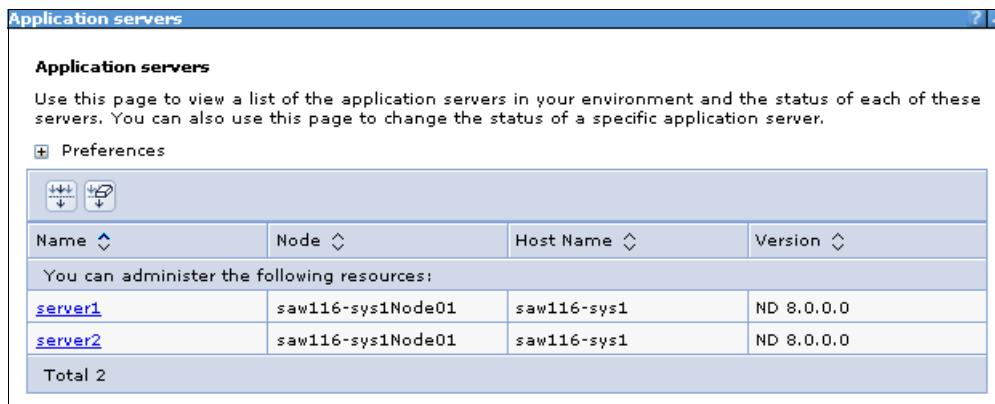


Figure 27-1 Template server imported as server2

Profile archives

You can also create a configuration archive file containing the configuration of an application server profile for later restoration. This archive file can be used to clone the original profile on another machine or system.

Note: You can only create an archive of an unfederated profile (stand-alone application server).

The following command executed from **wsadmin** from the *profile_home/bin* directory creates an archive of a profile:

```
$AdminTask exportWasprofile {-archive /tmp/archive/myProfile_archive.car}
```

The following command imports and overwrites the profile with the archive file configuration:

```
$AdminTask importWasprofile {-archive /tmp/archive/myProfile_archive.car  
-deleteExistingServers}
```

Note that the **-deleteExistingServers** parameter is optional. It deletes existing servers on the node configuration. Note also that when importing a profile with multiple servers, the node has to contain exactly the same number of servers. If this is not correct, the following error message will occur:

ADMB0016E: The number of servers in the configuration archive does not match the number of servers in the system configuration. The product only supports importWasprofile for profiles with the same number of servers.

There are also other options that enable you to import and export WebSphere Application Server proxy servers and proxy profiles. The **ConfigArchiveOperations** command group for the AdminTask object includes the following commands:

- ▶ **exportProxyProfile**
- ▶ **exportProxyServer**
- ▶ **exportServer**
- ▶ **exportWasprofile**
- ▶ **importProxyProfile**
- ▶ **importProxyServer**
- ▶ **importServer**
- ▶ **importWasprofile**

For more information about using these commands, refer to the Information Center at the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.express.doc/info/exp/ae/rxml_atconfigarchive.html

27.2 Recovery of transactions

WebSphere Application Server V8 stores information about transactions that it manages in the form of persistent logs. The default directory for the transaction logs is:

profile_home/tranlog

These logs can be used to recover transactions that did not complete due to, for example, a hardware or software server failure.

The WebSphere Application Server recovers transactions during startup. You can also force it to recover transactions by specifying a parameter in the **startServer** command:

`startServer server1 -recovery`

This additional **-recovery** parameter specifies that the server will start in special recovery mode. In this mode, it performs a transaction recovery only, after which it shuts down. During recovery, the server does not accept new transactions. After restart, it releases any resources that were unavailable due to questionable transactions.

The WebSphere Application Server also supports more complex recovery cases when in a clustered environment. In a clustered environment, a running instance of a server can recover failed transactions of another server from the same cluster. There is no need to restart a failed server to recover its transactions, because this can be done by another server in the cluster. This is important when a hardware failure is the cause of server unavailability, and the time to recover from the hardware fault might be long.

There are two possible ways to recover and release the locks of another server in a cluster:

Automated peer recovery	If an application server fails, WebSphere Application Server automatically selects another server to perform peer recovery processing on its behalf. Apart from enabling high availability and configuring the recovery log location for each cluster member, no additional WebSphere Application Server configuration steps are required to use this mode. This is the default recovery option for WebSphere Application Server installations.
Manual peer recovery	If an application server fails, an administrator can use the administrative console to select a particular server in the cluster to perform recovery processing on behalf of the failed server. To use this recovery mode, additional configuration steps are required.

Configuration and usage of the clustered environment transaction recovery is out of the scope for this book. However, a well documented article on this topic is available from IBM developerWorks® at the following website:

http://www.ibm.com/developerworks/websphere/techjournal/0504_beaven/0504_beaven.html

The latest documentation in the Information Center is available at the following websites:

- ▶ http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.nd.doc/info/ae/ae/tjta_administer.html
- ▶ http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.nd.doc/info/ae/ae/cjta_trans_ha.html

27.3 Environment recovery

WebSphere Application Server V8 introduces a new feature to simplify the recovery of nodes. The WebSphere Application Server **addNode** command is now extended with the additional **-asExistingNode** parameter. This parameter allows you to shorten the time required to:

- ▶ Recover an existing managed node of a deployment manager
- ▶ Move a node to a product installation on a different computer at the same or different path
- ▶ Create a cell from a template cell

After using this parameter, you should consider updating the virtual hosts (host aliases) to include the target host name of the application server node. If the node uses a Secure Sockets Layer certificate, you should also change the default certificate that contains the host name of the new node.

Refer to the Information Center at the following websites for information about replacing SSL certificates:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.nd.doc/info/ae/ae/tsec_sslreplaceselfsigncert.html

You might also need to update the configuration of other infrastructure components, such as web servers, that are statically configured to use application servers residing on specific hosts.

Note: Some of the **addNode** command parameters are incompatible with the **-asExistingNode** option. Do *not* use **-asExistingNode** with the following parameters:

- ▶ **includeapps**
- ▶ **includebuses**
- ▶ **startingport**
- ▶ **portprops**
- ▶ **nodeagentshortname**
- ▶ **nodegroupname**
- ▶ **registerservice**
- ▶ **serviceusername**
- ▶ **servicepassword**
- ▶ **coregroupname**
- ▶ **excludesecuritydomains**

27.3.1 Rapid node recovery

In previous WebSphere Application Server versions, to recover a damaged node without the backup, you had to complete the following steps:

1. Stop the deployment manager.
2. Make a backup of the deployment manager configuration.
3. Use **removeNode** on the deployment manager to remove the node to be recovered.
4. Reinstall WebSphere Application Server.
5. Create a profile for a stand-alone application server node.
6. Use **addNode** to add the application server node to the deployment manager.
7. Restore the backed up configuration to the deployment manager.
8. Run the **syncNode** command.

With WebSphere Application Server V8, this procedure is simplified to the following steps:

1. Reinstall WebSphere Application Server V8 in the same directory as previously used.
2. Create a profile with the same node name and path.
3. Recover the damaged node using **addNode -asExistingNode**.
4. Synchronize the nodes in the cell.

Figure 27-2 illustrates this procedure after a node1 failure. When the node is unavailable, but the node information remains on the deployment manager, use the `-asExistingNode` option to recreate the unavailable node.

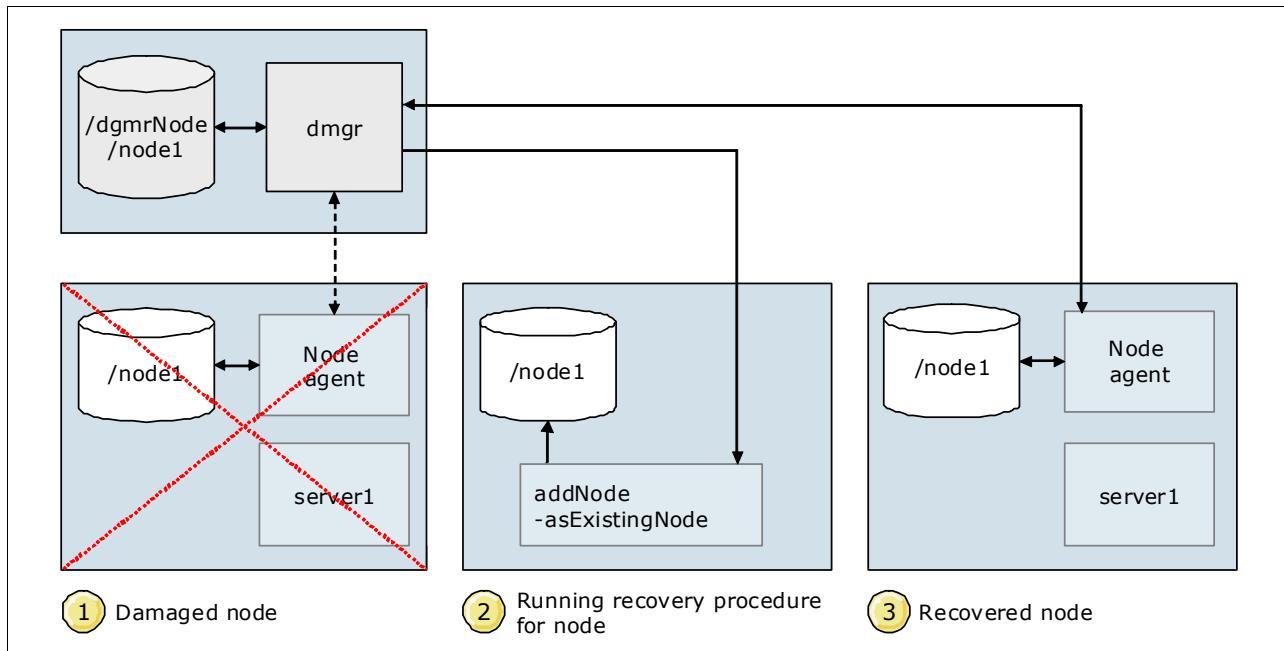


Figure 27-2 Recovering a failed node

In this example, an existing managed node of a deployment manager cell will be recovered. In Figure 27-3, you can see that node aix-target2Node01 is unavailable.

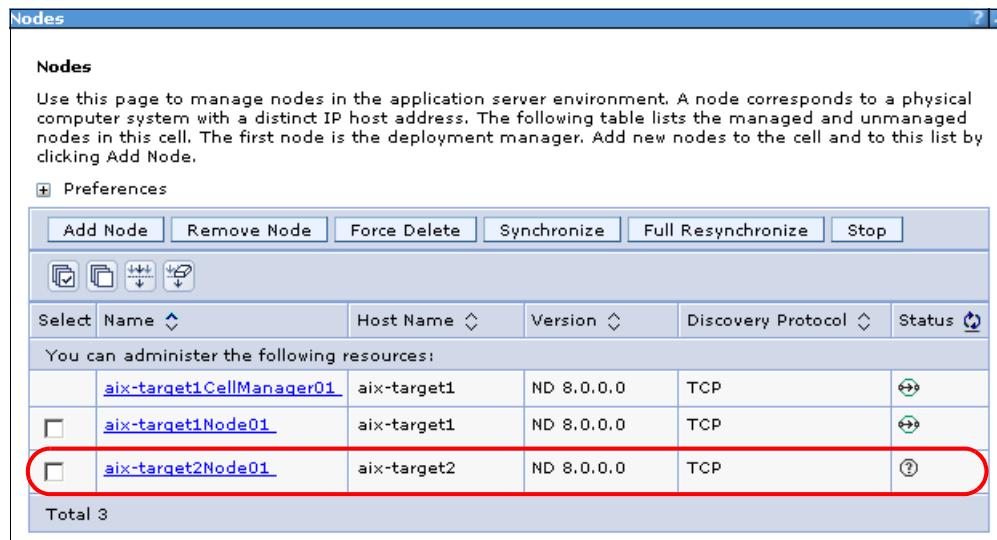


Figure 27-3 Unavailability of one of the nodes in the network manager cell

To accomplish rapid node recovery, complete the following steps:

1. Ensure that the existing damaged node is not running. Stop the node agent and any application servers that reside on the node. If the node cannot be stopped using standard commands, stop its processes. (Even if the node is not reporting to the deployment manager, it can still have running processes in the operating system.)

2. Remove the original, damaged profile, and create a profile to replace the damaged node. Use the same profile path, profile name, and node name as the unavailable node.

In this example, node aix-target2Node01 that has the profile name AppSrv8_02 stops functioning. To replace it with a new node, create an application server profile named AppSrv8_02 with node name aix-target2Node01, using the same directory path. This script creates a stand-alone server instance with the previous configuration names:

```
./manageprofiles.sh -create -profileName AppSrv8_02 -profilePath  
/opt/IBM/WebSphere/AppServer/profiles/AppSrv8_02 -templatePath  
/opt/IBM/WebSphere/AppServer/profileTemplates/default -nodeName  
aix-target2Node01 -hostName aix-target2 -adminUserName admin -adminPassword  
password -enableAdminSecurity true -samplesPassword admin
```

Note: You can also recreate the profile on a different computer from the original node if your original computer is unavailable, and if it was the cause of node unavailability, you can configure a new computer with the same host name.

During profile creation, you may receive the following message:

profileName: AppSrv8_02 is already in use. Specify another name

If so, run the following command:

```
./manageprofiles -delete -profileName AppSrv8_02
```

This command removes the profile from the WebSphere Application Server registry and retries the profile creation command. Because the profile will be partially or totally damaged, you can ignore error messages after running this command.

3. Run the **addNode** command with the **-asExistingNode** option from a command line at the bin directory of the recreated application server profile:

```
./addNode.sh dmgr_host dmgr_port -asExistingNode -username user_name -password  
password
```

After successfully running this command, you see the following log entry in the console:

```
ADMU0300I: The node aix-target2Node01 was successfully added to the  
aix-target1Cell101 cell
```

At this point, you are able to administer the node from the deployment manager console.

4. In the deployment manager console, synchronize all of the active nodes in the cell. Your node should be fully operational again.

27.3.2 Moving nodes to new environments

You can also use the **-asExistingNode** option to move the node to a different machine. The procedure is:

1. Delete the original profile without using the **removeNode** command.
2. On the deployment manager, change the host name of the node using the **AdminTask.changeHostName(...)** command.
3. Install WebSphere Application Server on the new machine at the same directory location.
4. Create a profile with the same node name and path.
5. Run the **addNode -asExistingNode** command.
6. Synchronize nodes in the network manager cell.

In the following example, three profiles are used:

Deployment manager profile This profile manages the source profile and will be used to manage the destination profile.

Source profile This profile will be migrated to a different destination. In this example, it is running on host aix-target2.

Destination profile This will be the new profile created on the source profile base. In this example, it is running on host aix-target3.

Figure 27-4 illustrates this procedure.

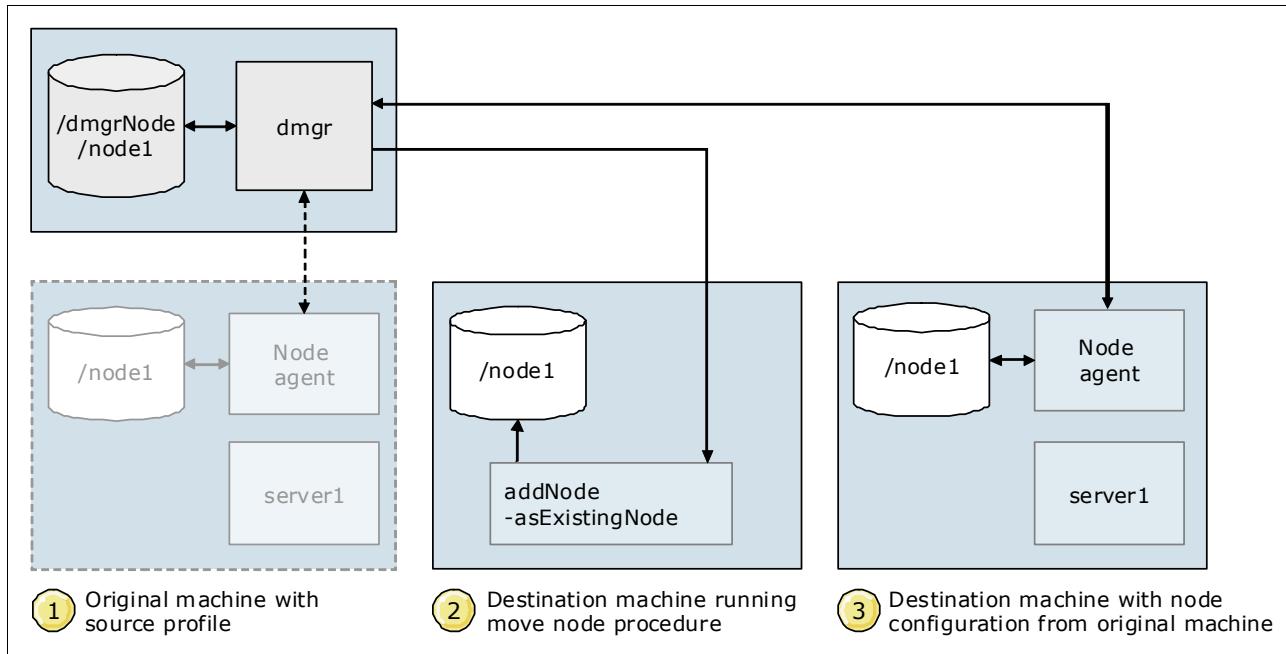


Figure 27-4 Moving a node to a destination machine

Note: In this procedure, we assume that the source and destination targets use the same WebSphere Application Server installation directory, profile name, profile path, and node name.

To move a node to a new environment, complete the following steps:

1. Ensure that the node agent or any application server processes are not running on the source profile, and back up the profile.
2. Change the host name of the source profile node within the master configuration present at the deployment manager:
 - a. Go to the deployment manager profile in the bin directory:
`profile_root/Dmgr01/bin`

- b. Run the **wsadmin** Jython or Jacl commands that change the host name of the node. The results in Example 27-5 assume that security is enabled and that the product requires that you enter a user name and password. For the new node host name, aix-target3 is used. It is the host of the new destination profile.

Example 27-5 Changing the node host name

```
./wsadmin.sh -lang jython -username admin -password admin
WASX7209I: Connected to process "dmgr" on node aix-target2CellManager01
using SOAP connector; The type of process is: DeploymentManager
WASX7031I: For help, enter: "print Help.help()"
wsadmin>AdminTask.changeHostName(['-hostName aix-target3 -nodeName
aix-target2Node01'])
'
wsadmin>AdminConfig.save()
'
wsadmin>quit
```

3. Log in to the destination profile target:

- a. Install WebSphere Application Server in a directory that has the same name as the product installation directory on the source profile target.
- b. Create a custom profile that has the same profile name, profile directory, and node name as the profile for the node that you want to move. When creating the custom profile, select to federate the node later. You can use the following command to create a destination profile from your WebSphere Application Server *install_root/bin* directory:

```
./manageprofiles.sh -create -profileName AppSrv8_01 -profilePath
/opt/IBM/WebSphere/AppServer/profiles/AppSrv8_01 -templatePath
/opt/IBM/WebSphere/AppServer/profileTemplates/managed -nodeName
aix-target2Node01 -hostName aix-target3 -federateLater true
```

Note that, in this command, the new target host name (aix-target3) is used together with the old node name (aix-target2Node01). This is because we just moved that node to a new location.

- c. Change your working directory to the newly created profile bin directory:

```
/opt/IBM/WebSphere/AppServer/profiles/AppSrv8_01/bin
```

- d. Run the **addNode** command with the **-asExistingNode** option to replace the application server node with the node that you want to move. The command that follows assumes that security is enabled and that the product requires you to enter a user name and password. For *dmgr_host* and *dmgr_port*, specify the host name and port number of the target deployment manager.

```
./addNode.sh dmgr_host dmgr_port -asExistingNode -username admin -password
password
```

Example 27-6 shows the result of running this command.

Example 27-6 Result of a successful node move to a different target

```
ADMU0116I: Tool information is being logged in file
/opt/IBM/WebSphere/AppServer/profiles/AppSrv8_01/logs/addNode.log
ADMU0128I: Starting tool with the AppSrv8_01 profile
CWPKI0308I: Adding signer alias "CN=saw116-sys1.itso.ral.ibm.com" to local
keystore "ClientDefaultTrustStore" with the following SHA digest:
56:34:3F:50:0F:84:53:74:F4:F3:54:8E:76:62:9F:ED:13:E0:F9:43
CWPKI0309I: All signers from remote keystore already exist in local keystore.
ADMU0001I: Begin federation of node aix-target2Node01 with Deployment Manager
```

```

at saw116-sys1:8879.
ADMU0009I: Successfully connected to Deployment Manager Server:
saw116-sys1:8879
ADMU0507I: No servers found in configuration under:
/opt/IBM/WebSphere/AppServer/profiles/AppSrv8_01/config/cells/aix-target3Node01Cell/
nodes/aix-target2Node01/servers
ADMU2010I: Stopping all server processes for node aix-target2Node01
ADMU0024I: Deleting the old backup directory.
ADMU0015I: Backing up the original cell repository.
ADMU0016I: Synchronizing configuration between node and cell.
ADMU0018I: Launching Node Agent process for node: aix-target2Node01
ADMU0020I: Reading configuration for Node Agent process: nodeagent
ADMU0022I: Node Agent launched. Waiting for initialization status.
ADMU0030I: Node Agent initialization completed successfully. Process id is:
467110
ADMU0300I: The node aix-target2Node01 was successfully added to the
aix-target2Cell01 cell.
ADMU0306I: Note:
ADMU0302I: Any cell-level documents from the standalone aix-target2Cell01
configuration have not been migrated to the new cell.
ADMU0307I: You might want to:
ADMU0303I: Update the configuration on the aix-target2Cell01 Deployment Manager
with values from the old cell-level documents.
ADMU0306I: Note:
ADMU0304I: Because -includeapps was not specified, applications installed on
the standalone node were not installed on the new cell.
ADMU0307I: You might want to:
ADMU0305I: Install applications onto the aix-target2Cell01 cell using wsadmin
$AdminApp or the Administrative Console.
ADMU0003I: Node aix-target2Node01 has been successfully federated.

```

- After successful node configuration and federation, you can use the administrative console of the target deployment manager or **wsadmin** to manage it (Figure 27-5). Consider updating it with any documents configured in the source profile at the cell-level and moving applications to the destination profile.

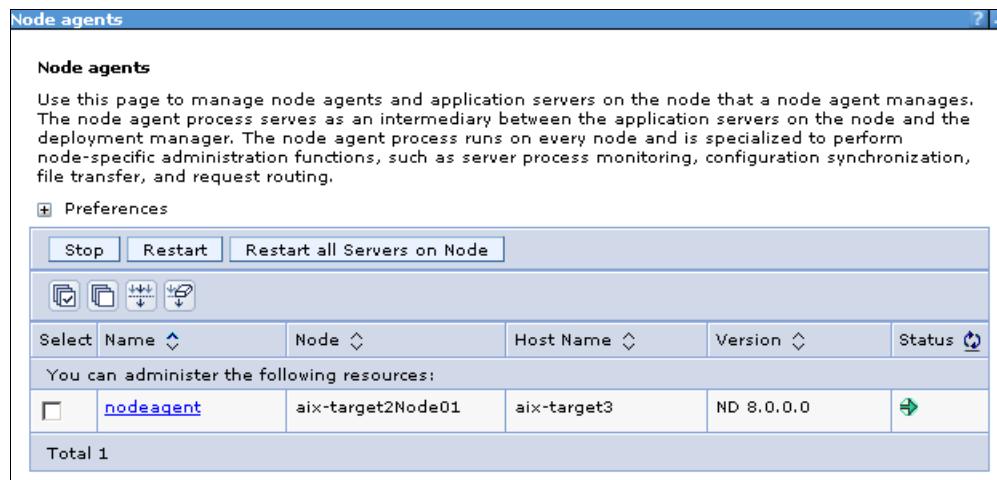


Figure 27-5 Node moved to new target aix-target3 - Administered from the deployment manager console

- Synchronize the nodes using the deployment manager console by clicking **System Administration** → **Nodes**, selecting the unsyncronized nodes, and clicking the **Synchronize** button.

6. If the moved node is fully operational, you can remove the source profile directory and its backup.

You can also move the node to a destination profile with a different path or a different operating system. However, moving the node to the z/OS platform is not supported. For more information, visit the Information Center at the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.nd.doc/info/ae/ae/tagt_addNode_asExistingNode.html

27.3.3 Recreating cells from a template

The WebSphere Application Server V8 **-asExistingNode** option of the **addNode** command can also be used to recreate a cell from deployment manager backup.

Note: This procedure requires a good knowledge of WebSphere Application Server configurations. You need to update several files generated by the server to migrate them to the new environment.

Assuming you have your deployment manager cell environment set up, complete the following steps:

1. Back up the deployment manager profile using the **backupConfig** command. Refer to 27.1.1, “Backing up a profile” on page 996 for more information.
2. Copy the backup file to a target where your deployment manager will reside.
3. On each new environment to be provisioned, install WebSphere Application Server and create a deployment manager profile or a managed node profile, depending on your environment topology.
4. Restore the new target deployment manager profile configuration using the **restoreConfig** command and customize each node configuration.

Important: Before running **restoreBackup**, consider updating backed up files with information from your new target profiles, such as:

- ▶ Port numbers
- ▶ Host names
- ▶ Virtual hosts
- ▶ Product installation paths
- ▶ Profile directories
- ▶ Security configuration

Configuration files that you might need to update include `wsadmin.properties`, `soap.client.props`, `variables.xml`, `security.xml`, `serverindex.xml`, `variables.xml`, and `virtualhosts.xml`.

For example, the following line from a `variables.xml` file might require updating if your target WebSphere Application Server product is installed in a different location than `/opt/IBM/WebSphere/AppServer`:

```
<entries xmi:id="VariableSubstitutionEntry_1310153920705"
symbolicName="WAS_INSTALL_ROOT" value="/opt/IBM/WebSphere/AppServer"
description="The filesystem path to the WebSphere product installation
directory."/>
```

5. Log in to each target on which you will use your nodes. From each node *profile_home/bin* directory, run:

```
./addNode.sh dmgr_host dmgr_port -asExistingNode -username user_name -password  
password
```

The *dmgr_host* and *dmgr_port* options are the new target deployment manager host and port. They replace the application server node with the node on the target cell, as described in 27.3.2, “Moving nodes to new environments” on page 1004.

6. Start the nodes and the servers.
7. Replace the Secure Sockets Layer certificates, if used.
8. Synchronize all of the nodes from the deployment manager.

Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this book.

IBM Redbooks

The following IBM Redbooks publications provide additional information about the topic in this document. Note that some publications referenced in this list might be available in softcopy only.

- ▶ *Getting Started with WebSphere Application Server Feature Pack for Service Component Architecture*, REDP-4633
- ▶ *IBM WebSphere Application Server V8 Concepts, Planning, and Design Guide*, SG24-7957
- ▶ *WebSphere Application Server V6 Scalability and Performance Handbook*, SG24-6392
- ▶ *WebSphere Application Server V6.1: System Management and Configuration*, SG24-7304 *WebSphere Application Server V8.0 Technical Overview*, REDP-4756
- ▶ *WebSphere Application Server V7.0 Security Guide*, SG24-7660
- ▶ *WebSphere Application Server V7 Messaging Administration Guide*, SG24-7770

You can search for, view, download or order these documents and other Redbooks, Redpapers, web docs, draft and additional materials, at the following website:

ibm.com/redbooks

Other publications

These publications are also relevant as further information sources:

- ▶ *TCAM Program Directory*, GI11-8919-01

Online resources

These websites are also relevant as further information sources:

- ▶ Apache Tuscany:
<http://tuscany.apache.org/>
- ▶ Command assistance and administrative scripting in WebSphere Application Server:
http://www.ibm.com/developerworks/websphere/library/techarticles/0812_rhodes/0812_rhodes.html
- ▶ Dynamic cache service:
http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=%2Fcom.ibm.websphere.nd.doc%2Finfo%2Fae%2Fae%2Fcontainer_dyn_admin.html

- ▶ Eclipse Equinox OSGi:
<http://eclipse.org/equinox>
- ▶ IBM AIX Toolbox for Linux Applications:
<http://www-03.ibm.com/systems/power/software/aix/linux/toolbox/date.html>
- ▶ IBM Fix Central:
<http://www.ibm.com/support/fixcentral/>
- ▶ IBM Installation Manager:
<https://www-304.ibm.com/support/docview.wss?uid=swg24029226#downloads>
- ▶ IBM Monitoring and Diagnostic tools for Java:
<http://www.ibm.com/developerworks/java/jdk/tools/>
- ▶ IBM Pattern Modelling and Analysis Tool for Java Garbage Collector (PMAT):
<http://www.alphaworks.ibm.com/tech/pmat>
- ▶ IBM Support Assistant:
<http://www.ibm.com/software/support/isa/>
- ▶ IBM Tivoli Composite Application Manager (ITCAM):
http://publib.boulder.ibm.com/infocenter/tivihelp/v24r1/index.jsp?topic=/com.ibm.itcamfad.doc_7101/ecam.html
- ▶ Java Diagnostics Guide:
<http://publib.boulder.ibm.com/infocenter/javasdk/v6r0/topic/com.ibm.java.doc.diagnostics.60/diag/welcome.html>
- ▶ Java Information Center:
http://publib.boulder.ibm.com/infocenter/javasdk/v6r0/index.jsp?topic=%2Fcom.ibm.java.doc.diagnostics.60%2Fdiag%2Funderstanding%2Fmemory_management.html
- ▶ Java Management Extensions (JMX) architecture:
http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.nd.multiplatform.doc/info/ae/ae/cxml_javamanagementx.html
- ▶ JDBC driver support and requirements:
http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=/com.ibm.websphere.nd.multiplatform.doc/info/ae/ae/udat_minreq.html
- ▶ OSGi applications:
<http://www.osgi.org/About/WhatIsOSG>
- ▶ Resource high availability:
http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=%2Fcom.ibm.websphere.nd.multiplatform.doc%2Finfo%2Fae%2Fae%2Fcdat_dsfailover.html
- ▶ Sample scripts for WebSphere Application Server V5 and V6:
<http://www.ibm.com/developerworks/websphere/library/samples/SampleScripts.html>
- ▶ Supported web servers:
<http://www.ibm.com/software/webservers/appserv/doc/latest/prereq.html>
- ▶ WebSphere Application Server V8 Information Center:
<http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp>

- ▶ WebSphere MQ queue manager:
http://publib.boulder.ibm.com/infocenter/wasinfo/fep/topic/com.ibm.websphere.nd.multiplatform.doc/info/ae/ae/tmj_wmq_miqm.html

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services

IBM



Redbooks

WebSphere Application Server V8: Administration and Configuration Guide

(1.5" spine)
1.5" <-> 1.998"
789 <-> 1051 pages



WebSphere Application Server V8: Administration and Configuration Guide

Learn about Websphere Application Server V8

This IBM Redbooks publication provides system administrators and developers with the knowledge to configure an IBM WebSphere Application Server Version 8 runtime environment, to package and deploy applications, and to perform ongoing management of the WebSphere environment.

Configure and administer a WebSphere system

As one in a series of IBM Redbooks publications and IBM Redpapers publications for V8, the entire series is designed to give you in-depth information about key WebSphere Application Server features. In this book, we provide a detailed exploration of the WebSphere Application Server V8 runtime administration process.

Deploy applications in a WebSphere environment

This book includes configuration and administration information for WebSphere Application Server V8 and WebSphere Application Server Network Deployment V8 on distributed platforms and WebSphere Application Server for z/OS V8.

The following publications are prerequisites for this book:

- ▶ *WebSphere Application Server V8.0 Technical Overview*, REDP-4756
- ▶ *IBM WebSphere Application Server V8 Concepts, Planning, and Design Guide*, SG24-7957

INTERNATIONAL TECHNICAL SUPPORT ORGANIZATION

BUILDING TECHNICAL INFORMATION BASED ON PRACTICAL EXPERIENCE

IBM Redbooks are developed by the IBM International Technical Support Organization. Experts from IBM, Customers and Partners from around the world create timely technical information based on realistic scenarios. Specific recommendations are provided to help you implement IT solutions more effectively in your environment.

For more information:
ibm.com/redbooks