



# **AppWorks™ Platform Administration Guide**

**Release 16.6**

**This documentation has been created for software version 16.6.**

It is also valid for subsequent software versions as long as no new document version is shipped with the product or is published at <https://knowledge.opentext.com>.

**Open Text Corporation**

275 Frank Tompa Drive, Waterloo, Ontario, Canada, N2L 0A1

Tel: +1-519-888-7111

Toll Free Canada/USA: 1-800-499-6544 | International: +800-4996-5440

Fax: +1-519-888-0677

Support: <https://support.opentext.com>

For more information, visit <http://www.opentext.com>

**Copyright © 2019 Open Text. All Rights Reserved.**

Trademarks owned by Open Text.

One or more patents may cover this product. For more information, please visit  
<https://www.opentext.com/patents>

**Disclaimer**

No Warranties and Limitation of Liability

Every effort has been made to ensure the accuracy of the features and techniques presented in this publication. However, Open Text Corporation and its affiliates accept no responsibility and offer no warranty whether expressed or implied, for the accuracy of this publication.

Last updated: 4/30/2019

---

# Table of Contents

<b>Part I AppWorks Platform Administration .....</b>	<b>39</b>
<b>    Chapter 1 Welcome to AppWorks Platform Administration .....</b>	<b>40</b>
<b>    Chapter 2 Managing organizations .....</b>	<b>42</b>
Creating an organization .....	42
Deleting an organization .....	43
Organization removal tool .....	44
<b>    Chapter 3 Managing security .....</b>	<b>47</b>
Managing authentication .....	48
Security Assertion Markup Language .....	49
Types of identity .....	55
Authentication mechanisms .....	66
Authentication through Single Sign On .....	68
Authentication plugins .....	77
Key store .....	83
AppWorks Platform password hashing mechanism through PBKDF2 .....	87
OpenText Directory Services .....	88
Synchronizing roles from AppWorks Platform to OTDS .....	101
Linking the partition to the resource .....	102
Managing users and group-membership for a specific organization .....	102
Consolidating the resource .....	103
Managing authorization .....	138
Configuring AppWorks Platform security policy .....	139
Managing an access control list .....	142
Digital certificates and trust stores .....	148
Managing a certificate .....	149
Application signing .....	156

Managing SAML trust .....	158
Using SSL in Platform connectors .....	159
Managing gateway with security options .....	162
<b>Chapter 4 Managing users and roles .....</b>	<b>163</b>
User Manager .....	164
Users - Roles view .....	165
Users - Tasks view .....	165
Managing users .....	166
User types .....	166
Working with User Manager .....	167
Creating users .....	168
Cloning users .....	170
Deleting users .....	171
Disabling users .....	172
Assigning roles to users .....	172
Revoking roles from users .....	173
Assigning tasks to users .....	173
Revoking tasks from users .....	174
Viewing the profile of a user .....	175
Changing the default organization of a user .....	175
Managing roles .....	176
Creating a role .....	177
Deleting roles .....	188
Assigning sub-roles .....	188
Revoking sub-roles from roles .....	190
Assigning tasks to roles .....	190
Revoking tasks from roles .....	191
Revoking users from roles .....	191
<b>Chapter 5 Managing system resources .....</b>	<b>195</b>
Managing service groups .....	196
Creating a service group .....	197
Modifying service group configuration .....	198
Modifying a monitor service group .....	199

Enabling payload validation .....	200
Disabling payload validation .....	200
Mapping service groups and web service interfaces .....	201
Managing service containers .....	203
Understanding service containers .....	204
Creating a service container .....	208
Modifying a service container .....	227
Configuring OS processes for a service container .....	228
Monitoring a service container .....	232
Pausing a service container .....	233
Managing connection points .....	234
Creating a connection point for a WebGateway instance .....	235
Modifying a connection point .....	235
Managing client connection point groups .....	236
Creating a client connection point group .....	236
Modifying a client connection point group .....	237
Deleting a client connection point group .....	237
Working with Web Service Interface Explorer .....	238
Validating Web service operations .....	239
Deleting a web service interface or operation .....	240
Viewing the implementation and interface of a Web service operation .....	240
Managing database configurations .....	241
Creating a database configuration .....	242
Modifying a database configuration .....	243
Deleting a database configuration .....	243
Viewing the database configuration usage report .....	244
Configuring service containers for reliable messaging .....	245
Configuring AppWorks Platform service containers for distributed transaction ..	246
Transaction configuration .....	247
<b>Chapter 6 Managing Web gateway .....</b>	<b>248</b>
Monitoring Web gateway .....	249
Configuring Web gateway timeout .....	251
Enabling payload validation at gateway (deprecated) .....	251

Enabling SOAP protocol validation at gateway .....	252
Enabling request validation at gateway .....	252
Enabling response validation at gateway .....	253
<b>Chapter 7 Managing applications .....</b>	<b>254</b>
Working with application deployer .....	255
Using application deployer .....	255
Deploying applications .....	256
Undeploying applications .....	259
Redeploying incomplete applications .....	260
Upgrading applications .....	260
AppWorks Platform application package .....	261
Managing application packages in the Organization space .....	262
Managing application packages across spaces .....	263
Dependencies between packages across spaces .....	267
Dealing with application package versions .....	268
Authorize organization administrator to manage packages .....	268
Runtime behavioral exceptions in the organization space .....	270
Using custom templates .....	273
Managing application packages in a primary distributed setup .....	275
Deploying application packages in a Primary-Distributed setup .....	275
Cluster summary view .....	276
Exceptions and recommendations .....	278
Customizing application installer wizard .....	280
Specifying custom page .....	280
Specifying operating system compatibility .....	281
Rolling back application packages .....	282
Installing an application in silent mode (deprecated) .....	282
Verifying applications .....	283
<b>Chapter 8 Managing licenses .....</b>	<b>286</b>
Viewing the usage report .....	287
Configuring the delivery mode of the usage report .....	288
Configuring AppWorks Platform to send usage reports .....	288
Viewing the master usage report .....	289

Sending the master usage report to the licensing server .....	290
Updating the license key automatically .....	290
Modifying the license key .....	290
Sending the usage report through proxy .....	291
<b>Chapter 9 Using composite application logging .....</b>	<b>293</b>
Viewing log messages using the Log Viewer .....	294
Log message .....	296
<b>Chapter 10 Auditing .....</b>	<b>298</b>
Database auditing .....	299
Configuring audit settings for a database .....	299
Viewing audit data for artifacts .....	301
Configuring audit for artifact types .....	304
Audit artifacts and actions .....	305
Archiving audit data .....	307
Installing the archiving scripts .....	307
Creating archive tables .....	308
Using the archive tables .....	309
Archiving criteria .....	310
Known limitations .....	312
Configuring audit filters for artifact types .....	313
<b>Chapter 11 Application server .....</b>	<b>316</b>
Managing Apache TomEE .....	316
Installing Apache TomEE .....	316
Adding servlet filters to Apache TomEE .....	317
Configuring Apache TomEE .....	317
Securing Apache TomEE .....	317
Monitoring Apache TomEE .....	318
Running service containers in Apache TomEE .....	318
Routing .....	318
JRE configuration .....	318
JMX connection .....	318
Logging .....	318
Configuring Apache TomEE .....	319

Running the configureapplicationserver script .....	319
Content expiration .....	320
Managing Apache TomEE .....	321
Log messages .....	321
JMX .....	321
Platform connector .....	322
Securing Apache TomEE .....	322
Hardening TomEE .....	322
Secure Sockets Layer .....	323
Windows authentication .....	323
Hardening the security of AppWorks Platform .....	323
<b>Chapter 12 Upgrading Java .....</b>	<b>327</b>
<b>Chapter 13 Setting themes at the organization level .....</b>	<b>329</b>
<b>Chapter 14 Deleting the runtime content .....</b>	<b>331</b>
Creating a signed-in machine user in AppWorks Platform .....	331
Creating archive tables .....	332
Creating the ArchivingDefinition .....	332
Registering the ArchivingDefinition .....	333
Executing the tool .....	334
Filter criteria .....	335
<b>Part II Business Activity Monitoring .....</b>	<b>341</b>
<b>Chapter 15 Business monitoring .....</b>	<b>342</b>
Key concepts .....	343
Business activity monitoring overview .....	345
Introduction .....	345
Key Features .....	345
Key Concepts .....	346
Business Activity Monitoring architecture .....	346
Architecture capabilities .....	347
Design architecture .....	348
Deployment architecture .....	349
Information architecture .....	351
Runtime architecture .....	353

<b>Chapter 16 Process Instance Manager .....</b>	<b>356</b>
Modifying monitoring and crash recovery settings .....	357
From deployed process models .....	357
Through ExecuteProcess Web service .....	357
Viewing process instances .....	358
Monitoring and managing process instances .....	359
Viewing a process instance in graphical view .....	360
Viewing activities of a process instance .....	362
Viewing subprocesses of a process instance .....	363
Performing operations in the Process Instance Manager .....	364
Viewing the message and error text of a process instance .....	367
Editing the message map data of a process instance .....	368
Skipping an activity .....	370
Adding a new process filter .....	371
Configuring the access to task parts from Process Instance Manager .....	371
Monitoring rules in the Process Instance Manager .....	373
Resetting a process instance from graphical mode .....	373
Restarting a process instance from graphical mode .....	374
<b>Chapter 17 Performing operations on deployed process models .....</b>	<b>376</b>
Setting authorization for deployed process models .....	378
<b>Chapter 18 Archiving process instances .....</b>	<b>380</b>
Defining an archive policy .....	381
Activating or deactivating an archive policy .....	381
Triggering an archive policy .....	382
Loading an archive .....	382
Viewing loaded archive details .....	383
<b>Chapter 19 Monitoring case instances .....</b>	<b>385</b>
Instances by case definition interface .....	386
<b>Chapter 20 Business Activity Monitoring administration .....</b>	<b>389</b>
Settings tab .....	389
Synchronize .....	389
Upload .....	390
Configure advanced settings .....	391

Trend Manager tab .....	391
Configuring advanced monitoring options .....	392
Disabling BAM synchronization .....	393
Enabling BAM synchronization .....	393
Uploading a business process .....	395
<b>Chapter 21 Managing and monitoring MDM .....</b>	<b>397</b>
MDM admin settings .....	398
Assigning MDM proxy user .....	399
Configuring MDM repository .....	399
Configuring conditions for receiving notifications .....	400
Configuring time period for storing logs .....	401
Downloading database scripts .....	401
Managing MDM models .....	402
Viewing the runtime MDM model in a dashboard .....	402
Performing runtime operations using an MDM model .....	412
Unpublishing an MDM model .....	418
Viewing MDM logs .....	419
MDM data steward cockpit .....	421
Monitoring MDM models .....	422
Viewing MDM logs .....	423
<b>Chapter 22 BAM standard dashboard .....</b>	<b>426</b>
<b>Part III JMX .....</b>	<b>433</b>
<b>Chapter 23 Managing operations through Java Management Extensions .....</b>	<b>434</b>
Applications .....	434
Management interface .....	434
<b>Chapter 24 JMX architecture .....</b>	<b>436</b>
<b>Chapter 25 Implementing Java Management Extensions in applications .....</b>	<b>438</b>
Alerts .....	439
Defining an alert .....	439
Issuing an alert .....	439
JMX counters .....	440
Types of counters .....	440
Using JMX counters .....	442

Operations .....	450
Problems .....	451
Defining a problem .....	451
Raising a problem .....	452
Resolving a problem .....	452
JMX settings for managed components .....	453
Setting types .....	453
Settings persistence .....	453
Defining settings .....	453
Legacy setting .....	456
InfoAttribute .....	457
<b>Chapter 26 Management interface .....</b>	<b>459</b>
Components .....	459
Settings .....	459
Info properties .....	459
Counters .....	460
Operations .....	461
Alerts .....	461
Problems .....	461
<b>Chapter 27 JConsole .....</b>	<b>463</b>
Connecting to JMX through JConsole .....	464
Obtaining the JMX address URL .....	465
Connecting to JMX using JConsole .....	466
<b>Chapter 28 MBeans .....</b>	<b>468</b>
Attributes .....	468
Operations .....	469
Notifications .....	469
<b>Chapter 29 Managed components .....</b>	<b>471</b>
Managed component types .....	472
Creating subcomponents .....	474
Creating WS-AppServer application components .....	475
Defining a managed component in an application connector .....	476
Defining a managed component in a Web application .....	477

AppConnector (Managed components) .....	478
Settings .....	478
Counters .....	478
Problems .....	478
Subcomponents .....	478
Auditing application connector (Managed components) .....	479
Settings .....	479
Counters .....	479
Alerts .....	479
Subcomponents .....	479
BAM business object dispatcher (Managed components) .....	479
Settings .....	479
Counters .....	480
Alerts .....	480
BAM configuration (Managed components) .....	481
Settings .....	481
Counters .....	485
Alerts .....	486
Problems .....	487
Subcomponents .....	487
BAM event dispatcher (Managed components) .....	487
Settings .....	487
Counters .....	488
Alerts .....	488
Branch cache .....	488
Settings .....	488
Counters .....	489
Alerts .....	489
Operations .....	489
Subcomponents .....	489
Bus gateway (Managed components) .....	490
Settings .....	490
Counters .....	490

Alerts .....	492
Problems .....	492
Subcomponents .....	492
Business Activity Monitoring service (Managed components) .....	492
Alerts .....	492
Problems .....	493
Business object cache .....	493
Settings .....	493
Counters .....	494
Operations .....	494
Alerts .....	494
Problems .....	494
Subcomponents .....	494
Business Process Engine (Managed components) .....	495
Settings .....	495
Operations .....	497
Counters .....	497
Alerts .....	499
Problems .....	499
Subcomponents .....	499
Business Process Management processor .....	500
Settings .....	500
Counters .....	500
Alerts .....	500
Problems .....	502
Subcomponents .....	502
CAP connector (Managed components) .....	503
Settings .....	503
Counters .....	503
Operations .....	503
Alerts .....	503
Problems .....	503
Subcomponents .....	503

Deployment (Managed components) .....	504
WebServiceOperations (Managed components) .....	506
Case Management Engine (Managed components) .....	506
Settings .....	506
Counters .....	507
Alerts .....	508
Problems .....	508
Subcomponents .....	508
CoBOCApConnector (Managed components) .....	508
Settings .....	508
Counters .....	508
Operations .....	508
Alerts .....	508
Problems .....	509
Subcomponents .....	510
CompressionSettings .....	510
Settings .....	510
Counters .....	510
Alerts .....	510
Subcomponents .....	511
CWS server (Managed components) .....	511
Settings .....	511
Counters .....	511
Operations .....	511
Alerts .....	511
Problems .....	511
Subcomponents .....	512
Data transformation service (Managed components) .....	512
Settings .....	512
Counters .....	512
Alerts .....	512
Problems .....	513
Subcomponents .....	514

DBConnectionPool (Managed components) .....	514
Settings .....	514
Counters .....	517
Alerts .....	518
Problems .....	518
Subcomponents .....	519
Decision table cache (Managed components) .....	519
Settings .....	519
Counters .....	519
Alerts .....	520
Problems .....	520
Subcomponents .....	520
Document cache .....	520
Settings .....	520
Counters .....	520
Alerts .....	521
Operations .....	521
Subcomponents .....	521
Document store connector .....	521
Settings .....	521
Counters .....	521
Alerts .....	522
Subcomponents .....	522
Download gateway (Managed components) .....	522
Settings .....	522
Counters .....	523
Alerts .....	524
Problems .....	525
Subcomponents .....	525
Email connector (Managed components) .....	525
Settings .....	525
Counters .....	526
Alerts .....	526

Problems .....	526
Subcomponents .....	526
Embedded data mapper .....	526
Settings .....	526
Counters .....	527
Alerts .....	527
Problems .....	528
Subcomponents .....	528
Event service (Managed components) .....	528
Settings .....	528
Counters .....	528
Alerts .....	529
Subcomponents .....	529
FTP connector (Managed components) .....	529
Settings .....	529
Counters .....	529
Alerts .....	529
Problems .....	529
Subcomponents .....	529
Gateway ThreadPool (Managed components) .....	529
Settings .....	529
Counters .....	530
GatewayWebApplication (Managed components) .....	530
Settings .....	530
Counters .....	530
Alerts .....	531
Problems .....	531
Subcomponents .....	531
Gossip (Managed components) .....	531
Settings .....	531
Info properties .....	534
Counters .....	535
Alerts .....	538

Subcomponents .....	538
HTTP connector (Managed components) .....	538
Settings .....	538
Counters .....	538
Operations .....	539
Alerts .....	539
Problems .....	540
Subcomponents .....	540
ISVP connector (Managed components) .....	540
Settings .....	540
Counters .....	540
Alerts .....	540
Problems .....	541
Subcomponents .....	541
LDAP cache (Managed components) .....	541
Settings .....	541
Counters .....	541
Alerts .....	543
Problems .....	543
Subcomponents .....	543
LDAP connector (Managed components) .....	543
Settings .....	543
Counters .....	543
Alerts .....	543
Problems .....	544
Subcomponents .....	544
LDAPConnection (Managed components) .....	544
Settings .....	544
Counters .....	544
Alerts .....	545
Problems .....	546
Subcomponents .....	546
License monitor (Managed components) .....	546

Settings .....	546
Counters .....	546
Alerts .....	546
Problems .....	548
Subcomponents .....	548
MDM hub publisher .....	548
Settings .....	548
Counters .....	548
Alerts .....	549
Operations .....	549
Problems .....	550
Subcomponents .....	550
Middleware wrapper (Managed components) .....	550
Settings .....	550
Counters .....	550
Alerts .....	551
Problems .....	551
Subcomponents .....	551
Monitor connector (Managed components) .....	551
Settings .....	551
Counters .....	552
Alerts .....	552
Problems .....	553
Subcomponents .....	553
Monitoring engine (Managed components) .....	553
Settings .....	553
Operations .....	554
Counters .....	554
Alerts .....	555
Problems .....	555
Subcomponents .....	555
Notification connector (Managed components) .....	555
Settings .....	555

Counters .....	558
Alerts .....	559
Problems .....	559
Subcomponents .....	559
Notification dispatcher .....	559
Settings .....	559
Counters .....	560
Alerts .....	561
Subcomponents .....	561
OS Process (Managed components) .....	561
Settings .....	561
Counters .....	561
Operations .....	563
Alerts .....	564
Problems .....	567
Subcomponents .....	567
PIM query connector connection pool .....	567
Settings .....	567
Counters .....	571
Alerts .....	573
Problems .....	573
Subcomponents .....	573
AppWorks Platform SAML protocol handler (Managed components) .....	574
Settings .....	574
Counters .....	574
Alerts .....	574
Problems .....	574
Subcomponents .....	574
Request monitor (Managed components) .....	575
Settings .....	575
Counters .....	575
Alerts .....	575
Problems .....	575

Subcomponents .....	575
Rule engine (Managed components) .....	576
Settings .....	576
Counters .....	577
Alerts .....	577
Operations .....	578
Subcomponents .....	578
Rule management .....	578
Settings .....	578
Counters .....	578
Alerts .....	578
Problems .....	578
Subcomponents .....	579
Scheduler engine (Managed components) .....	579
Settings .....	579
Counters .....	579
Alerts .....	579
Problems .....	579
Subcomponents .....	579
Scheduling .....	579
Settings .....	579
Counters .....	580
Alerts .....	580
Problems .....	580
Subcomponents .....	580
SearchConnector (Managed components) .....	580
Settings .....	580
Counters .....	580
Alerts .....	581
Problems .....	581
Subcomponents .....	581
Security administration (Managed components) .....	581
Settings .....	581

Counters .....	581
Alerts .....	581
Problems .....	582
Subcomponents .....	582
Service container (Managed components) .....	582
Processor - Settings .....	582
Info properties .....	583
Counters .....	583
Alerts .....	584
Problems .....	584
Subcomponents .....	584
Service container OS process (Managed components) .....	584
Settings .....	584
Counters .....	584
Alerts .....	584
Problems .....	584
Subcomponents .....	584
SOAP connector (Managed components) .....	585
Settings .....	585
Counters .....	585
Alerts .....	585
Problems .....	585
Subcomponents .....	585
SQLApp connector (Managed components) .....	586
Settings .....	586
Counters .....	586
Alerts .....	586
Problems .....	586
Subcomponents .....	586
SSO connector (Managed components) .....	586
Settings .....	586
Counters .....	587
Alerts .....	587

Problems .....	587
Subcomponents .....	587
State Syncup (Managed components) .....	587
Settings .....	587
Counters .....	587
Alerts .....	587
Subcomponents .....	588
Tag server (Managed components) .....	588
Settings .....	588
Counters .....	588
Alerts .....	588
Problems .....	588
Subcomponents .....	588
TaskIdentifierIndexer .....	589
Settings .....	589
Counters .....	589
Alerts .....	590
Subcomponents .....	590
UDDI connector (Managed components) .....	590
Settings .....	590
Counters .....	591
Alerts .....	592
Problems .....	592
Subcomponents .....	592
Upload gateway (Managed components) .....	592
Settings .....	592
Counters .....	592
Alerts .....	594
Problems .....	594
Subcomponents .....	594
UriBinding cache (Managed components) .....	595
Settings .....	595
Counters .....	595

Alerts .....	596
Problems .....	596
Subcomponents .....	596
Web application (Managed components) .....	596
Settings .....	596
Counters .....	596
Alerts .....	596
Problems .....	597
Subcomponents .....	597
WebApplication (Managed components) .....	597
Settings .....	597
Counters .....	597
Alerts .....	597
Problems .....	597
Subcomponents .....	598
WebGateway (Managed components) .....	598
Settings .....	598
Counters .....	598
asynchronousRequest .....	598
requestHandlingDuration .....	598
synchronousRequest .....	598
Alerts .....	598
Problems .....	599
Subcomponents .....	599
WebLogger connector .....	599
Settings .....	599
Counters .....	599
Alerts .....	599
Subcomponents .....	599
WS-AppServer (Managed components) .....	600
Settings .....	600
Counters .....	601
Alerts .....	601

Problems .....	602
Subcomponents .....	602
WS-AppServer SOAP Client .....	602
Settings .....	602
Counters .....	603
Alerts .....	603
Problems .....	603
Subcomponents .....	603
WSDLGateway Web application (Managed components) .....	603
Settings .....	603
Counters .....	603
Alerts .....	604
Problems .....	604
Subcomponents .....	604
XDS Synchup (Managed components) .....	604
Settings .....	604
Counters .....	605
Alerts .....	605
Subcomponents .....	605
XDSRepository (Managed components) .....	605
Settings .....	605
Counters .....	605
Alerts .....	605
Problems .....	605
Subcomponents .....	606
XForms application connector (Managed components) .....	606
Settings .....	606
Counters .....	607
Alerts .....	607
Problems .....	607
Subcomponents .....	607
XMLStore connector (Managed components) .....	607
Settings .....	607

Counters .....	608
Alerts .....	608
Problems .....	608
Subcomponents .....	608
<b>Chapter 30 Monitoring managed components .....</b>	<b>609</b>
Changing managed component settings .....	609
Invoking managed component operations .....	612
Monitoring managed component counters .....	615
Monitoring heap memory usage of service containers .....	618
Receiving managed component notifications .....	620
<b>Chapter 31 Monitoring and maintenance .....</b>	<b>622</b>
JMX attributes for authentication .....	622
JMX attributes in a single sign-on .....	623
JMS Connector Configuration interface .....	624
General .....	624
Triggers .....	625
Destination Managers .....	627
JMX counter statistics .....	630
Value counters .....	630
Event Occurrence counters .....	630
Event Value counters .....	631
<b>Part IV Utilities .....</b>	<b>633</b>
<b>Chapter 32 Utilities .....</b>	<b>634</b>
<b>Chapter 33 Management console .....</b>	<b>636</b>
Managing LDAP content .....	638
Adding an entry to LDAP .....	638
Diagnosing LDAP entries .....	639
Managing LDAP entries .....	639
Accessing Management Console .....	640
Administration recovery .....	641
Enabling AppWorks Platform to run in standalone or network mode .....	642
Managing AppWorks Platform properties .....	644
Managing service containers using the management console .....	644

Modifying AppWorks Platform license information .....	645
Platform properties .....	647
Using repository browser .....	650
Using state syncup to synchronize instances of AppWorks Platform .....	650
Viewing roles and ACLs .....	651
Global configuration properties for AppWorks Platform .....	652
LDAP related properties .....	652
Management Console properties .....	655
Authentication and single sign-on properties .....	655
XML parser (NOM) properties .....	661
UDDI connector properties .....	663
Web gateway .....	664
General properties .....	666
License properties .....	675
Port ranges within AppWorks Platform .....	676
<b>Chapter 34 Application Package Deployment utility .....</b>	<b>678</b>
List of Ant tasks .....	678
Configuring the server to use the Ant tasks .....	679
Sample build file .....	680
cloneservicecontainers - Ant task .....	681
Prerequisites .....	681
Sample usage .....	682
Parameters .....	682
Parameters as nested elements .....	682
searchCriteria .....	683
Supported search patterns .....	684
createconnectionpoint - Ant task .....	685
Sample usage .....	685
Parameters .....	685
createdatabaseconfiguration - Ant task .....	686
Parameters .....	686
Parameters for database and driver .....	687
createservicegroup - Ant task .....	689

Sample usage .....	689
Parameters .....	690
Parameters for WebserviceInterface .....	691
createservicecontainer - Ant task .....	691
Sample usage .....	692
Parameters .....	692
Parameters as nested elements .....	693
configurationFile example .....	694
deleteconnectionpoint - Ant task .....	695
Sample usage .....	695
Parameters .....	695
deletedatabaseconfiguration - Ant task .....	695
Sample usage .....	696
Parameters .....	696
deleteservicecontainer - Ant task .....	696
Sample usage .....	696
Parameters .....	697
deleteservicegroup - Ant task .....	697
Sample usage .....	697
Parameters .....	697
deployapplicationpackage - Ant task .....	698
Sample usage .....	698
Parameters .....	699
Sample user input file .....	701
undeployapplicationpackage - Ant task .....	701
Sample usage .....	702
Parameters .....	702
<b>Chapter 35 CoBOC browser .....</b>	<b>703</b>
Creating a business object .....	703
Managing a business object .....	704
<b>Chapter 36 Signing packages .....</b>	<b>705</b>
Application signing on Windows .....	705
Sign a package using a graphical user interface tool .....	706

Sign a package by running a batch file from the command prompt .....	707
Sign a package using a Java command .....	708
<b>Application signing on Linux .....</b>	<b>709</b>
Sign a package through a graphical user interface tool .....	709
Sign a package by running a batch file from the command prompt .....	710
Sign a package using a Java command .....	711
<b>Loading a signed package .....</b>	<b>712</b>
During AppWorks Platform installation .....	712
After AppWorks Platform installation .....	713
<b>Chapter 37 Formatting XML data .....</b>	<b>714</b>
<b>Chapter 38 HealthCheckURL .....</b>	<b>715</b>
<b>Chapter 39 LDAP Explorer .....</b>	<b>717</b>
Modifying single-value attributes of LDAP objects .....	717
Modifying multi-value attributes of LDAP objects .....	718
Modifying XML File Attributes of LDAP Objects .....	718
<b>Chapter 40 XMLStore Explorer .....</b>	<b>720</b>
Adding Items to XMLStore .....	721
Adding Folders to XMLStore .....	721
Deleting items from the XMLStore .....	722
Modifying items in the XMLStore .....	722
Renaming items in the XMLStore .....	723
<b>Part V How Tos .....</b>	<b>725</b>
<b>Chapter 41 How Tos .....</b>	<b>727</b>
<b>Chapter 42 Administration .....</b>	<b>728</b>
Configuring AppWorks Platform in DMZ .....	729
Configuring an active directory .....	729
Configuring key store and trust store manually for LDAP service container .....	732
Configuring single sign-on for failover .....	734
Configuring virtual memory for XML NOM documents .....	734
Changing the password for the key store and trust store of the LDAP client connection .....	736
Creating a plug-in for a content management system .....	737
Enabling event logging at the system and individual component level .....	739
Enabling SSL communication .....	740

Managing chain SOAP requests .....	740
Access restriction to public SOAP operations .....	741
Running the AppWorks Platform (<instance name>) in different user context in Linux .....	742
Setting up start-up applications in the AppWorks Platform desktop .....	742
Loading an application with system environment variables content .....	743
Restarting AppWorks Platform monitor .....	744
Starting or stopping AppWorks Platform monitor from the command prompt .....	744
Starting the service containers from the command prompt .....	745
Using Windows authentication to access SQL server .....	746
Removing stacktrace details from SOAP responses .....	748
gateway.fault.stripstacktrace .....	748
gateway.fault.blacklist .....	748
Changing the location of the upload and download folders .....	750
Modify wcp.properties .....	750
Set the appropriate permissions .....	750
Document store .....	751
Creating and modifying document store repository .....	753
Configuring document store connector with Apache CXF library .....	765
Configuring access URLs .....	766
Configuring an individual node .....	767
Configuring an internal load balancer .....	768
Configuring a public reverse proxy .....	768
Configuring Content Server with AppWorks Platform .....	768
Deploying the WAR file .....	769
Creating the AppWorks Platform resource in OTDS .....	769
Enabling impersonation for AppWorks Platform resources in OTDS .....	770
Creating AppWorks Platform users in OTDS .....	770
Activating AppWorks Platform resources .....	771
Adding the Content Server resource to AppWorks Platform .....	772
Performing a cleanup before creating the document store service container .....	772
Creating the document store service container .....	773
Granting access to the root folder and its subfolders .....	774
Configuring HTTP connector .....	774

Configuring the JMS connector .....	776
GUI configuration .....	778
Service container configuration .....	778
Configuring the E-mail connector .....	779
Creating a database configuration .....	779
Configuring the E-mail connector .....	780
Configuring the public URL path prefix .....	789
Defining a custom renderer class to log messages .....	790
Implementing projects combining Capture and AppWorks Platform .....	791
Providing log configuration details .....	795
Enabling logging for gateway .....	797
Customizing logging .....	798
<b>Chapter 43 Business Activity Monitoring (How Tos) .....</b>	<b>799</b>
Developing a dashboard .....	799
Deciding monitoring content .....	800
Identification of data sources .....	800
Process monitoring object for attributes to be monitored .....	801
Business measures for graphs and KPIs .....	801
Drilldown on dashboards .....	802
Email action aspects .....	803
Developing a Business Event Response .....	803
Developing a KPI .....	805
Developing an Enterprise Data Object in BAM .....	805
<b>Chapter 44 Extended ECM (How Tos) .....</b>	<b>807</b>
Business case .....	807
Roles involved .....	807
Scenarios .....	808
Extended ECM overview .....	808
Key concepts of xECM .....	809
Features of xECM integration with AppWorks Platform .....	809
Scenarios for business workspace creation .....	811
Creating a business workspace automatically .....	811
Creating a business workspace manually .....	814

Early creation of business workspace .....	815
Late creation of business workspace .....	818
Developing application in AppWorks Platform .....	821
Configuring AppWorks Platform in Content Server .....	825
Configuration in AppWorks Platform TomEE .....	825
Configuration in Content Server .....	825
Creating a category .....	827
Creating a classification .....	829
Creating a workspace type .....	830
Configuring document templates .....	836
Opening a document template .....	837
Configuring connections to external systems .....	838
Configuring business object types .....	839
Defining relations between business objects .....	842
Parent to Child relation .....	843
Peer relation .....	848
Reflexive relation .....	853
Cross application business workspace for multiple business objects .....	855
Shared business workspace .....	855
Related business workspace .....	858
Configuring a link to an existing business workspace .....	861
<b>Part VI Reference .....</b>	<b>865</b>
<b>Chapter 45 Reference .....</b>	<b>867</b>
Administration .....	868
Alert repository for AppWorks Platform .....	868
Categories and severity levels .....	907
Appenders for an alert system .....	910
Managing logging framework .....	922
AppWorks Platform user privileges .....	941
Change password for key store and trust store .....	941
Configuring file system permissions for CWS folders on Linux .....	942
Creating a configuration profile for an FTP server .....	945
Removing header details from SOAP responses .....	946

Restrict access to public SOAP methods .....	947
Starting a batch of service containers .....	947
User authentication .....	949
JDBC details interface .....	949
Process Engine .....	951
Memory status interface .....	951
New toolbar item configuration interface .....	952
Object tree settings .....	953
Operational-level views .....	954
Permitted activities in a transaction .....	955
Overview of integration with WS-AppServer .....	956
Prerequisites for installing applications .....	957
Accessing Web services using the Web services explorer .....	957
Runtime architecture .....	957
Removing a certificate from a trust relation .....	959
Reloading the database metadata .....	960
Replacing tags in a query .....	960
ACL types .....	961
Using APIs for database access .....	961
Viewing memory status details .....	966
Sending and receiving SOAP messages using Simple Client .....	967
Add new registry details interface .....	968
Advanced search options in Log Viewer .....	969
Result Messages interface .....	970
Private-public key pair .....	970
Application package loading interface .....	970
AppWorks Platform Web gateway .....	972
Publishing event logs .....	973
Audit Viewer interface .....	973
Base class .....	974
Relationships between objects .....	974
Methods involving relationships .....	975
Business Object Lifecycle tab .....	977

Business Object tab .....	978
Certificate types .....	980
Changing access rights of the user .....	981
Changing access rights of users .....	981
Rule Repository service interface .....	982
CoBOC .....	983
Business Objects .....	983
CoBOC - A Common Repository for Collaborative Business Objects .....	984
Key Features of CoBOC .....	984
Deploying CoBOC .....	984
CoBOC Transaction Monitor .....	985
CoBOC Transaction Monitor interface .....	985
Template interface .....	986
Configuration option for SOAP message .....	988
Configuring AppWorks Platform security .....	988
Running multiple service containers in a single JVM .....	988
Configuring certification authorities in the trust store .....	989
Configuring security settings .....	989
Configuring security settings for external Web services .....	990
Configuring the XForms service container .....	990
Configuring trusted certificates for WS-security .....	992
Contingency management .....	993
License key expiration .....	994
Invalid usage report .....	994
Internet failure .....	994
Master computer becomes unresponsive .....	995
Create Class from Object Layout interface .....	996
Creating custom classes .....	996
Creating nested classes .....	997
Adding attributes to classes .....	997
Creating a sample key store .....	1000
Creating a sample trust store .....	1000
Creating a schema fragment .....	1001

Creating an email template .....	1002
Custom settings .....	1003
Custom views .....	1003
Database configuration for applications .....	1003
Deleting a certificate .....	1004
Deleting a trust relation .....	1004
Deploying applications .....	1004
Deployment Descriptor interface .....	1005
Diagnostic options for LDAP entries .....	1006
Editing a custom application connector .....	1007
Editing role details .....	1008
Editing user details .....	1009
Email modeler .....	1009
Embedding Inbox as a URL .....	1010
Sample URLs .....	1012
Enabling Composite Application Logging .....	1013
Enterprise Data Object .....	1013
Extended key usage options .....	1014
Extension class .....	1014
Gateway Statistics .....	1014
HTTP connector features .....	1015
Multiple server support .....	1015
Organization-aware .....	1016
Standard web services and REST services support .....	1016
Multiple payload format .....	1017
Request and response transformation capability .....	1018
Authentication support .....	1018
Custom HTTP header support .....	1018
URI parameters mapping .....	1019
Proxy support .....	1020
JMX counters .....	1020
HTTP connector service contract .....	1020
Identity .....	1028

Inbox control .....	1028
Schema .....	1028
Properties .....	1031
Methods .....	1031
Installation role in licensing .....	1032
Installation types .....	1034
Instance-level views .....	1034
Instance-specific properties .....	1035
Instances by Process Definition interface .....	1037
Intranet user authentication .....	1040
Java Archive Definition interface .....	1040
Key usage options .....	1044
Keyboard Shortcuts in XPath Editor .....	1045
KPI composite control properties interface .....	1045
LDAP cache statistics .....	1049
LDAP logon interface .....	1049
Leveraging XQY extensions in the WS-AppServer .....	1050
License configuration interface .....	1055
Lightweight Directory Access Protocol .....	1056
Loading a certificate .....	1056
Localizing an email model .....	1056
Manage case instances interface .....	1057
MDM data entity interface .....	1060
MDM data store interface .....	1066
MDM default sniffer fields .....	1067
MDM-specific fields .....	1069
Memory status interface .....	1072
Message debugger .....	1072
Message Queue sniffer example .....	1072
Configuring the JNDI Parameters tab .....	1073
Configuring the Receiver Details tab .....	1073
Message repository .....	1074
Method properties interface .....	1075

Monitoring and crash recovery interface .....	1083
Monitoring user activity in AppWorks Platform .....	1085
Naming rule actions .....	1085
New entity configuration interface .....	1085
New service configuration interface .....	1086
New toolbar configuration interface .....	1087
Routing Web service requests .....	1087
Routing logic .....	1088
Routing optimization to keep requests inside an OS Process or TomEE .....	1089
Schedule interface fields .....	1089
Searching for Web services in Web service Interface Explorer .....	1090
Security settings and options .....	1091
SendMail .....	1092
Sequence Generator Properties interface .....	1094
Task example .....	1096
tModel Details interface .....	1096
tModel information .....	1096
Category bag .....	1097
Identifier bag .....	1097
Universal Discovery Description Integration (UDDI) .....	1097
UDDI Explorer interface .....	1097
UDDI Explorer tool .....	1098
Unconditional ACL .....	1098
Updating the WS-AppServer package with database details .....	1100
Upload and download properties interface .....	1101
Using a digital certificate .....	1102
Viewing access permissions granted for a role .....	1104
Viewing log messages at service container level .....	1104
Viewing log messages using Apache Chainsaw .....	1105
Web gateway statistics interface .....	1106
Web service operation interface .....	1106
Web service properties interface .....	1109
Working with State SyncUp .....	1111

Overview of state SyncUp .....	1112
State SyncUp architecture .....	1113
State SyncUp features .....	1113
State Syncup properties .....	1114
Components of a default cluster .....	1114
Modifying the SyncUp configuration of monitor service group .....	1115
Synchronizing service containers .....	1116



---

## Part I

# **AppWorks Platform Administration**

# Chapter 1

# Welcome to AppWorks Platform Administration

This guide is for administrators who manage, monitor, and secure AppWorks Platform resources.

During the development phase, you create the artifacts and build them into an application. After deploying the application, it is the responsibility of an administrator to manage these artifacts by performing the following tasks:

- Deploying the Application: The design time content is packaged and distributed as an application in the deployment environment. You must install the application to start using the content for your business needs. AppWorks Platform not only provides application development environment , it also serves as a framework for other Independent Software Vendors (ISVs) to run and develop applications. The section contains the procedures to deploy Applications and manage Java Management Extensions.
- Creating users: To enable users to work on AppWorks Platform, you must create users and configure appropriate security settings called as Access Control Lists (ACL). Depending on the roles associated with the users, different functions of applications are available. Any AppWorks Platform user requires a set of roles to access certain resources. See [Managing users and roles](#) to manage users, roles and tasks.
- Creating Organizations: Within AppWorks Platform environment, organizations are the primary level of grouping users and system resources such as Service Group, Service Container, Connection point. Users are bound to operate within an organization, hence it is essential to create the organizations. The section describes how to add, edit, and delete Organizations.
- Customizing the style of an Organization: AppWorks Platform provides the Organization Level Theme which helps you create an organization with your own defined style standards. See [Setting themes at the organization level](#).
- Managing Master Data: While developing applications, you can model the flow of data from the spoke to the hub and hub to the spoke. After running the Master Data Management (MDM) design-time model, monitoring activities such as uploading data and viewing MDM logs are performed by the administrator. See [Managing MDM models](#).
- Providing connectivity to the backend: AppWorks Platform enables you to retrieve data from various applications and backend databases, and use it in your organization. To

establish connectivity with the backend, create various system level resources such as service groups, service containers, and connection points. These play a vital role in administering AppWorks Platform. See [Managing system resources](#) for detailed procedures to manage these system resources.

- Monitoring the gateway: To track the back and forth movement of SOAP messages between the current application environment and the external applications, you must configure the Web gateway on a Web server. See [Managing Web gateway](#).
- Licensing: As an administrator, you must monitor the usage of AppWorks Platform and renew the license keys periodically. When the AppWorks Platform is installed, you get a license for 60 days in operational mode. The evaluation mode is 30 days. Before the 60 days expire, you must send a License report to AppWorks Platform. You can do this automatically. This license report gives an overview of the number of unique users for each application. By managing the licenses, you can monitor the usage reports and license keys that are necessary to track the usage of AppWorks Platform. See [Managing licenses](#).
- Logging: During the request flow, you may have to analyze the error state of a transaction or an activity and fix it. [Using composite application logging](#) (CAL) helps you debug a particular application for information that helps you diagnose or know information on how or what a Web service is running.
- Securing the resources: All organizations have resources that are critical to its business or operations and so need to be protected. The primary goal of implementing security in an organization is confidentiality and safety of its resources. Security is increasingly becoming an important aspect of any software product. In AppWorks Platform, you can manage security using various security options that can be incorporated at various stages of application development and administration. See [Managing security](#).  
AppWorks Platform supports task-level authorization that is based on the ACLs associated with users and roles.
- Auditing the artifacts: AppWorks Platform provides auditing features that help you monitor the changes made to artifacts and business processes. See [Auditing](#).
- Deleting runtime content: Using a content deletion tool, you can delete runtime content from the Business Process Management, Case Management, Notification, and Task components. See [Deleting the runtime content](#).

# Chapter 2

# Managing organizations

An organization is a logical grouping of users, artifacts, and system resources. Artifacts can include roles, tasks, and system resources can include service groups, service containers and so on. Each organization contains users specific to that organization and their context of operation is bound to that organization.

When you install AppWorks Platform, a default organization is automatically created. This organization is referred to as System organization. You can add users and roles to this default organization or create organizations within it. The System organization is meant for platform specific administrative operations such as creating an OS process, Security Administration acitivities. For other business needs such as developing an application or working with an application, you must create an organization and work within it. To create an organization, you must have the role of a System Administrator.

To manage organizations, click  (Organization Manager).

You can perform following activities to manage organizations:

- [Creating an organization](#)
- [Deleting an organization](#)
- [Organization removal tool](#)

## Creating an organization

An Organization serves as a container where you can create and manage entities required to work with AppWorks Platform. These entities can be users, functional components, and user interfaces. An Organization provides a unique space to develop your custom application. Organizations can be treated as virtual separators of various business divisions within an enterprise.

### Before you begin

You must have the role of a systemAdmin to create an organization.

### To create an organization:

1. On the Welcome page, click  (Organization Manager).  
The Organization Manager window opens and displays the existing organizations in a tabular format.

2. Click . A row is added to the table.
3. Type a unique and a detailed name for the organization in the **Name** and the **Full name** fields.

**Note:** The full name of an organization is editable only after the organization is created.

4. In the **Select Administrators** pane, select the users from the **Users** list and click **Add**.

**Note:** The **Users** list displays the names of all the users, from all the available Organizations. The selected users are added to the **Administrators** list and these users become the Organizational Administrators of the organization being created.

5. Click .

An organization is created.

When you are using OTDS, it is recommended to follow the steps described in [Configuring OTDS for AppWorks Platform](#) to fill the organization with users and have single sign on enabled between OpenText products.

## Deleting an organization

Before you begin, you must be assigned the systemAdmin role to delete an organization.

There are two ways to delete an organization:

- Delete an organization only from LDAP using the *Organization Manager*.
- Delete an organization and all its content using a command-line tool.

**Note:**

- You cannot use the command-line tool to delete an organization, if you delete the organization from LDAP using the *Organization Manager*.
- You cannot delete the System organization as it is a part of the AppWorks Platform installation and serves as the base for all the administrative activities.
- To delete the System organization, you must uninstall AppWorks Platform.

### To delete an organization only from LDAP using Organization Manager

1. Navigate to the Welcome page and click (Organization Manager). The *Organization Manager* page appears displaying the existing organizations in a tabular format.
2. Select the check box for the organization you want to delete.
3. Click . The Confirm dialog box opens. Click Yes to confirm the deletion of the organization.

**Note:** You can delete multiple organizations by selecting several organizations at a time.

### To delete an organization and all its content using a command-line tool

Deleting an organization from the Organization Manager will only delete its corresponding service groups and the organizational users from LDAP.

It will not delete the content available in the organization. To delete an organization and all its content from AppWorks Platform, you can use the *Organization removal tool* from the command prompt. See the [Organization removal tool](#) tool page for more information.

## Organization removal tool

The organization removal tool is a command-line tool, which deletes an organization and removes all the organization-specific data from the system. The data deleted includes entries from:

- LDAP
- Databases
- File system
- Document store repository

### To run the tool:

1. Install the Cordys Organization Removal Tool package.  
You can install this package using the Application Deployer. See [Deploying applications](#).
2. Ensure that the user executing the tool from the command prompt or a terminal client has the following:
  - User must be present either in the LDAP as an AppWorks Platform user or the OS identity of one of the existing users must be mapped to this user.
  - User must be assigned the System Administrator role and must also be the Administrator of the organization that is being deleted.
3. Set the CORDYS\_HOME environment variable in the command shell or environment to the folder where AppWorks Platform is installed.
  - On Windows, set `CORDYS_HOME=<AppWorks_Platform_installdir>`.
  - On Linux or Unix, set `CORDYS_HOME=<AppWorks_Platform_installdir>`.
4. On Windows, open the command prompt or terminal client and change the current working folder to `<AppWorks_Platform_installdir>/components/organizationremovaltool`.
  - On Linux, grant the user read and execute permissions for the `organizationremovaltool.sh` script file.

```
chmod 511 organizationremovaltool.sh
```

5. Execute the batch or script file with the name of the organization to be deleted as the parameter.

- For Windows

```
organizationremovaltool.cmd <name of the organization to be deleted>
```

- For Linux

```
./organizationremovaltool.sh <name of the organization to be deleted>
```

6. The tool triggers the deletion action. You can view the log messages in a file with the name of the organization in the <AppWorks Platform\_installdir>/Logs folder. By default, all the messages of type Info are logged in the file and the error messages are logged to console. The log configuration for the organization deletion is available in the **Log4jConfiguration.xml** file in the <AppWorks Platform\_installdir>/components/organizationremovaltool/configuration folder.
7. Restart the AppWorks Platform (<instance name>). If the tool is run for multiple times to delete multiple organizations, it is recommended to restart the monitor after deleting all the organizations.
8. Repeat from step 3 on all the computers in a cluster setup.

**Note:**

- Before running the tool, you must disable the *request validation* option if it is enabled and restart the Monitor service.
- It is not possible to delete the *system* organization.
- The organization data cannot be retrieved again once it is deleted.
- If an error occurs while executing the tool, the tool must be executed again after taking the steps to correct the cause of the error.
- If an error occurs while executing the tool, the data will be deleted till the point of the error.
- It is not possible to run this tool simultaneously on multiple computers in a cluster or delete multiple organizations at a time within a single computer.
- In Windows, the organization deletion tool will not delete the jar files from the current organization in any one of the following situations:
  - if they are specified in the **system classpath**.
  - if they are consumed by the service containers that are created outside the organization.
- In an MDM Model, if the **Enable Business Object Lifecycle**, **Enable Registry**, or **Enable Data Quality** options are enabled for any of the hub data entities, then the

corresponding tables -- **State Instance Table Name**, **Registry Table Name**, and **Match Object Table Name** and their content are not removed when the MDM Model is unpublished as these tables contain application data.

These tables are not removed if the organization removal tool is executed after any MDM Model is unpublished either from the MDM Cockpit or by undeploying any application package that contains an MDM model.

# Chapter 3

# Managing security

All organizations have resources that are critical to its business or operations that need to be protected. The primary goal of implementing security in an organization is confidentiality and safety of its resources.

The applications built on AppWorks Platform are provided with the following options that ensure security of the application:

- Authentication
- Authorization
- Digital certificates
- Web gateway configured with security options

## **Authentication**

Authentication is the process of establishing or confirming the credentials entered by a user. In the context of AppWorks Platform usage, this involves confirming the identity of the user. This section of documentation describes the various ways in which authentication is performed in AppWorks Platform. For information on managing authentication, see [Managing authentication](#).

The Authenticators tab in the Security Administration task is used to configure custom or external authentication mechanisms.

For more information on Configuring SAML 2.0 authentication, see [Configuring SAML 2.0 Authenticator](#) and [Managing SAML trust](#).

## **Authorization**

You can manage authorization by regulating the Access Control Lists (ACL) at various levels such as users, namespaces, service groups, metadata, XML Store Objects, and LDAP object. For information on managing authorization, see [Managing authorization](#).

## **Digital certificates**

Security Management provides you with an interface for managing security. The Certificates tab of Security Administration is used to manage root and intermediate certificates. The Code Signing tab is associated with signing applications and the SAML Trust tab to configure trust for SAML-based WS-Security authentication.

For an overview of digital certificates and trust stores, see [Digital certificates and trust stores](#).

## Web gateway configured with security options

In AppWorks Platform, the Web gateway can be configured with or without enabling the security features. The most secure way of configuring the Gateway is authenticating users with digital certificates. In addition to this, you can also configure security features in AppWorks Platform through the Security Properties page of Management Console. For more information on configuring security options for Web gateway, see [Managing gateway with security options](#).

# Managing authentication

Identity or digital identity is a claim made by users to uniquely identify themselves. For example, name or employee ID. Identity is the foundation for security management in any organization.

Authentication is the process of confirming the claim made by the user. Even though authentication and identity are not the same, the goal of authentication is to verify that users are actually who they claim to be. Authentication is accomplished by presenting one's identity and credentials. Examples of credentials are passwords, one-time tokens, and digital certificates. The medium of verifying the user's identity is by using a login form.

## Single Sign On

Single Sign On (SSO) is a security model that authenticates users the first time, after which they can access the services of multiple software systems. By doing so, SSO removes the need for redundant authentication. For information on using SSO in AppWorks Platform, see [Single Sign On](#).

The AppWorks Platform SSO feature is able to exchange identity information with other software systems that support the Security Assertion Markup Language (SAML) standard or OpenText Directory Services (OTDS) protocol.

Security standards currently supported by AppWorks Platform are:

- SAML 2.0
- WS-Security: SOAP Message Security 1.0
- WS-Security Username Token Profile 1.0
- WS-Security SAML Token Profile 1.1

According to the guidelines of Basic Security Profile 1.0, AppWorks Platform conforms to:

- Core
- Transport Layer Mechanisms such as SSL
- Username Token
- SAML Token

## SAML

- SAML is an XML standard for exchanging authentication and authorization data between security domains, that is, between an identity provider, that is a producer of assertions,

and a service provider, that is consumer of assertions. SAML is a product of the OASIS Security Services Technical Committee.

- AppWorks Platform supports SAML 2.0 as the authentication mechanism.

### **WS-Security**

- Communications protocol providing a means for applying security to Web Services.
- Working in the application layer, incorporates security features in the header of a SOAP message, thereby ensuring end-to-end security.

### **OTDS**

- OTDS is a proprietary protocol for identity management across OpenText products and supports SSO. OTDS also supports adding an authentication token to the header of a SOAP message.
- AppWorks Platform supports OTDS 10.2 and above as the authentication mechanism.

### **Trust relationships between service groups**

The SSO component provides statements about identity. Administrators can use the security administration task to manage trust relationships between service groups.

### **Custom forms**

The client-side SSO library facilitates easy development of custom login forms.

## **Security Assertion Markup Language**

Security Assertion Markup Language (SAML) is a standard, XML-based framework for creating and exchanging security information. The information about the identity of a user can be exchanged between AppWorks Platform and other software systems that support the SAML standards. When using SAML, if a protected Web service operation is invoked, a client must provide valid assertions on the identity of the user before access is allowed. The client can get assertions by verifying credentials with a special Web service operation of Single Sign-on (SSO). The client must send these assertions within each SOAP request to gain access. The receiving service group verifies and validates the assertions. When the assertions are valid, and the user has proper authorization, access is allowed.

There are two uses cases for SAML:

- Web single sign-on
- Authentication on service level, also known as system-to-system authentication

Web SSO improves security and user experience when several components are integrated, and the user has to authenticate all of them. The use of an external Identity Provider (IDP) helps users to provide their user credentials only once for all the systems, instead of providing them for every system. When using SAML for Web SSO, all the communication is done through the browser as described in the [SAML 2.0 specification](#).

With system-to-system authentication, SAML assertions are passed as part of the SOAP header, as described in [WS-Security SAML Token Profile document of Oasis](#).

When an external IDP provides SAML assertion for a user, the SAML assertion is signed with the certificate of the external IDP. While sending a SOAP request to AppWorks Platform, the SAML assertion is placed in the SOAP header. Add these certificates to the trust store using the security administration task. For information on adding certificates, see [Managing SAML trust](#) and [Managing a certificate](#).

AppWorks Platform implements the following for SAML 2.0:

- [WS-Services SAML Token Profile](#) to send SAML assertions as part of the SOAP header. This is used in system-to-system integrations. See [Managing SAML trust](#) for configuring trust with a SAML 2.0 assertion provider.
- Clear separation of the IDP from the Service Provider (SP).
- SAML 2.0 configuration in the form of metadata, which can be used to exchange the configurations between IDPs and SP. Import the metadata during authenticator configuration on security administration. For details about SAML 2.0 compliant IDP authenticator, see [Configuring SAML 2.0 Authenticator](#) and [SAML 2.0 variables](#).
- Single Logout (SLO) to terminate the user session.
- SP to initiate a Web SSO check to confirm authentication.

**Note:** AppWorks Platform is SAML 2.0 compliant only as an SP and validates SAML 2.0 assertions by other IDPs. It does not generate SAML 2.0 assertions and therefore cannot function as an IDP. The SAML 2.0 assertion is not used internally in AppWorks Platform. It is used only to authenticate the user, after which AppWorks Platform generates an internal SAML authentication token, which is further used in the users session.

#### **Benefits of using SAML protocol for authentication:**

- **Single repository for user information:** Maintaining a single repository for user information avoids inconsistency of user information and reduces the risk of privacy threats.
- **Reduction in administrative overheads:** Administrators can configure all their related sites or applications to depend on a single Identity Provider (IDP), reducing the maintenance overhead.
- **Better experience for users:** Seamless integration of applications which depend on a single IDP, providing an SSO experience for users. In addition, identity federation, that is, linking of multiple identities with SAML improves the customized user experience at each service while promoting privacy.
- **Decoupling security from application development logic:** SAML abstracts the security framework from software development, ensuring that the security aspect can be customized or changed later, based on requirements. This decoupled approach also helps to reduce the maintenance costs.

#### **Advantages of AppWorks Platform with SSO based on SAML:**

- **Improved security:** AppWorks Platform offers a more secure framework for handling service requests.

- **Increased interoperability:** AppWorks Platform is compliant with the latest security standards such as SAML, WS-Security, and XML Encryption.
- **Anonymous login:** People and systems that use Web services in AppWorks Platform can securely access these services based on SAML credentials without repeated authentication. This makes AppWorks Platform more efficient for end users who use a website that implements AppWorks Platform services internally, because they do not have to be registered in LDAP.
- **Single instance authentication:** AppWorks Platform does not require sender identification for each message that passes the Web Gateway. Identification is handled once, after which access is controlled through security tokens.

**Note:** If your AppWorks Platform application works in a portal and authenticates users against an external IDP, the authentication token of this external IDP is used to validate the user in AppWorks Platform. This validation is done in an AppWorks Platform SSO service container plugin. The SSO service container validates the user and the authentication token against the external IDP. If they are valid, it returns a set of SAML Assertions. Therefore, the AppWorks Platform SSO service container generates SAML assertions.

If your application is not on a portal and you wish to authenticate against an external IDP, then you can create a custom authentication plugin. When the user provides credentials, it is sent to the AppWorks Platform SSO service container. Usually, the SSO service container validates the credentials against the AppWorks Platform repository (LDAP), but in this case a custom SSO plugin is used. This plugin authenticates against the external IDP and after authentication returns a set of SAML assertions. These assertions are used when invoking AppWorks Platform. For more information on creating authentication plugins, see [Authentication plugins](#).

## **Configuring SAML 2.0 Authenticator**

This topic describes the procedure to configure a SAML 2.0 Authenticator used for user interface-based authentication.

### **Before you begin:**

You must get the metadata from the external Identity Provider (IDP) and save it at a known location. You need it during the configuration process.

Occasionally, there can be a requirement to use an external SAML 2.0 IDP for authenticating users in AppWorks Platform. With the SAML 2.0 functionality, it becomes easy to configure the integration between the IDP and AppWorks Platform, which acts as a Service Provider (SP). The IDP can be configured according to the security needs of the application and AppWorks Platform can be configured to trust this SAML 2.0 compliant IDP. If there is no specific security requirement for user authentication, then the AppWorks Platform built-in authentication mechanism can be used.

### **SAML 2.0 Metadata**

The configuration of a trust relation between an IDP and AppWorks Platform as SP can be done as follows:

1. Configure or import the SAML 2.0 IDP metadata in AppWorks Platform as described below and import the SAML 2.0 SP metadata in the IDP. The SAML 2.0 IDP metadata is added to a SAML Authenticator configuration.
2. From the Security Administration task, select the authenticator for which you want to export the SP SAML 2.0 metadata for the IDP, click **Get Metadata**, and then add it to the external IDP. In some IDPs, the SAML 2.0 SP metadata has to be imported through a URL. This can be done by using the metadata URL, which exports the SAML 2.0 SP metadata through this URL.

To setup AppWorks Platform as SP and work together with an external IDP, the administrator has to create an Authenticator. An authenticator defines which IDP is used for authenticating the users.

1. Navigate to the **Welcome** page > **Security Administration**, select **Authenticators**, and do one of the following:
  - Click  on the **Shared Authenticators** grid to create an authenticator for all the organizations.
  - Click  on the **Organizational Authenticators** grid to create an authenticator for all the users in a specific organization.

The Authenticator Properties dialog box opens.

2. Type a unique identifier for the authenticator in the ID field.
3. Select **SAML2.0 Authenticator** from the Type dropdown list.
4. Select **Default** to mark this authenticator as the default authenticator.
5. Select **Test only** if you only want to test this authenticator before making it the default. The Test Url field displays the URL that is used to access the AppWorks Platform instance with this authenticator configuration active.
6. Type a description in the **Description** field.
7. In the FrameProperties section, do the following:

Field	Description
No Frame	Select this option if you do not want any frame around the Login form. <b>Note:</b> This option does a complete page reload resulting in losing the browser context. With this option, users can view the complete URL of the external IDP, which gives more trust to the users as they can validate going to a trusted site before providing the username and password. This is the default and most secure setting.
Maximize	Select this option if you want the Login form to be displayed in a maximized frame.

Field	Description
Width, Height	Optional. Modify the width and height of the frame that displays the Login form.
Target Frame	Specify a value, for example, <code>CordysRoot</code> , to prevent the display of the complete form if AppWorks Platform forms are displayed in a portal. Usually, after the users are authenticated, the complete form is reloaded. If you specify <code>CordysRoot</code> as Target Frame, the complete form is not displayed. Only the part of the portal where the AppWorks Platform form is displayed is reloaded.

- Paste the metadata retrieved from the external IDP, in the **Entity Descriptor** field of SAML2.0 Properties tab.

**Note:** The SAML 2.0 IDP metadata is very important to establish trust between the IDP and the AppWorks Platform instance. The metadata contains information such as Login URL, Logout URL, and certificates used for signing the SAML2 assertion.

- Click **Save**.

Depending upon your choice, the SAML 2.0 Authenticator is configured and added to either the Shared Authenticators list or the Organizational Authenticators list.

### Using the CordysBuiltIn Authenticator

If accessing the default SAML Authenticator results in an unrecoverable error, the administrator still can login and correct the wrong configuration using the built-in authenticator of AppWorks Platform.

To access the AppWorks Platform built-in authenticator, you must use the `authID` URL parameter with the value `CordysBuiltIn`. This bypasses any default configured authenticator and displays the built-in AppWorks Platform Login page.

Example:

`https://www.acme.com/home/myorg/?authID=CordysBuiltIn`

**Note:** In case, for security reasons, the administrators do not want to allow access to the AppWorks Platform built-in authenticator, they can block access to the SAML Authentication Request Web service through ACL. See [Configuring ACL for Web service interfaces and operations](#).

### After you complete:

The signing certificate of the external IDP has a certificate chain. You must add the certificates in that certificate chain to the Trust Store. See [Adding a new certificate](#). Before you add the certificates, select **SAML 2.0 IDP Issuer Certificate** from the **Trust Context** list.

## SAML 2.0 variables

### Resolving Variables

When configuring an authenticator, you can use variables that can be added at any of the following levels:

- Authenticator
- Organization
- Shared

When resolving the variables, the resolve algorithm for a variable searches for the authenticator-specific variables first. If the variable is not found, it searches the organization-level variables followed by the shared-level variables.

### SAML Variables

You can specify the following variables:

Variable	Description
BASE_URL	Legacy variable. Define the public cluster URL instead. See <a href="#">Configuring access URLs</a> .
ENTITY_ID	When registering a Service Provider (SP) at an Identity Provider (IDP), the ENTITY_ID from the SAML 2 SP metadata is used as an identifier. If ENTITY_ID is already used, you can use the ENTITY_ID variable to provide the authenticator with a new ENTITY_ID that can be used in the SAML 2 SP metadata.
IDP_RETURN_URL	When the variable is set, the AppWorks Platform Assertion Consumer Service (ACS) redirects to this URL after validating the POSTed SAML assertion from the IDP. That is, after the user signs into the IDP, the browser is redirected to the URL as specified in this variable.  IDP_RETURN_URL is used when the No Frame option is set in FrameOptions. The IDP integration in AppWorks Platform is SP and UserAgent-based. All protocol redirects are done through the UserAgent. When the No Frame option is used, the entire browser is redirected to the IDP. The current browser context is completely lost in the browser. After the user has authenticated at the IDP, the ID does a SAMLResponse POST to the AppWorks Platform ACS. After validation, the ACS redirects the browser back to the URL configured in IDP_RETURN_URL. The IDP_RETURN_URL can also be used if you do not want to redirect to the AppWorks Platform Welcome page, but to another page, like an application XForm.  The value specified must be the path after the domain, for example /home/myorg or /home/myorg/com/acme/app/myform.caf
ACS_CLASS_	With this variable, the ACS class that is used by default can be

Variable	Description
HTTPPOST	<p>changed. If it is set, then the fully qualified class name is used as the ACS in the <code>SAMLRequest</code> of the IDP referring to where the IDP must POST the SAML2 assertion.</p> <p>To use this variable, create a new class that extends the <code>com.eibus.sso.web.saml2.acs.service.HttpPost</code> class. The custom ACS class can, for example, invoke a BPM which checks if the user already exists and creates one if needed.</p>

## SAML over HTTP Post

The SAML standard describes various ways of supplying credentials. AppWorks Platform supports SAML over HTTP post binding for supplying credentials.

### How it works

The SAML over HTTP post binding provides a means to integrate with Web Single Sign On (SSO) solutions. Before a Web service can be used, a client validates credentials with a SSO service. After verifying the credentials, the SSO component returns assertions.

The assertions are in HTML form. This HTML form is then posted to the AppWorks Platform Web Gateway. The service container reads the assertions and checks if the user is authorized to complete the request. After correct authorization, the request is executed.

The SAML assertions can be provided by AppWorks Platform SSO, or by any other component that correctly supports the SAML standard. The credentials supplied to SSO can consist of all the identity types that AppWorks Platform supports.

The SOAP request will be executed only if:

- The assertions are valid, well-formed, and not expired.
- The service container that received the request trusts the SSO entity which provided the SAML assertions (if signing is mandatory for the service group).
- The SAML assertions can be mapped to a valid OpenText AppWorks Platform User.
- The user has authorization to execute the SOAP request.

### Usage

The SAML token profile can be used when interoperability with other authentication or authorization systems is important. As message integrity is very important, it is advisable to use message-level or transport-level security. SAML and WS-Security are emerging standards and are used in many large enterprise systems.

## Types of identity

Digital identity or identity is a claim made by users to uniquely identify themselves, for example, name, and employee ID. This digital identity is a combination of identification and credentials. For example, user name and password. Authentication is done to confirm

the identity of the user. During the process of authentication, users must provide the correct credentials to identify themselves. This guarantees that users are who they claim they to be. From that moment, a computer system can recognize the identity of the user.

The applications, such as service containers or other software on AppWorks Platform must use one of the following identity types:

<b>Identity type</b>	<b>Implementation</b>
Anonymous	Client sends a SOAP message without any statement about identity. These SOAP messages are mapped to an anonymous user. Within AppWorks Platform, the anonymous user is represented by the authenticated user named <b>anonymous</b> , so that ACLs can be applied to this user. For more information, see <a href="#">Anonymous user</a> .
AppWorks Platform	Client captures the user credentials directly in the SOAP request. For more information, see <a href="#">AppWorks Platform Identity</a> .
WS-Security SAML token	Client provides a trusted statement on the identity of the user in the SOAP request. This statement can be acquired by authenticating with the Single Sign On (SSO) service. The reply from the SSO service contains a statement that will be trusted by the services that receive a message with that statements. For more information, see <a href="#">WS-Security SAML token</a> and <a href="#">Managing SAML trust</a> .
WS-Security User name token	Client provides a user name token stating the identity information of the user in the SOAP header. For more information, see <a href="#">WS-Security user name token</a> .

When there are no identity types present in the SOAP request, the SOAP message is considered as anonymous.

**Note:** AppWorks Platform does not support the use of multiple identities in the SOAP header.

### ***Anonymous user***

Anonymous user can be used in situations where identification is not necessary. This can be useful in a portal to provide public information without signing in. The information presented to such users is not confidential or specific to a particular user.

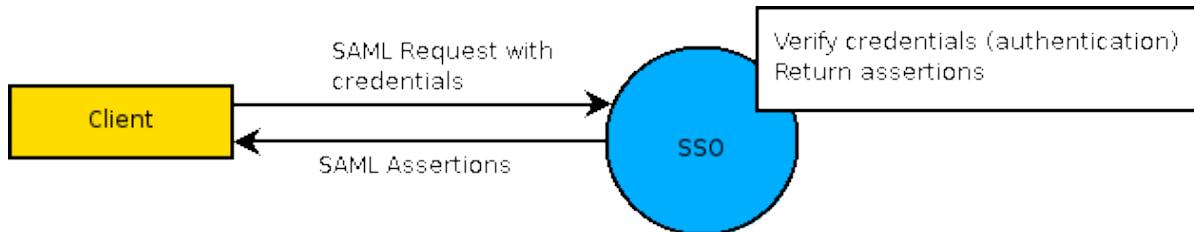
Usually, identity details are specified in a SOAP request. However, in the case of anonymous users, the identity details are not specified in the SOAP header. The Web server is configured to allow anonymous access. Therefore, when AppWorks Platform is accessed anonymously, the built-in authenticated user 'anonymous' is used. The ACL of that authenticated user is applied to every SOAP request. The SOAP request executes only if the user 'anonymous' is authorized to execute the SOAP request.

All the Web service operations, from an application point of view, must be accessible anonymously and must have the ACL configured to allow access to the authenticated user 'anonymous'.

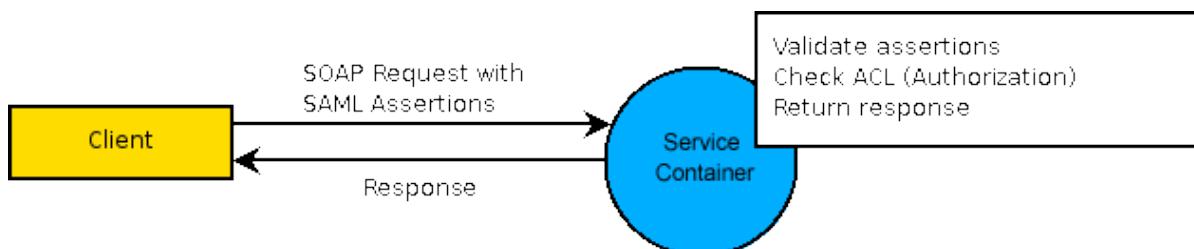
### **WS-Security SAML token**

WS-Security is a standard for securing Web services. It provides authentication and credential formats. The Security Assertion Markup Language (SAML) token profile can be used when there is a need to work with other authentication or authorization systems. As integrity of a message is important, you must use message-level or transport-level security, for example Secure Sockets Layer (SSL). SAML and WS-Security are emerging standards and are used in many large enterprise systems.

The WS-Security SAML token contains credentials in the form of assertions. These SAML assertions are proof that a user correctly authenticated with Single Sign On (SSO). The client verifies the user credentials with SSO before a Web service is used. SSO returns the assertions after verifying the credentials. The following diagram describes this process.



The client then includes these assertions in the request header of the service container. The service container reads the assertions and checks if the user is authorized to complete the request. The request is run only after ensuring that the user is authorized to work with the service container. The following diagram describes the request sent to the service container.



The SAML assertions can be provided by SSO or by any other identity provider that supports the SAML standard. The credentials supplied to SSO can consist of all the identity types that AppWorks Platform supports.

If the SAML assertions are signed, the service container must trust the signing certificate and the entire chain of certificates of SSO that provided the assertions. Trust relations can be set through the security administration task.

The SOAP request runs only if the following conditions are met:

- The assertions are valid, well-formed, and unexpired.
- The service container that received the request trusts the SSO that provided the SAML assertions. This is required if signing is mandatory for the service container.
- The SAML assertions can be mapped to a valid AppWorks Platform user.
- The user has authorization to run the SOAP request.

## Configuration

Configure the Web server explicitly to enable anonymous access. If the client provides valid credentials in the SOAP request, the Web server enables the assertion to pass through without authentication. You must configure an authenticator in the service group configuration in LDAP. Use this authenticator to verify the credentials.

## Example

The following is an example of a SAML 2 assertion in the SOAP header:

```
<SOAP:Envelope xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP:Header>
    <wsse:Security xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd">
      <saml2:Assertion xmlns:saml2="urn:oasis:names:tc:SAML:2.0:assertion"
        xmlns:xs="http://www.w3.org/2001/XMLSchema" ID="8101AB8C-A44A-4324-403F-A23C4FA596E0"
        IssueInstant="2014-01-28T08:55:55.489Z" Version="2.0">
        <saml2:Issuer>myIDP</saml2:Issuer>
        <ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
          <ds:SignedInfo>
            <ds:CanonicalizationMethod
              Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
            <ds:SignatureMethod
              Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
            <ds:Reference URI="#8101AB8C-A44A-4324-403F-A23C4FA596E0">
              <ds:Transforms>
                <ds:Transform
                  Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>
                <ds:Transform
                  Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
                <ec:InclusiveNamespaces
                  xmlns:ec="http://www.w3.org/2001/10/xml-exc-c14n#"
                  PrefixList="xs" />
```

```

        </ds:Transform>
    </ds:Transforms>
    <ds:DigestMethod
Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>

<ds:DigestValue>6qHuYkVqZ/7SNaSyNh8Hw9kVyoI=</ds:DigestValue>
    </ds:Reference>
    </ds:SignedInfo>
    <ds:SignatureValue>

O3Q1R+zmB0IfT8KmjRgAu4mSMuM2gssIo3H53V55FgsH4i9rlRQ1xb9LD1ncwuIv88Xds21Qw4g+KqglIvFh
sHFhOwgqcAECSVY4BxzXNEjkDONUGV1k8M22fmPAYnsy+HQj6TTvasO8fF4L5pR+Ya7b47rTofwED4lIIF1G
Qec=</ds:SignatureValue>
    <ds:KeyInfo>
        <ds:X509Data>
            <ds:X509Certificate>
MIIB0zCCATygAwIBAgIEUudzojANBgkqhkiG9w0BAQUFADAnMSUwIwYDVQQDExxTZWxmIFNpZ251
ZCBUXZN0IEN1cnRpZmljYXR1MB4XDTE0MDEyODA4NTU1M1oXDTE0MDEyOTA4NTU1M1owITEfMB0G
A1UEAxMWVmFsaWQgVGVzdCBDZXJ0aWZpy2F0ZTCBnzANBgkqhkiG9w0BAQEFAAOBJQAwgYkCgYE
11apB+4Ts9mmQL+1mpYQltRDrYmZJMxOkcmrOapOekhSHwQgi+MgU3kNEF5cz5f3mioO+U6sXp+Q
di0k1ncim+nAPpbN5jzw1bE5UcKoLQyFljZFQnwRAcc4pIIIt4TApxjoH+iWs0lpbxDZNyLxdr9LR
fdaBqVuAzryrfzXBdHdkCAwEAAaMSMBAwDgYDVR0PAQH/BAQDAgWgMA0GCSqGSIb3DQEBBQUAA4GB
ADNZPub1EH4aeZYIt/eBYmyB75z0y9g7sGFph0uISYoyTV340N14f0oG8ZDU8hMwImD5AkENxadR
NNLcQ+CqsdYnXylmvRI8kOY1rUwLm7H53b9rfxFZT3ofQ/AtlPvowGlNj4N7oEx5TIyyUA/V1M
PEx/YnJy8AA/FgdQIlbh</ds:X509Certificate>
        </ds:X509Data>
    </ds:KeyInfo>
    </ds:Signature>
    <saml2:Subject>
        <saml2:NameID Format="urn:oasis:names:tc:SAML:1.1:nameid-
format:emailAddress">
NameQualifier="myNameIDQualifier">someuser@example.com</saml2:NameID>
            <saml2:SubjectConfirmation
Method="urn:oasis:names:tc:SAML:2.0:cm:bearer">
                <saml2:SubjectConfirmationData NotOnOrAfter="2014-02-
04T14:46:30Z"/>
            </saml2:SubjectConfirmation>
        </saml2:Subject>
        <saml2:Conditions NotBefore="2014-02-04T14:26:30Z"
NotOnOrAfter="2014-02-04T14:46:30Z">
            <saml2:AudienceRestriction>
                <saml2:Audience>mySP</saml2:Audience>
            </saml2:AudienceRestriction>
        </saml2:Conditions>
        <saml2:AuthnStatement>
            <saml2:AuthnContext>
                <saml2:AuthnContextClassRef/>
            </saml2:AuthnContext>
        </saml2:AuthnStatement>
        </saml2:Assertion>
    </wsse:Security>
</SOAP:Header>
```

## User identity of SAML assertions

The user identity retrieved from a SAML assertion is based on the NameID tag.

```
<saml2:NameID Format="urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress"
NameQualifier="myNameIDQualifier">someuser@example.com</saml2:NameID>
```

The identity `someuser@example.com` is used as `osidentity`. The authentication framework uses the `osidentity` to find the authenticated user, and resolves the organization user in combination with the organization information.

## Sample SOAP messages for SAML authentication

Sample SOAP request to get SAML assertions from single sign on:

```
<SOAP:Envelope xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP:Header>
    <wsse:Security xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-
200401-wss-wssecurity-secext-1.0.xsd">
      <wsse:UsernameToken xmlns:wsse="http://docs.oasis-
open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd">
        <wsse:Username>jopl</wsse:Username>
        <wsse:Password>whateverthepasswordwillbeputtherel</wsse:Password>
      </wsse:UsernameToken>
    </wsse:Security>
  </SOAP:Header>
  <SOAP:Body>
    <samlp:Request IssueInstant="2009-04-01T10:23:11Z"
      MajorVersion="1" MinorVersion="1"
      RequestID="a997c83a8d-b5d7-b930-edba-02e37ab1765"
      xmlns:samlp="urn:oasis:names:tc:SAML:1.0:protocol">
      <samlp:AuthenticationQuery>
        <saml:Subject xmlns:saml="urn:oasis:names:tc:SAML:1.0:assertion">
          <saml>NameIdentifier Format="urn:oasis:names:tc:SAML:1.1:nameid-
format:unspecified">jopl</saml>NameIdentifier>
        </saml:Subject>
      </samlp:AuthenticationQuery>
    </samlp:Request>
  </SOAP:Body>
</SOAP:Envelope>
```

The following is the SOAP response with SAML assertions from SSO:

```

<SOAP:Envelope xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP:Body>
    <samlp:Response
      InResponseTo="a997c83a8d-b5d7-b930-edba-02e37ab1765"
      IssueInstant="2009-04-01T11:23:11.679Z" MajorVersion="1"
      MinorVersion="1"
      ResponseID="A9D550166-1DC3-4AF4-9C56-271E028C2DFE"
      xmlns:samlp="urn:oasis:names:tc:SAML:1.0:protocol">
      <samlp:Status xmlns:samlp="urn:oasis:names:tc:SAML:1.0:protocol">
        <samlp:StatusCode Value="samlp:Success"/>
      </samlp:Status>
      <saml:Assertion
        AssertionID="A4B413744-287C-4A8F-8D0D-C9283F19A339"
        IssueInstant="2009-04-01T11:23:11.679Z"
        Issuer="https://www.cordys.com/SSO" MajorVersion="1"
        MinorVersion="1" xmlns:saml="urn:oasis:names:tc:SAML:1.0:assertion">
        <saml:Conditions NotBefore="2009-04-01T11:18:11.679Z"
          NotOnOrAfter="2009-04-01T19:23:11.679Z"
        <ns1:saml="urn:oasis:names:tc:SAML:1.0:assertion"/>
        <saml:AuthenticationStatement
          AuthenticationInstant="2009-04-01T11:23:11.679Z"
          AuthenticationMethod="urn:oasis:names:tc:SAML:1.0:am:password"
        <ns1:saml="urn:oasis:names:tc:SAML:1.0:assertion">
          <saml:Subject
        <ns1:saml="urn:oasis:names:tc:SAML:1.0:assertion">
          <saml:NameIdentifier
            Format="urn:oasis:names:tc:SAML:1.1:nameid-
            format:unspecified">jopl</saml:NameIdentifier>
          </saml:Subject>
        </saml:AuthenticationStatement>
      </saml:Assertion>
      <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
        <SignedInfo>
          <CanonicalizationMethod
            Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
          <SignatureMethod
            Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
          <Reference URI="#A4B413744-287C-4A8F-8D0D-C9283F19A339">
            <Transforms>
              <Transform Algorithm="http://www.w3.org/2001/10/xml-exc-
              c14n#"/>
            </Transforms>
            <DigestMethod
              Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
              <DigestValue>WbVQ557X2lU2TcrmQb1hn4yOPkI=</DigestValue>
            </Reference>
          </SignedInfo>
        <SignatureValue>R4fQ+wNie9Rey1+hAcDY3JVz3Dr7rUPOrd4JZuW7vhbTSJFSkaUw+1PYB/18dEJIMTm6
        99GwAq+mJ3jIV+ybea7eQ9XQTAfhZppAQnr+6k8kdkQnILLiYJLk0WIzOII119OI/vi+AL8PzolYowSQhvru
        sK13izKcAk9d+vL+6QY=</SignatureValue>
        <KeyInfo>

```

```

<X509Data>
<X509Certificate>MIIB4DCCAUmgAwIBAgIQaJsGtYWAXgC78iR/9KXGDDKNBggkqhliG9w0BAQUFADArMQ8
wDQYDVQQKEwZzeXN0ZW0xGDAWBgNVBAbUDk8vbm10b3JAQ05EMTMwMDAeFw0wOTAzMDkxMzEyNTBaFw0xOTA
zMDcxMzEyNTBaMDIxHzAdBgNVBAMTFnNpbmdsZSBzaWduLW9uIHNlcnZpY2UxDzANBgNVBAoTBnN5c3R1bTC
BnzANBgkqhkiG9w0BAQEFAAOBJQAwgYkCgYEaur1RNnLrS9RepnKA1ZMyfzcfv4B5b2NxWDbTxbpWVbJq/p+
Tp9r+akWWiRc20cZQH9esSJ9n3K8KLge/VLUPYSWKJt7P+gqlRaze4a/W51cYAC5QP+U/KP/UJ2csDw11MI0
magDVQZ1fZTj02s+j9LCcpq00LBzXQTEhmrnZbxECawEAATANBgkqhkiG9w0BAQUFAOBgQCm3a4Eg94g9xq
IswNhHu6b+yJpmHO8WDgHdGyZ2kQ0VezBa0ECit57aZszco7qG2ZIwnV5WxBYFD+PmcqjcXFhzkviHileZoT
pWFcfEpySvokwlrGz9BDyrn6FyGC3YfZ8N0eeXyJW5AkNAD59CSnEblilPNON2TQpeOaeL0roUg==
</X509Certificate>
</X509Data>
</KeyInfo>
</Signature>
<samlp:AssertionArtifact
xmlns:samlp="urn:oasis:names:tc:SAML:1.0:protocol">MDF3eiA3HTMvRoDydILiGLihWu7akA65U
vZOe0p5hka4siLYQInR/N1C</samlp:AssertionArtifact>
</samlp:Response>
</SOAP:Body>
</SOAP:Envelope>

```

### Example of a SAML assertion

The following is an example of a SAML 2 assertion in the SOAP header:

```

<SOAP:Envelope xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/">
<SOAP:Header>
<wsse:Security xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-
200401-wss-wssecurity-secext-1.0.xsd">
<saml2:Assertion xmlns:saml2="urn:oasis:names:tc:SAML:2.0:assertion"
      xmlns:xs="http://www.w3.org/2001/XMLSchema" ID="8101AB8C-A44A-4324-
403F-A23C4FA596E0"
      IssueInstant="2014-01-28T08:55:55.489Z" Version="2.0">
<saml2:Issuer>myIDP</saml2:Issuer>
<ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
<ds:SignedInfo>
<ds:CanonicalizationMethod
Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
<ds:SignatureMethod
Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
<ds:Reference URI="#8101AB8C-A44A-4324-403F-A23C4FA596E0">
<ds:Transforms>
<ds:Transform
Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>
<ds:Transform
Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
<ec:InclusiveNamespaces

```

```

xmlns:ec="http://www.w3.org/2001/10/xml-exc-c14n#"
      PrefixList="xs"/>
    </ds:Transform>
  </ds:Transforms>
  <ds:DigestMethod
Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>

<ds:DigestValue>6qHuykVqZ/7SNaSyNh8Hw9kVyoI=</ds:DigestValue>
</ds:Reference>
</ds:SignedInfo>
<ds:SignatureValue>

O3Q1R+zmB0IfT8KmjRgAu4mSMuM2gssIo3H53V55FgsH4i9rlRQ1xb9LD1ncwuIv88Xds21Qw4g+KqglIvFh
sHFhOwgqcAECSVY4BxzXNEjkDONUGV1k8M22fmPAYnsy+HQj6TTvasO8fF4L5pR+Ya7b47rTofwED41IIF1G
Qec=</ds:SignatureValue>
<ds:KeyInfo>
<ds:X509Data>
<ds:X509Certificate>
MIIB0zCCATygAwIBAgIEUudzojANBgkqhkiG9w0BAQUFADAnMSUwIwYDVQQDExxTZWxmIFNpZ251
ZCBUXN0IEN1cnRpZmljYXR1MB4XDTE0MDEyODA4NTU1M1oXDTE0MDEyOTA4NTU1M1owITEfMB0G
A1UEAxMWVmFsaWQgVGVzdCBDZXJ0aWZpy2F0ZTCBnzANBgkqhkiG9w0BAQEFAAOBJQAwgYkCgYE
11apB+4Ts9mmQL+1mpYQLtRDrYmZJMxOkcmrOapOekhSHwQgi+MgU3kNEf5cz5f3mioO+U6sXp+Q
di0k1ncim+nAPpbN5jzw1bE5UcKoLQyFljZFQNwRAcc4pIIt4TAploquent+iWs0lpbxDZNyLxdr9LR
fdabqVuAzryrfzXBdHdkCAwEAAsMSMBAwDgYDVR0PAQH/BAQDAgWgMA0GCSqGSIb3DQEBBQUAA4GB
ADNZPub1EH4aeZYi/eBYmyB75z0y9g7sGFph0uISyoyTV340N14f0oG8ZDU8hMwImD5AkENxadR
NNLcQ+CqsdYnXylmvRI8kOY1rUwLm7H53b9rfxFZT3ofQ/AtlpvowG1Nj4N7oEx5TIyyUA/V1M
PEx/YnJy8AA/FgdQIlbh</ds:X509Certificate>
</ds:X509Data>
</ds:KeyInfo>
</ds:Signature>
<saml2:Subject>
<saml2:NameID Format="urn:oasis:names:tc:SAML:1.1:nameid-
format:emailAddress">
NameQualifier="myNameIDQualifier">someuser@example.com</saml2:NameID>
<saml2:SubjectConfirmation
Method="urn:oasis:names:tc:SAML:2.0:cm:bearer">
<saml2:SubjectConfirmationData NotOnOrAfter="2014-02-
04T14:46:30Z"/>
</saml2:SubjectConfirmation>
</saml2:Subject>
<saml2:Conditions NotBefore="2014-02-04T14:26:30Z"
NotOnOrAfter="2014-02-04T14:46:30Z">
<saml2:AudienceRestriction>
<saml2:Audience>mySP</saml2:Audience>
</saml2:AudienceRestriction>
</saml2:Conditions>
<saml2:AuthnStatement>
<saml2:AuthnContext>
<saml2:AuthnContextClassRef/>
</saml2:AuthnContext>
</saml2:AuthnStatement>
</saml2:Assertion>

```

## User identity of SAML assertions

The user identity retrieved from a SAML assertion is based on the NameID tag.

```
<saml2:NameID Format="urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress"  
NameQualifier="myNameIDQualifier">someuser@example.com</saml2:NameID>
```

The identity `someuser@example.com` is used as `osidentity`. The authentication framework uses the `osidentity` to find the authenticated user and then in combination with the organization, the organization user is resolved.

## WS-Security user name token

WS-Security is a standard for securing Web services. This standard provides various formats for authentication and credentials. AppWorks Platform supports the WS-Security user name token to be able to provide credentials.

The user name token profile can be used when there is a need for a standard way of authentication and authorization. WS-Security is an emerging standard, used in many large enterprise systems. As message integrity is important, you must use message-level or transport-level security.

The WS-Security user name token contains credentials including a user name and password. The service container verifies the given credentials before the SOAP request is run. The request is then run on behalf of the user and the organization. A password in a WS-Security User name token is of type `PasswordText`, which is a plain text password and must be protected against security attacks. Protection against attacks must be provided using SSL for the connection from the IE browser to the Web server. For more information on how this authentication is part of a SOAP request, see [AppWorks Platform Identity](#).

**Note:** The WS-Security user name token is the identity type used to retrieve SAML assertions from AppWorks Platform Single Sign On.

## Configuration

The Web server must be configured explicitly to allow anonymous access. When this is configured, the Web server does not try to authenticate; however, the client must provide valid credentials in the SOAP message.

The service group must have an authenticator configured in their service group configuration in LDAP. This authenticator is used to verify the credentials.

## Example

The following is an example of WS-Security Username Token using PlainText password:

```
<wsse:Security xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
```

```

wssecurity-secext-1.0.xsd">
<wsse:UsernameToken xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-
200401-wss-wssecurity-secext-1.0.xsd">
    <wsse:Username>jopl</wsse:Username>
    <wsse:Password>jopl</wsse:Password>
</wsse:UsernameToken>
</wsse:Security>

```

**Note:** The layout of this example conforms to the Web Services Security UsernameToken Profile 1.0.

## AppWorks Platform Identity

AppWorks Platform identity is the most simple form of authentication in AppWorks Platform. The AppWorks Platform identity can be used if the ability to work with other authentication or authorization systems is not required. As message integrity is important, it is recommended to use message-level or transport-level security. AppWorks Platform identity must preferably be used only on protected networks behind the AppWorks Platform Web gateway, because SOAP headers can easily be forged. The AppWorks Platform Web gateway verifies credentials before setting the AppWorks Platform identity SOAP header and for an external client it is not possible to manually set the AppWorks Platform Identity header.

The client encapsulates the user credentials in AppWorks Platform Identity format in the SOAP request header. The service container verifies the given credentials before the SOAP request is run. The request is then run on behalf of the user and organization.

The SOAP request is run only if:

- The user name and password are valid.
- The user name and password can be mapped to a valid AppWorks Platform user.
- The user has authorization to run the SOAP request.

### Example of an AppWorks Platform Identity SOAP Header

The following is a sample SOAP request header to identify a user on AppWorks Platform. Replace USER and ORGANIZATION with the actual values of the user and organization on behalf of which this SOAP request is to be run.

```

<SOAP:Header>
    <header>
        <sender>
            <user>USER@ORGANIZATION</user>
        </sender>
    </header>
</SOAP:Header>

```

## Authentication mechanisms

Authentication is the process of ensuring that a user is who or what they claim to be. User authentication can be done at various locations within the AppWorks Platform framework. Authentication mechanism refers to the part of the framework that handles authentication.

Authentication mechanism also includes authorization of a user. Authorization is the process of ensuring that a particular user has the required permissions to work on a particular resource. A resource can be an application, a service group, or a service container.

Authentication mechanisms supported in AppWorks Platform are described in the table below:

Name	Authentication or authorization level	Description	Identity types
Web Server	Authentication occurs in the Web server. Authorization occurs in the service group.	The Web server (TomEE) is configured to handle authentication. The Web server can be configured to authenticate against various repositories, such as NTLM, Active Directory, or LDAP. The Web gateway receives the user name of the authenticated user through the AUTH_USER variable that is set by the Web server.	<a href="#">AppWorks Platform Identity</a> or <a href="#">WS-Security SAML token</a>
WS-Security	User name token: Authentication and authorization occur in the service group.  SAML token: Authentication occurs at the identity provider. Authorization occurs in the service group.	When using the WS-Security authentication mechanism, the client inserts a WS-Security identity directly in the SOAP request header. The WS-Security identity can consist of various token profiles.	<a href="#">WS-Security user name token</a> or <a href="#">WS-Security SAML token</a>
SAML	Authentication occurs in an external SAML2	User are authenticated by a SAML2 compliant	<a href="#">AppWorks Platform Identity</a> or <a href="#">WS-Security SAML token</a>

Name	Authentication or authorization level	Description	Identity types
	compliant Identity Provider. Authorization occurs in the service group.	Identity Provider (IDP). AppWorks Platform redirects the browser to the IDP for user authentication. This redirect flow is based on the SAML2 protocol. After the authentication, an AppWorks Platform SAML token is generated.	<a href="#">Security SAML token</a>
OTDS	Authentication occurs in a OTDS server. Authorization occurs in the service group.	Users are authenticated in OTDS. AppWorks Platform redirects the browser to OTDS for user authentication. After the authentication, an AppWorks Platform SAML token is generated.	<a href="#">AppWorks Platform Identity or WS-Security SAML token</a>
Anonymous	Authorization takes place in the service group. No authentication is done.	When there is no identity provided using the other authentication mechanisms, the SOAP request is sent anonymously. It depends on the authorization of the anonymous user if access is allowed.	<a href="#">Anonymous user</a>

### One Web gateway, multiple authentication mechanisms

It is possible to use one Web gateway for multiple authentication mechanisms, as long as the individual SOAP requests only use one mechanism. Some mechanisms cannot be used simultaneously due to technical limitations. Web server authentication rules out all other options because the Web server authenticates all requests. It is advisable to create multiple Web gateways if you want to use different authentication mechanisms, to avoid confusion.

### One SOAP request, one authentication mechanism

Per SOAP request, only one authentication mechanism can be used. It is not possible to use multiple authentication mechanisms to authenticate a SOAP request simultaneously as the identity to be used on the bus is not clear.

**Note:** SOAP requests which use multiple authentication mechanisms simultaneously (for example SAML over HTTP GET and Web server authentication) are rejected by the Web gateway.

### Web server authentication

AppWorks Platform uses a web server for serving the user interface to the browser and for receiving web services requests. These requests must be run in a specific user context. To determine the user context, a user must be authenticated.

The Web server can be configured to authenticate users. TomEE support various authentication mechanisms such as basic authentication, NT LAN Manager (NTLM), and database authentication. TomEE configuration, which uses Tomcat internally, is the responsibility of the administrator. See the [generic documentation of Tomcat](#) for more information on this topic. The [AppWorks Developer community](#) provides additional resources that you may find helpful.

To use the default Platform authentication, the web server has to skip authentication and leave that to AppWorks Platform. If the web server does not perform user authentication, the authentication is triggered by the single sign on component in the Platform, if the user context is needed.

## Authentication through Single Sign On

AppWorks Platform provides various mechanisms for authentication and authorization including integration with the existing authentication mechanisms using the WS-Security and SAML standards. Besides support for these standards, AppWorks Platform can also integrate with [OpenText Directory Services](#) based on the proprietary OTDS protocol.

The following are the key authentication and authorization characteristics of the Single Sign On (SSO) feature of AppWorks Platform:

- SSO functionality using the WS-Security, SAML standards, and the OpenText-specific OTDS protocol.
- Support for various third party SSO solutions.
- Support for [Form-based authentication](#).
- Authentication using various WS-Security profiles.
- Configurable anonymous access to parts of AppWorks Platform.
- Support for storing and retrieving authentication information in different repositories using plugins.
- Support for participation in Identity and Access Management (IAM) systems.

- Support for federated identity. Federated identity management is a network of authentication systems that trusts each other. The user must provide credentials to one system and the user can continue working with other partner systems. For example, a sales representative can access the supplier database after authentication is done at the vendor's location.
- Backward compatibility with the standard AppWorks Platform authentication.

### ***Form-based authentication***

Form-Based Authentication (FBA) is the authentication mechanism that uses a login form in a Web page. With FBA, users can log in by entering their credentials directly on the form. After submitting, the credentials are verified and access is granted to the applicable resources.

Sample login form:

Login Name:	<input type="text"/>
Password:	<input type="password"/>

### ***Single Sign On***

AppWorks Platform uses SOAP-wrapped XML messages for all the communication in the platform. For example, to retrieve a list of customers from an SAP system, you require a number of SOAP messages between the required systems and components. In the traditional security mechanism, the AppWorks Platform Web gateway authenticates the sender of a SOAP message, after which the SOAP message is routed to a service container. The service container then checks the Access Control List (ACL) to decide if the request has to be run or denied service access. Although unlikely, this approach theoretically poses a security vulnerability if a user is able to evade the identity verification by the Web gateway and manages the ACL.

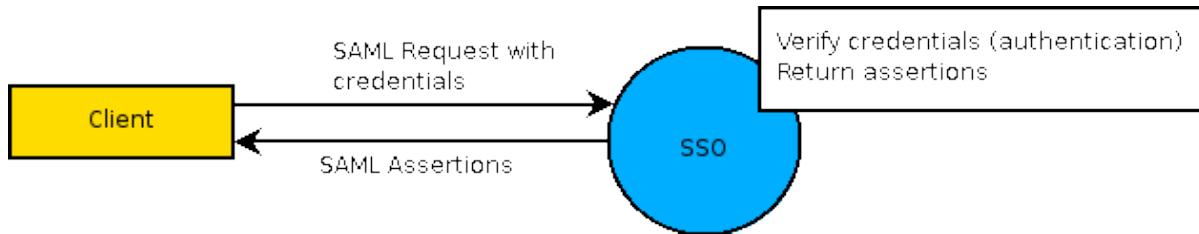
Therefore, to minimize the security risks, the new Single Sign On (SSO) implementation based on the Security Assertion Markup Language (SAML) standard works differently. The SOAP message senders are no longer required to validate their identity through the Web gateway. Instead, senders have to request SAML credentials from the new AppWorks Platform SSO component in the form of tokens. These tokens are attached to every SOAP message that is sent and the receiving service container uses the SSO component to validate the identity based on these tokens. The service container then uses the ACL mechanism to determine if a user is allowed to access the requested service. The SAML tokens are reusable and can be used for other service requests as well, potentially directed at a different service container.

AppWorks Platform provides the SSO feature to handle the process of authentication and authorization of users in a reliable and secured manner. SSO is based on the SAML and the Web Service Security (WS-Security) standards.

SSO provides a trusted identity to users of AppWorks Platform. See [Identity](#). This trusted identity is given in the form of signed SAML assertions. A client can get these SAML

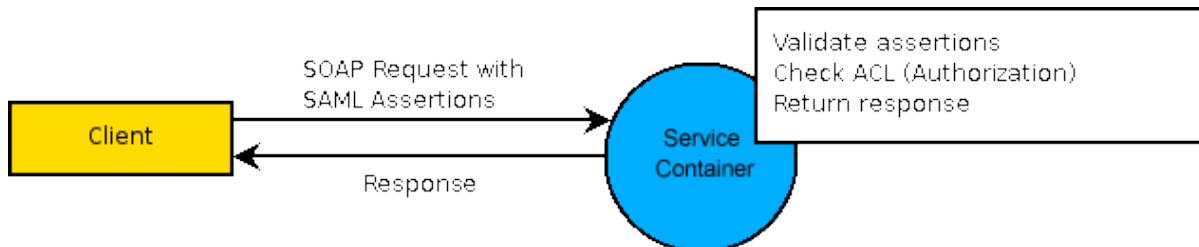
assertions by correctly authenticating with SSO. SAML assertions prove that a user is correctly authenticated with SSO.

The following image describes this process:



The client then includes these assertions in the request header of a SOAP request to the service container. The service container reads the assertions and checks if the user is authorized to complete the request. After the authorization is verified to be correct, the request is completed.

The following image describes the request to the service container:



The SAML assertions can be provided by the AppWorks Platform SSO or by any other source that supports the SAML standard. The credentials supplied to SSO can consist of all the identity types that AppWorks Platform supports.

If the SAML assertions are signed, the service container must trust the SSO that provided the assertions. You can set the trust relations in the Trust Stores tab of the Security Administration task.

### ***Using SAML artifacts***

When users sign into the client application, they receive a security token from the back-end server. This security token is called the SAML Artifact (SAMLart). The SAMLart contains encrypted data used to identify the user ID in the back-end server.

An AppWorks Platform client uses SAMLart in any of the following ways.

- [Using SAMLart as a URL parameter](#)
- [Sending SAMLart as a HTTP header variable](#)
- [Sending SAMLart as a cookie](#)

The Identity Framework checks the options used and when a SAMLart is used, the WebGateway will try to resolve it to an organizational user. If it cannot be resolved, a `SAML_ARTIFACT_UNBOUND` SOAP fault is displayed.

## Before you begin:

- You must send an authentication request to AppWorks Platform to get a SAML assertion containing a SAMLart. A sample SAML assertion request and response are as follows:

### SAML Assertion Request:

```
<SOAP:Envelope xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/">
<SOAP:Header>
  <wsse:Security xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-
wss-wssecurity-secext-1.0.xsd">
    <wsse:UsernameToken xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-
200401-wss-wssecurity-secext-1.0.xsd">
      <wsse:Username>jDoe</wsse:Username>
      <wsse:Password>whateverthepasswordisputithere</wsse:Password>
    </wsse:UsernameToken></wsse:Security>
  </SOAP:Header>
<SOAP:Body>
  <samlp:Request xmlns:samlp="urn:oasis:names:tc:SAML:1.0:protocol" MajorVersion="1"
MinorVersion="1"
IssueInstant="2009-04-01T10:23:11Z" RequestID="a997c83a8d-b5d7-b930-edba-
02e37ab1765">
    <samlp:AuthenticationQuery>
      <saml:Subject xmlns:saml="urn:oasis:names:tc:SAML:1.0:assertion">
        <saml:NameIdentifier Format="urn:oasis:names:tc:SAML:1.1:nameid-
format:unspecified">jDoe</saml:NameIdentifier>
      </saml:Subject>
    </samlp:AuthenticationQuery>
  </samlp:Request>
</SOAP:Body>
</SOAP:Envelope>
```

### SAML Assertion Response:

```
<samlp:Response ResponseID="A9D550166-1DC3-4AF4-9C56-271E028C2DFE" MajorVersion="1"
MinorVersion="1" IssueInstant="2009-04-01T11:23:11.679Z" InResponseTo="a997c83a8d-
b5d7-b930-edba-02e37ab1765" xmlns:samlp="urn:oasis:names:tc:SAML:1.0:protocol">
  <samlp:Status xmlns:samlp="urn:oasis:names:tc:SAML:1.0:protocol">
    <samlp:StatusCode Value="samlp:Success"/>
  </samlp:Status>
  <saml:Assertion AssertionID="A4B413744-287C-4A8F-8D0D-C9283F19A339"
MajorVersion="1" MinorVersion="1"
IssueInstant="2009-04-01T11:23:11.679Z" Issuer="https://www.cordys.com/SSO"
xmlns:saml="urn:oasis:names:tc:SAML:1.0:assertion">
    <saml:Conditions NotBefore="2009-04-01T11:18:11.679Z" NotOnOrAfter="2009-04-
01T19:23:11.679Z" xmlns:saml="urn:oasis:names:tc:SAML:1.0:assertion"/>
    <saml:AuthenticationStatement
    AuthenticationMethod="urn:oasis:names:tc:SAML:1.0:am:password"
```

```

AuthenticationInstant="2009-04-01T11:23:11.679Z"
xmlns:saml="urn:oasis:names:tc:SAML:1.0:assertion">
  <saml:Subject xmlns:saml="urn:oasis:names:tc:SAML:1.0:assertion">
    <saml:NameIdentifier Format="urn:oasis:names:tc:SAML:1.1:nameid-
format:unspecified">jDoe</saml:NameIdentifier>
  </saml:Subject>
  </saml:AuthenticationStatement>
</saml:Assertion>
<Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
  <SignedInfo>
    <CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
    <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
    <Reference URI="#A4B413744-287C-4A8F-8D0D-C9283F19A339">
      <Transforms>
        <Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
      </Transforms>
      <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
      <DigestValue>WbVQ557X21U2TcrmQb1hn4yOPkI=</DigestValue>
    </Reference>
  </SignedInfo>

  <SignatureValue>R4fQ+wNie9Rey1+hAcDY3JVz3Dr7rUPOrd4JZuW7vhbTSJFSkaUw+1PYB/18dEJIMTm6
99GwAq+mJ3jIV+ybea7eQ9XQTAfhZppAQnr+6k8kdkQnILliYJLk0WIzOII119OI/vi+AL8PzolYowSQhvru
sK13izKcAk9d+vL+6QY=</SignatureValue>
  <KeyInfo>
    <X509Data>

      <X509Certificate>MIIB4DCCAUmgaWIBAgIQaJsGtYWAXgC78iR/9KXGDDKNBgkqhliG9w0BAQUFADArMQ8
wDQYDVQQKEwZzeXN0ZW0xGDAWBgNVBAbUDk8vbml0b3JAQ05EMTMwMDAeFw0wOTAzMDkxMzEyNTBaFw0xOTA
zMDCxMzEyNTBaMDIxHzAdBgNVBAMTFnNpbmdsZSBzaWduLW9uIHNlcnZpY2UxDzANBgNVBAoTBnN5c3RlbTC
BnzANBgkqhkiG9w0BAQEFAAOBjQAwgYkCgYEaur1RNrLrS9RepnKA1ZMyfzcfv4B5b2NxWDbTxbpWVbJq/p+
Tp9r+akWWiRc20cZQH9esSJ9n3K8KLge/VLUPYSWKJt7P+gqlRaze4a/W51cYAC5QP+U/KP/UJ2csDw1MI0
magDVQZ1fZTjO2s+j9LCcpq00LBzXQTEhmrnZbxECAwEAATANBgkqhkiG9w0BAQUFAAOBgQCm3a4Eg94g9xq
IswNhHu6b+yJpmHO8WDgHdGyZ2kQ0VezBa0ECit57azszco7qG2ZIwnV5WxBYFD+PmcqjcXFhzkviHileZoT
pWFcfEpySvokwlrGz9BDyrn6FyGC3YfZ8N0eeXyJW5AkNAD59CSnEbliLPNON2TQpeOaeL0roUg==</X509C
ertificate>
    </X509Data>
  </KeyInfo>
</Signature>
<samlp:AssertionArtifact
  xmlns:samlp="urn:oasis:names:tc:SAML:1.0:protocol">MDF3eiA3HTMvRoDydILiGLihWu7akA65U
vZ0e0p5hka4siLYQInR/N1C</samlp:AssertionArtifact>
</samlp:Response>
```

The SAML artifact in the response is as follows:

```

<samlp:AssertionArtifact xmlns:samlp="urn:oasis:names:tc:SAML:1.0:protocol">
  MDF3eiA3HTMvRoDydILiGLihWu7akA65UvZ0e0p5hka4siLYQInR/N1C
</samlp:AssertionArtifact>
```

## Using SAMLart as a URL parameter

To use the SAML Artifact as a URL parameter, you must encode it as a URI parameter and then add it to the URL.

### To use the SAML artifact as a URL parameter:

- The following sample JavaScript code encodes the SAMLart and adds it to the URL:

```
function addParameterToURL(url, name, value)
{
    var questionMarkPosition = url.indexOf("?");
    var addAmpersand = questionMarkPosition>=0 &&
questionMarkPosition<url.length-1 && ! /&/.test(url);
    var addQuestionMark = questionMarkPosition<=0;
    if (addQuestionMark) url += "?";
    if (addAmpersand) url += "&";
    url += name + "=" + encodeURIComponent(value);
    return url;
}
// call this function as follows:
addParameterToURL(url, "SAMLart", samlArtifactFromRequest);
```

The sample URL returned is as follows:

```
http://srv-nl-
sso1/test/com.eibus.web.soap.Gateway.wcp?SAMLart=MDF3eiA3HTMvRoDydILiGLihWu7
akA65UvZ0e0p5hka4siLYQInR%2FN1C
```

**Note:** Though you can use this URL in any request to the Web server, from a security point of view, passing a security token in the URL is less secure.

## Sending SAMLart as a HTTP header variable

SAMLart can also be sent as a HTTP header, where the name of the HTTP header must be SAMLart and its value must be the value of the AssertionArtifact in the SAML response.

### To send SAMLart as a HTTP header variable:

- The following sample request demonstrates sending SAMLart as a HTTP header variable.

```
POST https://srv-nl-sso1/home/system/com.eibus.web.soap.Gateway.wcp HTTP/1.1
Host: srv-nl-sso1
Connection: keep-alive
Content-Length: 181
Content-Type: text/xml; charset=UTF-8
SAMLart: MDF3eiA3HTMvRoDydILiGLihWu7akA65UvZ0e0p5hka4siLYQInR/N1C

<SOAP:Envelope xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/">
```

```
<SOAP:Body>
  < GetUserDetails xmlns="http://schemas.cordys.com/1.0/ldap"/>
</SOAP:Body>
</SOAP:Envelope>
```

## Sending SAMLart as a cookie

After a user is authenticated in AppWorks Platform, the SAMLart value is stored in a session cookie. This cookie is sent with each Web service request automatically. As it is a session cookie, it will only be available as long as the session exists. In this context, you must also note that the browser session is shared between different instances of a browser.

If the standard Platform client component is not used, you can also set the SAMLart cookie in JavaScript. However, you must ensure that the name of the cookie is in the format - `<instanceName>_SAMLart`, for example `defaultInst_SAMLart`, and that the value of the cookie is the SAMLart value from the AuthenticationQuery response. Also, ensure that this value is not encoded.

- The following sample JavaScript code demonstrates the process to store a SAML artifact in a cookie:

```
var artifact = "MDF3eiA3HTMvRoDydILiGLihWu7akA65UvZOe0p5hka4siLYQInR/N1C"; // No
encoding done
document.cookie="defaultinst_SAMLart="+artifact+"; path=/ ";
```

When sending SOAP requests to AppWorks Platform, the cookie is sent along with the request automatically.

- If you need to delete the SAMLart cookie in JavaScript, you can use the following function.

```
function deleteSAMLartCookie()
{
    document.cookie="defaultinst_SAMLart=;expires=Thu, 01-Jan-1970 00:00:01 GMT;
path=/ "; // Delete SAMLart cookie by setting date to past
}
```

## Retrieving SAMLart cookie name from SSO

You can also retrieve the name of the SAMLart session cookie from AppWorks Platform using the `GetPreLoginInfo` SOAP request, which returns the name of the cookie as a response in the `SamlArtifactCookieName` tag.

The following sample request and response demonstrate the usage of `GetPreLoginInfo` SOAP request to retrieve SAMLart cookie name.

### GetPreLoginInfo SOAP request

```
<SOAP:Envelope xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP:Body>
    <GetPreLoginInfo xmlns="http://schemas.cordys.com/SSO/Runtime/1.0"/>
  </SOAP:Body>
</SOAP:Envelope>
```

### GetPreLoginInfo SOAP response

```
<GetPreLoginInfoResponse xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/">
  <xmlns="http://schemas.cordys.com/SSO/Runtime/1.0">
    <Authenticator>
      <Type>cordys</Type>
      <MaximizedFrame/>
    </Authenticator>
    <BaseUrlPath>/home</BaseUrlPath>
    <SamlArtifactCookieName>defaultinst_SAMLart</SamlArtifactCookieName> //Name of
the SAMLart session cookie
    <SamlArtifactCookiePath>/</SamlArtifactCookiePath>
    <CheckName>defaultinst_ct</CheckName>
  </GetPreLoginInfoResponse>
```

**Note:** For the correct path to set the SAMLart cookie, check the `SamlArtifactCookiePath` value.

## Custom login options

This topic lists the options for customizing the login page.

### Suppressing the default error messages while signing in

In AppWorks Platform, when a user name is wrongly provided, an alert displays with a message that the user name is invalid. However, in highly secure environments, administrators may not want to display alerts that the user name is invalid. The administrators may want to display a custom message such as 'Invalid Credentials'. AppWorks Platform provides an option to suppress the default alert and give the custom alerts.

The default alerts can be suppressed by adding the following code to the custom login page:

```
CordysRoot.sso.showError(false) // Do not show the error message UFO.
```

### Exception handling

By default, AppWorks Platform captures all the exceptions during the authentication as:

```
User validation failed for:<User Identity>
```

For example, a custom SSO plug-in authenticates the user credentials against an external system other than LDAP. The custom SSO plug-in may have various checks and each of the checks may have a corresponding user-defined exception. The exceptions are provided by the Validator. AppWorks Platform provides the `onafterlogin` event to capture all the user-defined exceptions. The `onafterlogin` event has two fields for handling the exception:

- `responseXML`: Contains the entire the SOAP fault information.
- `customException`: Contains the custom exception as provided by the Validator.

The following is a sample code for capturing the exceptions:

```
function afterLogin(eventObject)
{
if (eventObject.customException)
{
// customException contains exception text that is thrown in the custom
authentication class var firstColonPos = eventObject.customException.indexOf(": ");
var javaTextPos = eventObject.customException.indexOf("at com.validator"); var
errorMsg = eventObject.customException.substring(firstColonPos+2,javaTextPos); alert
(errorMsg);
}
else alert(eventObject.responseXML);
}
```

## Customizing the login page

In case the authentication type is Cordys, the login page can be completely customized. The [AppWorks Developer community](#) provides additional resources that you may find helpful.

## Configuring auto logout

The auto logout feature enables you to automatically log out the user after a certain period of inactivity. If the period is defined as 15 minutes and if a user does not move the mouse or press any key on the keyboard within the AppWorks Platform application, the user is logged out automatically. After the log out, AppWorks Platform desktop is reloaded.

**Note:** The auto logout feature is not supported for standalone form. It only works when using the Start Page. See The Start Page in the *AppWorks Platform Overview Guide*.

### To automatically log out the user:

1. On the **Welcome** page, click  (XMLStore Explorer).  
The XMLStore Explorer window opens.
2. Access **XMLStore > collection > AppWorks Platform > WCP**, right-click **Desktop**, select **New Folder**, and name it **Application**.

**Note:** You must name the folder **Application**. The folder Application is added to the tree.

3. Right-click **Application**, select **New Item**, and name it **cordys\_sso\_AutologoffApp**.  
The item cordys\_sso\_AutologoffApp is added to the tree.
4. Click **cordys\_sso\_AutologoffApp**.  
The **XML Editor - cordys\_sso\_AutologoffApp** dialog box opens.
5. Add the following code in the dialog box for logging out the user after 15 minutes of inactivity:

```
<Application automatic="true" display="hidden" taskbar="false"
    xmlns="http://schemas.cordys.com/task/1.0/"
    xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/">
    <id>cordys_sso_AutologoffApp</id>
    <name xmlns="">SSO Auto logout</name>
    <url>/cordys/wcp/sso/autologout.htm</url>
    <caption>SSO Auto Logoff</caption>
    <description/>
    <info/>
    <data>
        <logoffTimeout>15</logoffTimeout>
    </data>
</Application>
```

**Note:** The `logoffTimeout` tag specifies the time in minutes for the auto logout to be initiated.

6. Click  (Save).  
The auto logout feature is configured.

## Authentication plugins

Authentication of the WS-Security user name token profile is done at the service group level. Before a SOAP request is processed, the WS-Security user name token is processed and authenticated with an authentication plugin.

The responsibility of the authentication plugin is to check the specified credentials against an authentication backend. Examples of the back end authentication: SQL databases, LDAP directories, and Pluggable Authentication Modules (PAM).

By default, every service group uses an authentication plugin that authenticates against the AppWorks Platform LDAP. The default plugins provided with AppWorks Platform are:

- Standard authenticator. See [Configuring a standard authenticator for a service group](#)
- OpenText CARS authenticator. See [Configuring OpenText CARS Authenticator for a service group](#)
- LDAP authenticator. See [Configuring LDAP authenticator for service group](#)

However, you can write custom authentication plugins.

## Writing a Custom Authentication Plugin

If you want to let service groups authenticate against an external backend, you can write a custom authentication plugin. See [Writing an authentication plugin](#) for the procedure to write a custom authentication plugin.

It is important to install the authentication plugin correctly in every service group that must authenticate against the plugin.

## Installing an Authentication Plugin

If you have an authentication plugin and you want service groups to authenticate using that plugin, you must install the plugin and configure the service group to use the plugin.

The Single Sign On component is a service group and must be configured the same way.

See [Installing an authentication plugin](#) for the procedure to install a custom authentication plugin.

## ***Configuring a standard authenticator for a service group***

This topic describes the procedure to configure an authenticator for a service group.

### **Before you begin:**

You must have the system administrator or organizational administrator role to perform this task.

This configuration must be done for every group that handles SOAP requests that use the WS-Security user token profile.

1. On the **Welcome** page > **My Applications**, click  (LDAP Explorer).  
The LDAP Explorer window opens.
2. Navigate to **cordys > <organization>** and select the required service group.  
The service group details are displayed.
3. Click  in the bussoapnodeconfiguration row.  
The XML Editor – Edit XML of bussoapnodeconfiguration window opens.
4. Add the authenticator node by copying the following text in the `<configuration>` tag.  
`<authenticator  
implementation="com.eibus.security.authentication.CARSAuthenticator"/>`
5. Change the authenticator node if you want to use another validator.
6. Click .
7. Restart the service as follows:
  - a. On the **Welcome** page > **My Applications**, click  (System Resource Manager).  
The System Resource Manager window opens.
  - b. On the Service Containers pane, right-click the required **<service container>** and click **Restart**.  
The service group is now configured to handle the SOAP requests using the WS-Security user token profile.

When SSO is being used, usually only the SSO service group needs to be configured for authentication, since all other service groups are accessed with a SAML assertion in the header instead of the user name token.

## **Configuring OpenText CARS Authenticator for a service group**

The OpenText CARS Authenticator class is written for user validation against the AppWorks Platform Admin Repository Server (OpenText CARS). It supports the WS-Security username token profile.

This configuration must be done for every service group that has to handle SOAP requests with WS-Security username token profile as the authentication method.

### **Before you begin:**

You must have the system administrator or organizational administrator role to perform this task.

### **To configure OpenText CARS authenticator for a service group:**

1. On the **Welcome** page > **My Applications**, click  (LDAP Explorer).  
The LDAP Explorer window opens.
2. Navigate to `cordys > <organization>` and select the required service group to be configured with the OpenText CARS authenticator.  
The service group details are displayed.
3. Click  in the `bussoapnodeconfiguration` row.  
The `String (xml) - Edit XML for string` window opens.
4. Add the authenticator node by copying the following text in the `<configuration>` tag.  
`<authenticator  
implementation="com.eibus.security.authentication.CARSAuthenticator"/>`
5. Click .
6. Restart the service as follows:
  - a. On the **Welcome** page > **My Applications**, click  (System Resource Manager).  
The System Resource Manager window opens.
  - b. Right-click the required **<service container>** and click **Restart**.

If you want to adjust the expiry timeframe for the user name tokens that contain a creation timestamp, set the property `bus.identity.wsusername.expirytime` in the `wcp.properties` file.

The service group is now configured to handle the SOAP requests using the WS-Security user token profile for user authentication.

When AppWorks Platform authentication is used, usually only the SSO service group has to be configured for authentication, since all other service groups are accessed using a SAML assertion in the header instead of the user name token. The SSO service group supports WS-Security username token profile by default.

## Configuring LDAP authenticator for service group

You must configure the LDAPAuthenticator for every service group that handles SOAP requests using the WS-Security user token profile against an active directory. The LDAPAuthenticator is an authenticator implementation that provides authentication against the LDAP directories such as Active Directory or ADAM. The authenticate Web service operation is used to create a connection to an Active Directory and checks if the user name and password combination is valid.

### Before you begin:

You must have the system administrator or organizational administrator role to perform this task.

1. On the **Welcome** page, click  (LDAP Explorer).  
The LDAP Explorer window opens.
2. Navigate to **cordys > <organization> ><soap nodes>** and select the service group that must have the LDAP authenticator configured.  
The service group details are displayed.
3. Click  in the bussoapnodeconfiguration row.  
The **String (xml) - Edit XML for string** window opens.
4. Add the authenticator node by copying the following text inside the `<configuration>` tag.

```
<authenticator  
implementation="com.eibus.security.authentication.LDAPAuthenticator">  
<bus.authenticator.ldap.host>dc.acme.com</bus.authenticator.ldap.host>  
<bus.authenticator.ldap.port>3268</bus.authenticator.ldap.port>  
<bus.authenticator.ldap.ssl>false</bus.authenticator.ldap.ssl>  
<bus.authenticator.ldap.bind>NTDOM\{0}</bus.authenticator.ldap.bind>  
</authenticator>
```

### Note:

- If you must log in to AppWorks Platform using the login page but the credentials are to be validated against the Active Directory entries, then the configuration must be changed for the Single Sign-On service group only.
- The bind property is a template that is used for binding to LDAP.
- Replace '{0}' with the user name.
- Define all the properties. If the `bus.authenticator` properties are not defined in the authenticator tag, the default settings specified in the `wcp.properties` file are considered. The sample properties that can be added to the `wcp.properties` file are as follows:

```
bus.authenticator.ldap.host=cnd1123 (replace it with your LDAP hostname)
bus.authenticator.ldap.port=6366
bus.authenticator.ldap.ssl=true
bus.authenticator.ldap.bind=cn={0},cn\authenticated
users,cn\cordys,o\acme.com
```

5. Change the authenticator node if you want to use another validator.
6. Click  (Save).
7. Restart the service as follows:
  - a. On the **Welcome** page, click  (System Resource Manager).  
The System Resource Manager window opens.
  - b. Right-click **<service container>** and click **Restart**.

## **Writing an authentication plugin**

AppWorks Platform Single Sign On provides a set of predefined authentication plugins for AppWorks Platform. You can also add a custom authentication plugin.

AppWorks Platform forwards the validation of the user credentials to the configured custom authentication plugin. See [Installing an authentication plugin](#). The plugin then validates the username and password, and returns true if the credentials are correct; otherwise, it returns false. The logic to validate the credentials is implemented in the Java code and can check a database or an external identity provider, depending on the project requirements.

The custom authentication plugin must implement the Java API  
 com.eibus.security.authentication.Authenticator.

This API contains the following interface:

```
/*
void open(Properties props) throws InvalidAuthenticatorException;

/**
 * Authenticate the credentials with the implementation. Credentials are
 * authentication type specific. It is advised to check whether the
 * implementation understands the credentials given (Use instanceof to check the
 * type)
 *
 * @param credentials
 *          given credentials to authenticate.
 * @return true when the authentication credentials are valid, false when the
 * credentials are not valid
 * @throws InvalidCredentialsException
 */
```

```

 * @throws AuthenticationException
 * @see com.eibus.security.identity.Credentials
 */
public boolean authenticate(Credentials credentials) throws
    InvalidCredentialsException, AuthenticationException;

/**
 * Close the implementation of the Authenticator
 */
void close();
}

```

## ***Installing an authentication plugin***

### **Before you begin:**

You must have an authentication plugin written in Java and packaged in a `.jar` file. The JAR file must exist in the system where AppWorks Platform is running.

If you want to authenticate against an unsupported backend, you can write your own custom authentication plugin. See [Writing an authentication plugin](#) for more information.

1. On the Welcome page, click  (System Resource Manager).  
The System Resource Manager window opens.
2. Double-click the <service container> to which you want to add the custom authentication plugin.  
The Properties - <service container> window opens.
3. Click the **JRE Configuration** tab to open the Java Runtime Environment configuration pane.
4. Add the location of your <custom authentication plugin JAR> file to the classpath field.
5. Click .
6. On the **Welcome** page > **My Applications**, click  (LDAP Explorer).  
The LDAP Explorer window opens.
7. Navigate to <organization> and click the service group that needs to have the authenticator configured.  
The Properties - <service group name> window opens.
8. Click  in the bussoapnodeconfiguration row.  
The String (xml) - Edit XML for string window opens.
9. Copy the following XML to the XML edit field inside the <configuration> tag.  
<authenticator  
implementation="com.example.security.authentication.MyAuthenticator" />

**Note:** Replace `com.example.security.authentication.MyAuthenticator` with the name of your authenticator class.

10. Click .
11. Click  to close the **Properties - <service group name>** window.
12. Click  to close the LDAP Explorer window.
13. On the **Welcome** page > **My Applications**, click  (System Resource Manager).  
The System Resource Manager window opens.
14. Right-click the required <service container> and click **Restart**.

Repeat the above steps for every service you want to configure.

## Key store

Each service group can have a key store. This key store is a container in LDAP in which the private keys of the service group are stored. Private keys are used for signing and encryption of parts of XML data, such as SOAP messages. Someone with the public key can verify that nobody changed the data in transport.

When starting a service group for the first time, the AppWorks Platform Monitor creates a key store for the service group, and generates a private-public key pair. The Monitor also creates a certificate for the public key and signs it.

### Protection of the key store

Protection of the key store is very important because it is used in authentication processes and contains private keys. The key store is stored in LDAP in the service group configuration. To prevent stealing, it is encrypted with a shared key. This shared key is used by all the processes of Monitors in a distributed environment.

To protect the shared key, it is encrypted with the public certificate of the Monitor and stored in the service group configuration of the Monitor. It can only be decrypted with the private key of the Monitor. Only the person who has access to the hard disk of a computer can access the private key of the Monitor. Security can be increased by using file-based protection available in every operating system.

This infrastructure of shared, private, and public keys, and certificates is set up during installation of AppWorks Platform.

### ***Configuring single sign on in distributed installation using key store***

This topic describes the procedure to set up single sign-on in a distributed installation with key stores.

#### **Before you begin:**

- You must have the role of system administrator or organizational administrator to perform this task.

- You must have installed AppWorks Platform on both the computers and one of them as a distributed installation so that it is connected to the primary, that is the first monitor.

### To set up single sign-on in a distributed installation with key stores:

1. Copy the following file from the first AppWorks Platform installation to the second AppWorks Platform installation:  
`<AppWorks Platform_installdir>/certificates/keystore/server1_monitor.p1`
2. Change the following properties in the `wcp.properties` configuration file of the secondary AppWorks Platform installation to the first:  
`bus.keystore.file`  
`bus.keystore.password`  
`bus.keystore.privatekey.password`

Both systems now use the same key store for the monitor, and it can be used for single sign-on in a distributed installation. See [Key store](#).

### ***Creating monitor and SAML2 key infrastructure***

This topic describes how to create a new monitor and SAML2 key pair and key store.

**Note:** This finishes the existing trust relations that depend on the certificate of the monitor.

AppWorks Platform provides the utility to create a monitor and SAML2 key store and trust store.

### **To create a new monitor and SAML2 key store and trust store:**

1. Open a command line on the computer where the Monitor and SAML2 key store and trust store must be recreated.
2. Start the following command from the command line:

- To create key infrastructure for both monitor and SAML2:

```
java com.eibus.security.x509.KeyStoreCreator
```

By default the `KeyStoreCreator` creates a new monitor and SAML2 private key and certificate, and a new default keystore and truststore for the Monitor and SAML2.

- To create key infrastructure for SAML2 only:

```
java com.eibus.security.x509.KeyStoreCreator -saml2_only
```

Using the `-saml2_only` option creates a new private key and certificate, and a new default keystore and truststore for SAML2 integration only.

### **After you complete this task:**

Restart AppWorks Platform.

## ***Creating a monitor key manually***

You can set up the key infrastructure in AppWorks Platform manually. The key infrastructure consists of key stores and the corresponding chain of trust.

**Note:** This is done only when AppWorks Platform must be used with a Certificate Authority (CA) based set-up. Default installations have a self-signed monitor certificate. See [Creating monitor and SAML2 key infrastructure](#) for the procedure to create new key stores for the monitor and the SAML2.

### **To manually create a monitor key:**

1. Depending on the operating system, prepare your environment by setting the location of the OpenSSL configuration, using the following code in the command line:

<b>Windows</b>	set OPENSSL_CONF=<AppWorks Platform_installdir>\Admin Repository Server\bin\openssl.cnf
<b>Linux</b>	export OPENSSL_CONF=<AppWorks Platform_installdir>/Admin Repository Server/bin/openssl.cnf

2. Generate a private key for the monitor, using the following code in the command line:

```
openssl genrsa -des3 -out process_platform_monitor.key 2048
```

A prompt for a pass phrase is displayed.

3. Enter a pass phrase.

A private key for the monitor with a key-size of 2048 bits and 3DES encryption is generated in the file `process_platform_monitor.key`.

**Note:** Make a note of the pass phrase as you will need it while configuring AppWorks Platform to use the new private key.

4. Sign the monitor certificate. Choose between a self-signed certificate or a certificate signed by a third-party CA.

- To create a self-signed certificate:

- a. Run the command:

```
openssl req -new -x509 -days 1825 -key process_platform_monitor.key -out process_platform_monitor.crt
```

- b. Enter the pass phrase of the private key that you configured.

- c. Enter the name of your organization or press **Enter** for the default value.

- d. Enter the name of your organization unit or press **Enter** for the default value.

- e. Enter a common name, for example, a Server name.

**Note:** All fields are mandatory.

- To get a signed certificate from a third-party CA:

- a. Run the following command to generate a certificate signing request:  
`openssl req -new -key process_platform_monitor.key -out <MachineName>_monitor.csr`
- b. Enter the pass phrase of the private key that you configured.
- c. Enter the name of your organization or press **Enter** for the default value.
- d. Enter the name of your organization unit or press **Enter** for the default value.
- e. Enter a common name, for example, a Server name.

**Note:** All fields are mandatory.

A certificate signing request is generated. This request can be sent to a CA for signing. The CA sends back a signed certificate.

- f. Store this certificate as the file `process_platform_monitor.crt`.  
The monitor certificate is signed.
5. Generate a monitor key store in the **PKCS #12** format with the private key in it.
  - a. Enter the following code in the command line:  
`openssl pkcs12 -export -inkey process_platform_monitor.key -in process_platform_monitor.crt -name "monitor@<MachineName>" -out <MachineName>_monitor.p12.`  
The system prompts you for an export password.
  - b. Enter the password again to verify it.
6. Copy the key store file to `<AppWorks Platform_installdir>/certificates/keystore/<MachineName>_monitor.p12`.
7. Configure the `wcp.properties` with the private key and public key passwords. Update the following properties:
  - `bus.keystore.file`
  - `bus.keystore.password`
  - `bus.keystore.certificate.alias`
  - `bus.keystore.privatekey.password`. For more information, see [Authentication and single sign-on properties](#).

#### **After you complete this task:**

Restart AppWorks Platform.

### **Monitor key store**

#### **Monitor Key Store**

Each Monitor has a key store, which is used for the internal security mechanism. When installing a new node a new key store is created for the AppWorks Platform Monitor. When adding a new node to a cluster then the key store must be copied as explained in the Installation Guide.

## Key Store Format

The format of key stores is in the Personal Information Exchange Syntax Standard, also known as the Public Key Cryptography Standards (PKCS) #12 format. This is a file format commonly used to store private keys with accompanying public key certificates, protected with a password-based symmetric key.

## Key Store Usage

The key store is used for the following purposes:

- Signing of internally used SAML assertions by Single Sign On. See [Single Sign On](#).
- Verification of the signature of internally used SAML assertions.
- Protecting internal bus communication.

## Protection of the Key Store

The key store is protected in various ways. The store is encrypted by the Monitor with a protected symmetric key. The details of protection are documented in [Key store](#).

## AppWorks Platform password hashing mechanism through PBKDF2

Password-Based Key Derivation Function (PBKDF2) is a password hashing algorithm. It is a part of the RSA Public-Key Cryptography Standards (PKCS), which is also published as an [RFC 2898](#). Many hashing algorithms used for password storage, such as MD5 and SHA1, are designed for fast computation and therefore are not very effective in preventing password cracking, especially with techniques like [rainbow table](#). The use of key derivation algorithms to construct password hashes reduces the rate at which passwords can be tested. For a list of systems that use PBKDF2, see [PBKDF2](#).

### Algorithm specification

The input parameters of the PBKDF2 are as explained below:

`DK = PBKDF2 (PRF, Password, Salt, C, mkLen)`

Where:

- `DK` is the derived key.
- `PRF` is the pseudo random function. For example: [HMAC](#).
- `Password` is the password of the user.
- `Salt` is the random data used as an additional input to the key derivation function. Using Salt makes it impossible for an attacker to reuse pre-computed hashes against multiple passwords. See [Cryptographic Salt](#) for more information. Further, using Salt results in different hash values for the same password.
- `C` is the number of times the PRF iterates to generate MK.
- `mkLen` is the desired length of the generated key.

The main purpose of the iteration count (C) is to increase the amount of computation needed to derive a key from a password. While this increases the workload of dictionary attacks, it also increases the authentication time of a legitimate user. For more information on how to set the iteration count, see [OWASP password storage cheat sheet](#) and [NIST recommendation for password-based key derivation](#).

The iteration count and the desired length of the derived key can be set with the following properties:

```
bus.authenticator.cars.password.algorithm.PBKDF2WithHmacSHA1.iterations
```

The default number of iterations is 64,000 in 2012, doubling every two years. When the above property is set, the iteration count is fixed to the given value. Otherwise, the algorithm adapts the iteration count over time.

## OpenText Directory Services

OpenText Directory Services (OTDS) is a product for identity management and user authentication across OpenText products. OTDS provides the Single Sign On (SSO) feature for users across the OpenText products. Users can be managed in OTDS, and information about their identity can be exchanged between products that support OTDS. In OTDS, a user can have a different account for each product. It has the functionality to map user accounts between different products. With this, they provide authentication proof for a user account in one product based on the authentication proof of the same user in another product.

When configuring OTDS for AppWorks Platform, the following concepts are important:

Concept	Description
Users	Typically represents a person. The user entry has personal data such as names, addresses, and email address. It has one password using which the user can sign in to all OpenText products configured for an OTDS instance. A user can be a member of a group.
Groups	Logical or functional groups of users such as teams, departments, business units, or people with the same responsibilities, location or line of reports. These groups can then be provided with certain privileges.  In AppWorks Platform, roles provide a coherent set of privileges within a AppWorks Platform application. OTDS does not have the concept of roles. Roles in AppWorks Platform are groups in OTDS. Roles are considered to be managed by AppWorks Platform and must not be created, deleted, or modified in OTDS manually. Instead, you must synchronize them using the OTDS Push roles feature in the Security Administration task after upgrading AppWorks Platform or an application. For more information about synchronizing roles, see <a href="#">Synchronizing roles</a> .

<b>Concept</b>	<b>Description</b>
	When roles are added to OTDS, they are displayed as <code>Package name#Role name</code> . Changing the role name in OTDS blocks it from being synchronized by the push connector. In roles, you can only change its membership.
Partitions	Set of users and groups. A partition can be synchronized or non-synchronized. A synchronized partition is filled with data from an LDAP directory or active directory. With a non-synchronized partition, users and groups are created and managed in OTDS. In both cases, the typical use case of OTDS is to push the data in OTDS to other resources using a push connector.
Resources	OpenText product instance. In this topic, it is assumed that an OTDS resource represents an instance of AppWorks Platform or a tenant, for example, an organization, in AppWorks Platform.
Access roles	Access for an entire partition or to a subset of users and groups from a partition, to a resource.
Push connector	Synchronizes or pushes users and groups within an access role to a resource. These users and groups can then be used in AppWorks Platform. Users can sign in, and users and groups can have tasks assigned to them.
Impersonation	Enables a resource to act on behalf of a user. This is useful for system-to-system integration between OpenText Products. For example, this is used in the integration between OpenText AppWorks Platform and OpenText Content Server when adding a document in Content Server.
Resource activation	Resources can be activated to set up a trust relationship between the resource and OTDS. They agree on a secret by which they can perform certain actions, such as validating credentials and impersonating users.

## Use cases for OTDS

The main use cases for using OTDS with AppWorks Platform are:

- **Externalizing user authentication to OTDS:** Users can use the SSO feature where they only need to sign in to OTDS once instead of signing in to different applications, leading to better security and less passwords in different systems.
- **Providing a single repository for user administration:** AppWorks Platform can leverage a part of the user repository functionality of OTDS. Users must still be available in OpenText CARS, but user passwords are managed through OTDS. OTDS can be configured to synchronize user and group information to OpenText CARS.

## Benefits

Using OTDS in combination with AppWorks Platform offers a set of benefits, including:

- **Single repository for user information:** Maintaining a single repository for user information avoids inconsistencies. OTDS has the functionality to synchronize users across different products.
- **Enterprise synchronization:** OTDS can be integrated with the active directory. This means that administrators can manage users in the active directory, synchronize the users with OTDS and, from there, to all of the connected applications.
- **Reduction in administrative overheads:** Administrators can configure all their related sites or applications to depend on a single Identity Provider (IdP), reducing the maintenance overhead.
- **Better experience for users:** Seamless integration of applications that rely on a single identity provider offers an SSO experience for users.

### Authentication through OTDS

OTDS can be used with AppWorks Platform with the following types of authentication:

- **UI based authentication:** Provides the SSO feature to users. For more information and configuration details, see [Configuring OTDS authenticator for AppWorks Platform](#).
- **Inbound API service calls:**
  - **OTDS token authentication:** Use OTDS tokens for authentication on inbound web-service calls. For details, see [OTAuthentication SOAP header](#).
  - **OAuth authentication:** Use OAuth access tokens for authentication on inbound web-service calls. For details, see [OTDS OAuth integration](#).

### Additional resources

For information about using OTDS with AppWorks Platform, see the following topics:

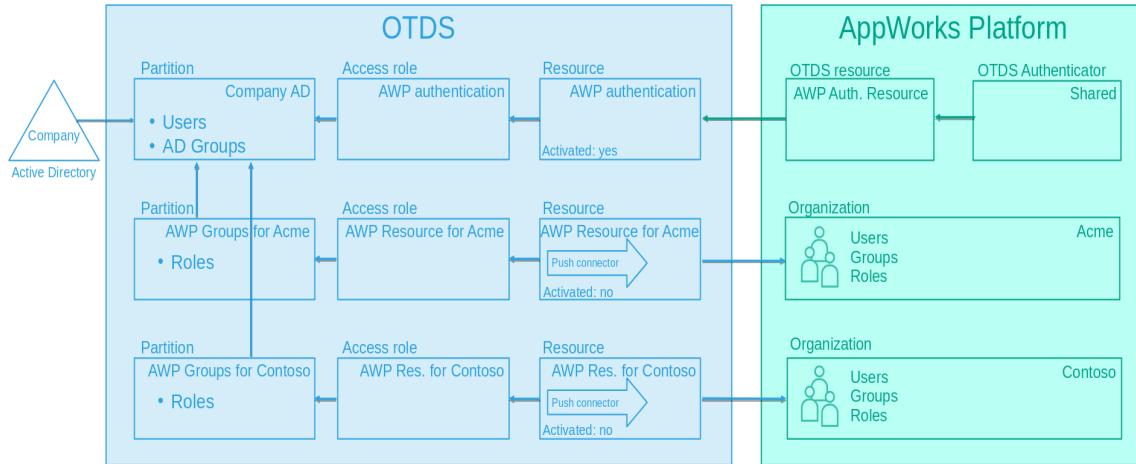
- [Configuring OTDS for AppWorks Platform](#)
- [Configuring OTDS authenticator for AppWorks Platform](#)
- [Configuring OTDS Push connector](#)
- [Synchronizing roles](#)
- [Using OTDS for AppWorks Platform user management](#)
- [Using OTDS with SSL](#)
- [OTDS resources and trust](#)
- [Configuring authentication handler for a repository](#)

### ***Configuring OTDS for AppWorks Platform***

You can configure OpenText Directory Services (OTDS) for AppWorks Platform in multiple ways. This topic describes the preferred way of setting up OTDS for AppWorks Platform, as the number of places where you need to administer on a regular basis is less. There is only one OTDS resource and one OTDS authenticator in AppWorks Platform, and there is a single definition for all the users. While these users can be shared across organizations, there is also the flexibility of assigning users to groups to each organization.

AppWorks Platform has the notion of a shared level and an organization level. In the following image, Acme and Contoso represent different organizations.

**Note:** The names of the organizations used in the image are used for indicative purposes only.



### Use case for configuring OTDS

This use case for configuring OTDS for AppWorks Platform has the following setup:

- The Company AD partition stores all the users and groups of all organizations. The Company AD partition is typically a synchronized partition that is configured on an Active Directory (AD). Based on the setup, the AD groups or distribution lists show up in OTDS as groups. This simplifies integration because adding a user to an AD group automatically reflects in OTDS and AppWorks Platform. Alternatively, if no Active Directory or LDAP is available, a non-synchronized partition is used for user management.
- If you already have an AppWorks Platform instance with users and groups in it, you can run the OTDS Migrator tool to import these users and groups into OTDS. See [Running OTDS migrator for AppWorks Platform](#).
- The AppWorks Platform authentication access role grants users and groups access to an organization through a resource.
- The AppWorks Platform authentication resource enables users to sign in to AppWorks Platform. In AppWorks Platform, this resource is referenced from an OTDS resource, which is again referenced from an OTDS authenticator. The OTDS resource and OTDS authenticator are created only once. They must be created in the shared space because they can be reused by all organizations in AppWorks Platform. As AppWorks Platform and OTDS must communicate to validate the credentials, this resource must be activated.
- The Acme and Contoso organizations in AppWorks Platform both have their own partition, access role, and resource in OTDS.
  - The partition contains groups and roles specific to the organization that can be assigned to users from the company AD partition.

- The access role links the partition to the organization-specific resource.
- The resource can be configured with a push connector to enable users who are members of a group in the organization-specific partition, to become users of the corresponding organization in AppWorks Platform. However, this resource does not have to be activated, because activation is required only for pushing users to AppWorks Platform. It is not required for impersonation and validation of login credentials.

This configuration setup involves the following:

- [Adding the Company AD partition](#)
- [Adding the AppWorks Platform authentication resource](#)
- [Configuring AppWorks Platform for OTDS authentication](#)
- [Synchronizing roles from AppWorks Platform to OTDS](#)

### **Adding the Company AD partition**

#### **To add a partition in OTDS:**

1. Connect to OTDS server using the OTDS Web Administration by launching a browser, type the following URL and sign in with the appropriate credentials:  
`https://<hostname>:<port no>/otds-admin`

For example: `https://localhost:8433/otds-admin`

**Note:** Retrieve the OTDS administrator username and password from the Installation Worksheet used while installing AppWorks Platform, or contact your system administrator.

The OTDS Admin page opens.

**Tip:** Whenever you need additional information in an OTDS Admin page, click Help (?) on the top right of the page to get to an OTDS help page.

2. Click **Partitions**.

The Partitions page displays a list of existing partitions.

**Note:** The `otds.admin` partition was created during the installation.

3. Based on your requirement, do one of the following:

- a. If you have an existing AD/LDAP to use, click **Add > New Synchronized User Partition** and follow the wizard to set it up.
- b. If you do not have an existing AD/LDAP, click **Add > New Non-synchronized User Partition**, and then enter a name and description.

**Note:** The remainder of this document assumes that the name of the partition is Company AD, but you can use another name.

4. Click **Save**.

The partition is added.

5. **Optional, but recommended for non-synchronized user partitions:** For the new partition you created, click **Actions > Password Policy**.

**Note:** If you did not specify a secure global policy, then configure it, or clear the **Use global policy** check box, and set the **Password Quality** and **Security Options** to the required settings for your environment.

6. Click **OK**.

The Partitions dialog box displays the newly created partition.

### Adding the AppWorks Platform authentication resource

#### To create a resource for AppWorks Platform authentication in OTDS:

1. Connect to OTDS server using the OTDS Web Administration by launching a browser, type the following URL, and then sign in as an administrator:  
`https://<hostname>:<port no>/otds-admin`

The OTDS Admin page opens.

2. Click **Resources** on the navigation bar.

The Resources page opens.

3. Click **Add (+)** to create a resource.

The New Resource page opens.

4. For:

Field	Value
Resource name	Type a descriptive name for this resource. For example: AppWorks Platform Authentication Resource
Display Name	<b>Optional.</b> Type a display name for this resource.
Description	<b>Optional.</b> Type a description for this resource. For example: Resource that enables Single sign on for AppWorks Platform
Login UI Version	Retain the default value.
Login UI Style	Leave the box empty.

5. Click **Save**.

The Resource Activation dialog box opens.

6. Highlight and copy the resource identifier and paste it into a text editor or write down this identifier to provide it later when configuring AppWorks Platform to use OTDS for authentication.

7. Click **OK**.

The resource for AppWorks Platform is created in OTDS.

### To allow all users to attempt to login:

1. Connect to OTDS server using the OTDS Web Administration by launching a browser, type the following URL, and then sign in as an administrator:  
`https://<hostname>:<port no>/otds-admin`  
The OTDS Admin page opens.
2. Click **Access Roles** on the navigation bar.  
The Access Role page opens.
3. For the Access to AppWorks Platform Authentication Resource access role, click **Actions > View Access Role Details**.
4. On the User Partitions tab, click **Add** to add a trust relation between the access role and the partition.
5. Select the **Company AD** partition, and then **click Add Selected Items to Access Role**.
6. Click **Close Dialog**, and then click **Save**.

### To enable system-to-system integration to other OpenText Products:

1. Connect to OTDS server using the OTDS Web Administration by launching a browser, type the following URL, and then sign in as an administrator:  
`https://<hostname>:<port no>/otds-admin`  
The OTDS Admin page opens.
2. Click **Resources** on the navigation bar.  
The Resources page opens.
3. For the Access to AppWorks Platform Authentication Resource access role, click **Actions > Impersonation Settings**.
4. Select **Allow this resource to impersonate its own users**.
5. Click **OK** to save the change.

## Configuring AppWorks Platform for OTDS authentication

This section describes the configuration steps required to enable OTDS authentication for AppWorks Platform. It involves configuring the following:

- AppWorks Platform OTDS authentication
- AppWorks Platform Trusted Sites

## AppWorks Platform OTDS authentication

To use OTDS for authentication in AppWorks Platform, you must create an authenticator and set up a trust relationship (activated resource) between OTDS and AppWorks Platform. This trust is used in the authentication process of users and works only for those users that have access to AppWorks Platform.

### To configure AppWorks Platform OTDS authentication:

1. Open a browser and access **AppWorks Platform Explorer > Security Administration**.  
The Security Administration page opens.
2. Click the **OTDS Resources** tab.  
The OTDS Resources page opens.
3. Click **+** (Add).  
The OTDS Resources Details information displays.
4. For:

Field	Value
Resource Name	Type the name with which the OTDS resource must be identified, for example: AppWorks Platform Authentication Resource.
Space	Select Shared from the list. <b>Note:</b> This is the space for which the OTDS Resource is active. An OTDS resource in the Shared space can be used in all organizations. An OTDS resource in an Organization space can only be used in that specific organization.
OTDS Server URL	Type the URL to access the OTDS Server in the following format: <code>https://&lt;computer name&gt;:&lt;port number&gt;</code> This must be a valid URL, starting with either http or https, for example: <code>https://server1:8443</code>
Resource ID	Type the resource identifier of the resource previously created as AppWorks Platform Authentication Resource.

5. Click **Save**, and then click **Activate**.  
A Notification page opens indicating that the resource was activated.
6. Click **Close** on the message notification.
7. Switch to the tab **Authenticators > Shared**.  
The Shared Authenticators page opens.
8. In the Shared Authenticators table, click **+** (Insert).  
The Authenticator Details page opens.

9. For:

Field	Value
Id	Type an ID for the authenticator. For example: otds
Default or Test Only	Select the Default check box.
Type	Select OTDS Authenticator from the list.

**Note:** Leave all other values or modify as needed.

10. On the OTDS Properties tab, click **Select Resource**.

The OTDS Resources page opens.

11. Select the **AppWorks Platform Authentication Resource** resource, and then click **OK**.

On the Authenticator Details page, the OTDS Properties section automatically populates with the selected resource information.

12. **Optional:** For Public OTDS login URL, copy the OTDS Server URL value and append with /otdssws/login. This step is required only when the URL by which OTDS is exposed to users differs from the URL by which AppWorks Platform communicates to OTDS, such as in an SSL offloader or (reverse) proxy scenario.

For example: <https://public.otds.company.com/otdssws/login>

13. Click **Save**.

**To assert the activation of the resource in OTDS:**

1. Connect to OTDS server using the OTDS Web Administration by launching a browser, type the following URL, and then sign in as an administrator:

<https://<hostname>:<port no>/otds-admin>

The OTDS Admin page opens.

2. Click **Resources** on the navigation bar.

The Resources page opens.

3. For the resource PP Authentication Resource, click **Actions > Activation Status**.

The Resource Activation page opens.

4. Click **Verify Activation**.

A green box displays stating that the resource is activated.

5. Click **OK**.

### AppWorks Platform Trusted Sites

For authentication, the domain level URL for AppWorks Platform must be registered as a trusted site in OTDS. This is to ensure that OTDS can safely redirect to AppWorks Platform after the user has been authenticated.

Without configuring this, users are likely to receive an error "Not a trusted referral site, please contact your administrator" after the user has been authenticated and before returning to the originally requested application.

#### To add a trusted site:

1. Connect to OTDS server using the OTDS Web Administration by launching a browser, type the following URL, and then sign in as an administrator:  
`https://<hostname>:<port no>/otds-admin`  
The OTDS Admin page opens.
2. Click **Trusted Sites**.  
The Trusted Sites page opens.
3. Click **Add (+)** to add a trusted site.
4. For **Addresses Directory Services will redirect requests to**, type an appropriate URL in the format `<protocol>://<FQDN>:<ProcessPlatformPort>`.  
For example: `https://ProcessPlatformServer.domain.suffix:81`
5. Click **Save**.

### Configuring resources per organization

You can configure the resources created for AppWorks Platform authentication in OTDS at a shared-level or an organization-level. The following section describes the procedure to configure resources at an organization-level.

- [Creating an organization-specific partition for groups and roles](#)
- [Optional: Importing users from an existing AppWorks Platform instance](#)
- [Creating a user to push users and groups into AppWorks Platform](#)
- [Creating a resource for the organization in OTDS](#)
- [Synchronizing roles from AppWorks Platform to OTDS](#)
- [Linking the partition to the resource](#)
- [Managing users and group-membership for a specific organization](#)
- [Consolidating the resource](#)

**Note:** The instructions in this section assume the name of the organization as Acme as specified in the use case above.

#### Creating an organization-specific partition for groups and roles

1. Connect to OTDS server using the OTDS Web Administration by launching a browser, type the following URL, and then sign in as an administrator:  
`https://<hostname>:<port no>/otds-admin`  
The OTDS Admin page opens.
2. Click **Partitions** on the navigation bar.  
The Partitions page opens.

3. Click **Add > New Non-synchronized User Partition**.
4. Type a name for the partition. For example: AppWorks Platform Groups for Acme
5. Click **Save**.

The partition is added.

#### **Optional: Importing users from an existing AppWorks Platform instance**

If you opted for a non-synchronized user partition (because you do not have an AD/LDAP Directory), but your instance of AppWorks Platform is already has users and groups, you can use the OTDS Migrator tool to import users and groups into OTDS. Ensure that you specify the Company AD partition as the partition for users, and the AppWorks Platform Groups for Acme partition as the partition for groups, and provide the organization name Acme.

**Important:** When you have used the OTDS Migrator, you must ensure that the 'principal attribute' of the AppWorks Platform Authentication Resource is set to **oTExternalID1**.

#### **To set the principle attribute:**

1. Connect to OTDS server using the OTDS Web Administration by launching a browser, type the following URL, and then sign in as an administrator:  
`https://<hostname>:<port no>/otds-admin`  
The OTDS Admin page opens.
2. Click **Resources** on the navigation bar.  
The Resources page opens.
3. Search to find **AppWorks Platform Authentication Resource**, and then click **Actions > Properties**.
4. On the **Principal Attribute** tab, for Authentication principal attribute, select **oTExternalID1**.
5. Click **Save**.

#### **Creating a user to push users and groups into AppWorks Platform**

##### **To create and configure a user for the OTDS push connector:**

1. Open a browser and access **AppWorks Platform Explorer > User Manager**.  
The **User Manager** window opens.
2. Click + (Add a User) on the toolbar.  
The Create User page is displayed.
3. For:

Field	Value
Authentication Type	Select <b>Cordys</b> .

Field	Value
User Name	Type the unique name of the user.
User Full Name	Type the full name of the user.
User ID	<p>Type the operating system identity of the user in the User ID field. This acts as a unique user identifier, which is used to sign in to AppWorks Platform.</p> <p>Note: If the User ID of an organizational user being created already exists (because you have set up another organization already), you can map it to the existing authenticated user in the dialog box that displays.</p>
Password	Type the password for the user.
Confirm Password	Confirm the password.

4. Click **Save**.

The user is created and displayed on the Users pane. Ensure that the user is selected.

5. Select **Include Internal Roles**.

The Roles pane on the right displays the available roles.

6. On the Roles pane, click **Search** and type **OTDS Push Service**.

7. Right-click **OTDS Push Service**, and then select **Assign to selected User(s)**.

The OTDS Push Service role is assigned to the created user.

### Creating a resource for the organization in OTDS

**Important:** During the resource creation, do not click Save until it is explicitly stated. You must wait until all parameters are configured because OTDS schedules a synchronization task after saving a resource. Synchronization without everything configured properly might result in duplicate or lost data in AppWorks Platform.

#### To create and configure a resource for the organization in OTDS:

1. Connect to OTDS server using the OTDS Web Administration by launching a browser, type the following URL, and then sign in as an administrator:  
`https://<hostname>:<port no>/otds-admin`

The OTDS Admin page is displayed.

2. Click **Resources** on the navigation bar.

The Resources page is displayed.

3. Click **+** (Add) to create the resource.

The New Resource page is displayed.

## 4. For:

Field	Value
Resource name	Type a descriptive name for this resource. For example: AppWorks Platform Resource for Acme
Display Name	Optional. Type a display name for this resource.
Description	Optional. Type a description for this resource. For example: Resource that pushes users and groups for organization Acme
Sign in UI Version	Retain the default value.
Sign in UI Style	Leave the box empty.
Sign out URL	Leave the box empty.
Sign out Method	Leave the box empty.

5. Click **Next**.

The New Resource > Synchronization tab is displayed.

## 6. For:

Field	Value
User and group synchronization	Select this check box.
Synchronization connector	Select REST (Generic) connector from the list.
This connector will	<p>Leave the selected defaults:</p> <ul style="list-style-type: none"> <li>■ Create users and groups</li> <li>■ Modify users and groups</li> </ul> <p><b>Note:</b> Open Text recommends not selecting the Delete users and groups check box in production environments.</p>

7. Click **Next**.

The New Resource > Connection Information tab is displayed.

## 8. For:

Field	Value
Base URL	Type the AppWorks Platform server URL to access AppWorks Platform.

Field	Value
	<p>The URL comprises the protocol, host name, and port of the AppWorks Platform server, including the (URL encoded) name of the organization to push to, and the fixed name of the push connector application (otdspush).</p> <p>For example:</p> <pre>https://&lt;my process-platform-server.domain.com&gt;/home/Acme/app/otdspush https://&lt;my process-platform-server.domain.com&gt;/home/acme%20org/app/otdspush</pre> <p><b>Note:</b> You can retrieve the URL encoded name of the organization by signing in to AppWorks Platform and checking the browser URL.</p>
Username	Type the username of the user created for the OTDS push connector.
Password	Type the password of that same user.

9. Click **Test Connection** to verify that the details provided are correct and a connection between AppWorks Platform and OTDS can be established.  
When successful, a Connected Successfully message is displayed.
10. Click **Next**.  
The New Resource > User Attribute Mappings tab is displayed.  
**Note:** If the page does not display any values, click **Reset to Default**.
11. Accept the default values, and then click **Next**.  
The New Resource > Group Attribute Mappings tab is displayed.  
**Note:** When you use the OTDS Migrator tool, ensure that the OTDS attribute for the 'NAME' resource attribute is set to oTExternalID1.
12. Accept the default values, and then click **Save** to create the resource.  
The dialog box displays the resource identifier.
13. Click **OK** to dismiss this dialog box because resource activation is not required for the push connector.

The resource for the Acme organization in AppWorks Platform is created in OTDS.

## Synchronizing roles from AppWorks Platform to OTDS

To assign roles to users in OTDS, the package roles from AppWorks Platform must be imported into OTDS.

See [Synchronizing roles](#). Enter the name of the organization Acme, and then enter the name of the partition AppWorks Platform Groups for Acme.

## Linking the partition to the resource

To provide users access to an organization, they need to be linked to the organization-specific resource. This can be achieved in multiple ways, but the preferred way is to add the entire AppWorks Platform Groups for Acme partition to the access role of the organization-specific resource. After that is arranged, you can set up group-membership within the partition.

### To add the partition to the access role:

1. Connect to OTDS server using the OTDS Web Administration by launching a browser, type the following URL, and then sign in as an administrator:  
`https://<hostname>:<port no>/otds-admin`  
The OTDS Admin page is displayed.
2. Click **Access Roles** on the navigation bar.  
The Access Roles page is displayed.
3. Search to find **Access to AppWorks Platform Resource for Acme**.
4. If the role is not included, then click **Actions > Include groups**.
5. Click **Actions > View Access Role Details**.  
The Access Role Details page is displayed.  
This view contains several tabs for the kind of types that can be granted access to the resource: User Partitions, Organizational Units, Groups, and Users.  
The Resources tab provides information on the resources to which this access roles grants access. This can remain limited to the AppWorks Platform Resource for Acme resource only.
6. Navigate to the User Partitions tab, and then click **Add**.  
The Add Partitions - Access to AppWorks Platform Resource for Acme page is displayed.
7. Search to select the AppWorks Platform Groups for Acme partition, and then click **Add Selected Items to Access Role**.
8. Click **Close Dialog** to close the dialog box.
9. Click **Save**.

## Managing users and group-membership for a specific organization

After the AppWorks Platform Groups for Acme partition are added to a resource, you can add user and groups to the organization as follows:

- Adding a user to a group in the partition.
- Adding a group to another group in the partition.
- **Not recommended:** Adding a user to the access role directly. Read the recommendation at the end of this section.

**To add a user or group to a group:**

1. Connect to OTDS server using the OTDS Web Administration by launching a browser, type the following URL, and then sign in as an administrator:  
`https://<hostname>:<port no>/otds-admin`  
The OTDS Admin page opens.
  2. Click **Partitions** on the navigation bar.  
The Partitions page opens.
  3. Search to find the AppWorks Platform Groups for Acme partition, and then click Actions > **View Members**.  
The list of users is displayed.
  4. Search to find the required group to add members, and then click **Actions > Edit Membership**.  
The membership page opens, displaying a list of all the current members.
  5. Click **Add Member**.  
The Users and Groups Associations dialog box is displayed.
  6. Search for users in the Company AD partition, select the corresponding check box, and then click **Add Selected**.
  7. Search for groups in the Company AD partition, select the corresponding check box, and then click **Add Selected**.
- Important:** Ensure that you select the groups from the AppWorks Platform Groups for Acme and Company AD partitions only, and not the groups from other AppWorks Platform Groups partitions, such as AppWorks Platform Groups for Contoso.
8. After adding all users and groups, click **Close** to close the dialog box.

**Tip:** In the unlikely event of the requirement to have additional users in AppWorks Platform without any role in an organization, you can follow the steps above to add a user to an access role. However, to simplify and reduce overhead, the following is recommended:

1. Create an extra group called Guest users in the partition AppWorks Platform Groups for Acme, which is a part of the access role already.
2. Add the users, who do not have a role in AppWorks Platform, to that group.

## Consolidating the resource

When a change is made to a user, group, or resource, changes must automatically be pushed to AppWorks Platform. If you realize that users are not automatically pushed to AppWorks Platform, or you want to ensure that the changes are pushed, you can consolidate a resource.

**To consolidate a resource:**

1. Connect to OTDS server using the OTDS Web Administration by launching a browser, type the following URL, and then sign in as an administrator:  
`https://<hostname>:<port no>/otds-admin`

The OTDS Admin page is displayed.

2. Click **Resources** on the navigation bar.  
The Resources page is displayed.
3. For the AppWorks Platform Resource for Acme resource, click **Actions > Consolidate**.
4. Click **Consolidate**.

The consolidation is triggered, but depending on the number of users and groups, it might take some time. Consolidation checks every user and group, and pushes updates to AppWorks Platform.

**Note:** Consolidating changes can be done on different levels in OTDS such as the partition level, user level, and group level. For more information on consolidation, click **Help (?)** at the top right corner of the OTDS Admin page.

## OTDS variables

This topic describes the available variables you can use in an OTDS-type authenticator.

### Resolving variables

Use variables while configuring an authenticator. You can add them at the authenticator level, organization level, and shared level. The resolve algorithm for a variable searches for authenticator-specific variables first. If it is not found at the authenticator level, a search is performed at the organization level. If it is not found at both authenticator and organization level, it searches in the shared level variables.

### OTDS variables

The following variables can be specified:

Variable	Description
BASE_URL	Legacy variable that must be left blank. The public URL configured on the Web server is used to sign in or sign out from AppWorks Platform.
IDP_RETURN_URL	When this variable is set, the AppWorks Platform OTDS TCS redirects to this URL after validating the POSTed OTDS ticket from OTDS. That is, after the user signs in to OTDS, the browser is redirected to the URL as specified in this variable. The IDP_RETURN_URL is used when the <b>No Frame</b> option in FrameOptions is set. AppWorks Platform IDP integration is browser based. All protocol redirects are done through the browser. When the <b>No Frame</b> option is used, the complete browser is redirected to the OTDS server URL. This means that the AppWorks Platform context is completely lost in the browser. After you have authenticated in OTDS, an OTDS ticket is POSTed to the AppWorks Platform

Variable	Description
	<p>OTDS TCS. After validation, the TCS redirects the browser back to the URL configured in <code>IDP_RETURN_URL</code>.</p> <p><code>IDP_RETURN_URL</code> can also be used when you do not want to redirect to the Welcome page, but to another page, such as an application XForm. The value specified must be the part after the domain, for example, <code>/home/myorg</code> or <code>/home/myorg/com/acme/app/myform.caf</code>.</p>

## Configuring OTDS Push connector

The AppWorks Platform OTDS push connector can be used to synchronize users and groups to OpenText CARS and the Identity entities. See the AppWorks Platform Low-Code Design Guide for more information on managing Identity components. An OTDS push connector configuration manages the synchronization for one specific organization in AppWorks Platform. The generic OTDS REST push connector is used for synchronization.

**Note:** The Push Connector updates data in OpenText CARS and Identity components only for those attributes that are specified in the User and Group Attribute Mapping of the resource in OTDS. Any data in OpenText CARS and Identity components that is not mapped is unchanged.

### Before you begin:

- Ensure that OTDS is installed.
- If OTDS and/or AppWorks Platform is configured with SSL, follow the instructions in [Using OTDS with SSL](#) to configure the trust between AppWorks Platform and OTDS.

### To configure the AppWorks Platform Push connector in OTDS:

#### In AppWorks Platform:

1. On the AppWorks Platform explorer Welcome page, click **User Manager**. The User Manager window opens.
2. Create a new user with the Authentication Type as **Cordys**. See [Creating users](#)
3. Assign the OTDS Push Service role. See [Assigning roles to users](#).
4. Click **Include internal roles** to see this role.

#### In OTDS:

1. Access the OTDS Web administration client.
2. Create a new Resource and enable User and group synchronization.
3. Select the **REST (Generic)** connector for synchronization.
4. Enter the AppWorks Platform server URL, and provide the username and password as created in previous steps.

The URL comprises the protocol, host name, and port, including the URL-encoded name of the organization to push to, and the fixed name of the push connector application **otdpush**.

**For example:** `https://process-platform-server.domain.com/home/acme%20corp/app/otdpush`

**Note:** The above URL must point to a single node, which can be one of the cluster nodes, and must not point to a load balancer.

5. Click **Test Connection** to check the connectivity.
6. **Optional.** You can change the User and Group Attribute mappings if you want to use a different format or mapping.

**Note:** To map a custom property of the Person entity from identity components to an OTDS attribute, click **Reset To Default** on the User Attribute Mappings page. A newly added custom attribute (with Identity-PropertyName) is displayed in User Attribute Mappings. You can map this custom attribute to the required OTDS attribute and its format. Click **Consolidate** to view the changes in identity runtime.

**Note:** The following characters are not supported: `,=+<>#;"\V:*?'&|`. For groups, a default mapping of **-and-** is made for **&**. This format mapping can be changed or similar format mappings can be made for non-supported characters.

```
%js:function format(name) { return name.replace(/&/g, "-and-"); }
```

7. Save the resource. Activation of the resource is not mandatory. Therefore, you can cancel the Resource Activation window that opens.
8. Open Access Roles, find the corresponding access role and click **Actions > View Access Role Details**.
9. Add the Partitions you want to synchronize.
10. If you want to synchronize the Groups from the Partition as well, click **Actions > Include Groups from OUs**.

The OTDS Groups are organizational roles in AppWorks Platform.

**Note:** To force a synchronization of the users and groups, you must click **Actions** on the required resource, partition, user, or group. From the Consolidate options, select **Verify and repair**, and then click **Consolidate**.

## Synchronizing roles

This topic describes the procedure to synchronize AppWorks Platform package roles with OTDS.

### Before you begin:

- Ensure the following:
  - OTDS is installed and configured with AppWorks Platform. See the AppWorks Platform Installation Guide on the Knowledge Center for details.
  - the user using this functionality has the Security Administrator role.

- a Platform resource is defined and activated. See [Managing OTDS platform resources](#).
- If OTDS is configured with SSL, see [Using OTDS with SSL](#) to configure the trust between AppWorks Platform and OTDS.

Functional and application roles from packages that are available in the shared space and in the specified organization context are created in OTDS.

The package roles in AppWorks Platform are created in the <Package Name>#<Role Name> format, for example, Case Management#Case Worker.

### **One partition per organization**

OpenText recommends that you use only one OTDS partition per organization in AppWorks Platform. This partition contains all package roles that you can use in that organization. You can also create groups in this partition. The administrator can assign you to these groups; this adds you as a user in that organization and assigns the role to you. Only users who are members of one or more groups in this partition are created in the AppWorks Platform organization.

### **Granting administrator permissions**

**Note:** The procedure on granting administrator permissions is applicable only if you use OTDS 16.6 or later versions.

To enhance security, the AppWorks Platform Authentication resource principal user must be granted administrator permissions on the partition where the roles are created.

#### **To grant permissions on the partition:**

1. Connect to OTDS server using the OTDS Web Administration by launching a browser, type the following URL, and then sign in as an administrator:  
`https://<hostname>:<port no>/otds-admin`  
The OTDS Admin page is displayed.
2. Click **Partitions** on the navigation bar.  
The Partitions page is displayed.
3. Search to find the required AppWorks Platform resource, for example **AppWorks Platform Resource for Acme**.
4. Click **Actions > Edit Administrators**.  
The list of administrators is displayed.
5. Click **Add Administrator**.  
The Member Selection dialog box is displayed.
6. Search for **AppWorks Platform Authentication** to find a user with a UUID as ID and AppWorks Platform Authentication as the display name. Select the corresponding check box, and then click **Add Selected**.
7. Click **Save**.

The AppWorks Platform Authentication resource principal user is granted permissions on the partition

## Synchronizing roles

### To synchronize roles from AppWorks Platform to OTDS:

1. On the Welcome page, click  (Security Administration).  
The Security Administration window opens and the **Certificates** tab is displayed by default.
2. Click the **OTDS Resources** tab.
3. Click the **OTDS Push Roles** tab.  
The OTDS push role page opens and the OTDS server URL is prefilled from the configured Platform resource.
4. Type a name for the OTDS partition in OTDS partition name.
5. Optional. Select **Delete roles** if you want the package roles not found in AppWorks Platform to be deleted from OTDS.

**Tip:** This is useful when you renamed a role. Since the OTDS Push Roles feature does not support renaming roles, you can first push without selecting the Delete roles option, copy the membership from the old role to the new role, and then push the role again with the Delete roles option selected.

6. Click  to save the configuration.
7. Click **Push roles**.

The status field displays the progress of the role synchronization. You can click  to obtain the latest status and to see if the role synchronization is complete.

## Running OTDS migrator for AppWorks Platform

This topic describes the procedure to migrate users, roles, and membership data from AppWorks Platform to OTDS.

### Before you begin:

- Ensure that OTDS is available. For more information, see the **AppWorks Platform Installation Guide** on My Support.
- If OTDS is configured with SSL, configure the trust between AppWorks Platform and OTDS. See [Using OTDS with SSL](#).
- If you have an OTDS resource configured for user synchronization to AppWorks Platform, see [Configuring OTDS for AppWorks Platform](#), and ensure that it is disabled. Do not enable it until the migration to OTDS is complete.
- Before starting the migration, OpenText recommends that you create a backup of both the OTDS server and OpenText CARS. The OTDS Migrator inserts, updates, and moves data in both OTDS and OpenText CARS. The best configuration for migration is deployment specific.
- If you are using the OTDS Migrator with the property `otdsmigrator.otds.migrate.passwords=true`, ensure that you can log in with

credentials with administrator rights in AppWorks Platform, through OTDS authentication. After running the OTDS Migrator, you can no longer log in to AppWorks Platform with AppWorks Platform authentication. You might choose to skip migration of some users through the property `otdsmigrator.excluded.os.identities`. For information about setting OTDS authentication, see [OpenText Directory Services](#).

The OTDS Migrator (`otdsmigrator`) is a command line tool that enables mass migration of existing users, package roles, organizational roles, and role memberships from a specific organization in AppWorks Platform to OTDS. The tool performs the following operations:

- OTDS partitions for users and groups are created, if needed. OpenText recommends that you use a single shared partition for OTDS users and separate organization-specific partitions for OTDS groups.
- The corresponding OTDS users for AppWorks Platform users, in the specified organization, are checked for existence and added, if required. Optionally, the user password hashes found in AppWorks Platform can be migrated to their corresponding OTDS users.
- The corresponding OTDS groups for AppWorks Platform package roles, of either functional or application type, are looked up and added or updated, if required.
- The corresponding OTDS groups for AppWorks Platform organizational roles, of functional type, are checked for existence and added, if required.

The corresponding OTDS group memberships for AppWorks Platform role assignments of these package roles, organizational roles, and organizational users are checked for existence and added, if required.

#### To run the OTDS migrator:

1. Verify if OpenText OTDS Platform configurator application package is already deployed. If not, deploy it using the Application Deployer task from the Welcome page. See [Deploying applications](#)
2. Ensure that the `CORDYS_HOME` environment variable is set to `<AppWorks_Platform_installdir>`.
3. Ensure that the user running the tool is a user in AppWorks Platform. There must be a user with the User ID set as the user name of the operating system user account.
4. Navigate to `CORDYS_HOME/config`, create and edit the `otdsmigrator.properties` file to set the parameters according to the Properties table.

**Note:** Ensure that the `otdsmigrator.properties` file does not have read access permission for untrusted users, as it contains the OTDS administrator password.

5. Open a command prompt, navigate to `<CORDYS_HOME>/bin` and run `otdsmigrator.cmd` or `otdsmigrator.sh`
6. Optionally, override any of the properties in `otdsmigrator.properties`, run `otdsmigrator.cmd` or `otdsmigrator.sh` with the `-overrides <property file path>` command line option.

#### Relationship with the OTDS Push Roles feature

After the migration, OTDS becomes the authoritative source for AppWorks Platform users and organizational roles, while AppWorks Platform remains the authoritative source for package roles (which are then synchronized with OTDS by using the OTDS Push Roles feature from the Security Administration task). Therefore, OTDS must be used to manage these users and groups.

Use the `otdsmigrator` tool for the migration of data to OTDS. Although the tool can be run more than once, it does not modify any users or organizational roles that already appear in OTDS.

### **Configuration file and command line**

By default, the `otdsmigrator` tool is configured by the property values specified in the file called `otdsmigrator.properties` in the `config` subdirectory of the AppWorks Platform installation. Each of the property values specified in `otdsmigrator.properties` can be overridden by a property value from a file named by the command line option `-overrides <filename>`. In a multi-organization AppWorks Platform deployment, each organization must have its own overriding property file. No further command line options or arguments are defined.

### **Important: Use `oTExternalID1` for `__NAME__` User Attribute Mapping in Push Connector configuration**

To prevent an OTDS Push connector, configured for a resource, from creating new users with a name like `<user-id>@<partitionname>`, the Resource Attribute '`__NAME__`' in the User Attribute Mappings of that resource must be set to **`oTExternalID1`**.

### **Rejections listing**

If OTDS rejects any group, user, or membership, its name, along with the reason for rejecting it, is appended to a text file in the Logs subdirectory of the AppWorks Platform installation. By default, the name of this file is `otdsmigrator.rejections`. In addition, rejections are reported to the console. The occurrence of any rejection causes the `otdsmigrator` tool to report a non-zero exit status at the end of the migration. The system administrator can take action if required, and rerun the `otdsmigrator` tool.

### **Properties**

The `otdsmigrator` tool has the following ISO-8859-1-compliant properties:

Name	Description
<code>otdsmigrator.otds.url</code>	URL of the OTDS server. Default value: <code>http://localhost:8080</code> <b>Note:</b> The <code>otdsmigrator.otds.url</code> must be URL-encoded.
<code>otdsmigrator.otds.username</code>	User name for administrative access to the OTDS server. Default value is <code>otadmin@otds.admin</code> .

Name	Description
otdsmigrator.otds.password	Password for administrative access to the OTDS server.
otdsmigrator.processplatform.organization	<p>AppWorks Platform organization name from which to migrate.</p> <p>Example: for the organization name 東京, use the following: otdsmigrator.processplatform.organization=\u6771\u4eac.</p>
otdsmigrator.otds.groups.partitionname	OTDS target partition name for groups.
otdsmigrator.otds.users.partitionname	OTDS target partition name for users.
otdsmigrator.otds.migrate.passwords	Indicates migration of password hashes from AppWorks Platform to OTDS. Default value is false.
otdsmigrator.rejections.filename	Name of rejections file under AppWorks Platform's Logs directory. Default value is otdsmigrator.rejections.
otdsmigrator.excluded.os.identities	Comma-separated list of OS identities that must not be migrated.
otdsmigrator.mappings.os.identity.to.user	<p>Derivation of the OTDS user name from the AppWorks Platform user's OS identity. Default value: beforeLastAtSign</p> <p>The following are the possible values for otdsmigrator.mappings.os.identity.to.user:</p> <ul style="list-style-type: none"> <li>▪ <b>beforeLastAtSign:</b> All characters before the last '@' (all characters if no '@' is found)</li> <li>▪ <b>afterLastBackslash:</b> All characters after the last '\' (all characters if no '\' is found)</li> <li>▪ <b>fullIdentity:</b> All characters</li> </ul>
otdsmigrator.mappings.ot.telephone.number.index	Index in OpenText CARS telephoneNumbers field for OTDS oTTelephoneNumber field. Default value is 0.
otdsmigrator.mappings.ot.home.phone.index	Index in OpenText CARS telephoneNumbers field for OTDS oTHomePhone field. Default value is -1 (not mapped).

Name	Description
otdsmigrator.mappings.ot.mobile.index	Index in OpenText CARS telephoneNumbers field for OTDS oTMobile field. Default value is 1.

### Using the OTDS migrator in a high availability deployment scenario

To migrate the users, roles, and membership data from AppWorks Platform to OTDS, in a High Availability AppWorks Platform environment, select the node in the cluster, on which you want to run the OTDS Migrator.

### Troubleshooting

If you face issues when running the OTDS Migrator, an error message is displayed where you ran the OTDS Migrator. Use the following files from the AppWorks Platform Logs directory to analyze the issues:

- Rejections file
- OTDSMigrator.xml

### Using OTDS for AppWorks Platform user management

Every user must have a role in AppWorks Platform to access it, use some of its services and applications, and work in one or more organizations. The user's information is stored in OpenText CARS.

User management is an administration process that controls a user's access to a specific service or application as well as the usage of a specific functionality. Therefore, based on the authentication requirements, it is an administrator's responsibility to perform the user management tasks such as create, update, and delete a user.

AppWorks Platform supports the following authentication types:

Authentication type	Description
AppWorks Platform	Authentication is done in AppWorks Platform. Usernames and password hashes are stored in OpenText CARS.
Web application server	Authentication is done by the Web application server. AppWorks Platform relies on the Web application server to provide a trusted identity. Authentication in a Web application server can be done in several ways. Filters/Plugins, such as NTLM, CA Siteminder are most commonly used authentication methods in a Web application server.
SAML	Authentication is done by an external Identity Provider based on the standard SAML 2 protocol. AppWorks Platform relies on the external identity provider for its

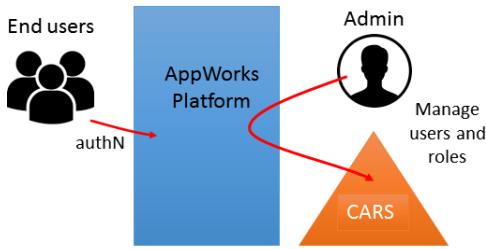
<b>Authentication type</b>	<b>Description</b>
	authentication.
OTDS	<p>Authentication is done by an OpenText Directory Services (OTDS) server. AppWorks Platform relies on an external OTDS server for its authentication.</p> <p>OTDS login, Active Directory are few authentication methods supported by OTDS.</p>

See [Authentication mechanisms](#) for more information on the authenticated mechanisms supported by AppWorks Platform.

Depending on the authentication type used, user management must be done in AppWorks Platform or another product.

### User management in AppWorks Platform

- When AppWorks Platform authentication is used, users sign in to AppWorks Platform through a native sign-in form. With this approach, users are available in OpenText CARS. Creating, updating, or deleting users and assigning roles to the users is done through the User Manager task in AppWorks Platform. Additionally, administrators and users can also change their passwords in AppWorks Platform. See [Managing users](#) for information on user management.
- When the Web application server authentication is used, users are available in OpenText CARS, but their passwords are not stored in OpenText CARS. The Web server is responsible for authenticating the user and handles it according to its mechanism, for example, through NTLM.
- When the SAML authentication is used, users are available in OpenText CARS but the user passwords are not stored in OpenText CARS. The SAML identity provider manages the user authentication. Therefore, user passwords are handled by that identity provider and not by AppWorks Platform.
- With the OTDS authentication, AppWorks Platform redirects users to the configured OTDS server to login. In this case also, users must still be available in OpenText CARS, but the user passwords are not stored in OpenText CARS as authentication is handled by OTDS.



User information might exist in different places. For example, user information resides in OpenText CARS, in SAML identity provider, or the OTDS server. To keep the user information consistent across all the products, OTDS synchronizes users from OTDS into other products such as OpenText CARS using OTDS Push connector. On synchronizing, whenever a user is created, deleted, or updated, the changes reflect in OpenText CARS database. However, it is not possible to synchronize such changes from OpenText CARS to OTDS. Therefore, an administrator must identify the appropriate user management method when using OTDS. For instance, when OTDS is used for user management, administrator must not use the User Manager task in AppWorks Platform. OTDS must be used as the primary component for managing users.

The following scenarios describe various approaches to manage the user details:

### **Using OTDS for user management**

OTDS manages users at a centralized location for multiple Open Text products, such as Content Server, Archive Center, or AppWorks Platform. A single OTDS server can be used to manage the user information and this information can be synchronized with different products using OTDS Push connectors. See [Configuring OTDS Push connector](#) on setting up and configuring the OTDS REST Push connector for AppWorks Platform.

The users in OTDS can be maintained in a non-synchronized partition in the OTDS server itself. In this case, the administrator creates, updates, or deletes users (including passwords) directly in the OTDS Admin user interface; users can also be made members of different groups in OTDS.

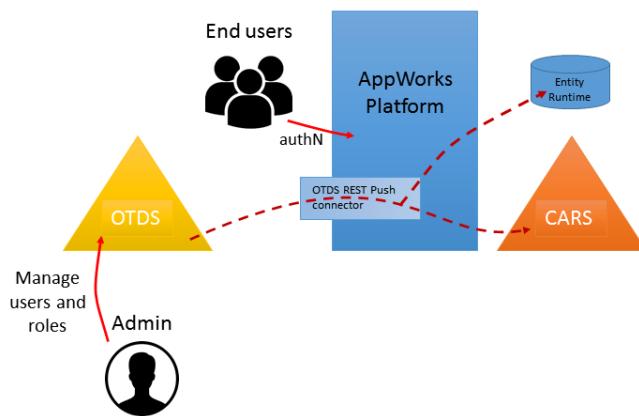
OTDS also supports synchronized partitions, where users, groups, and group memberships are synchronized from other user databases such as Active Directory (AD).

When a user is created in AD and synchronized with OTDS, the user details except the passwords are synchronized with OTDS; As the passwords are not synchronized, OTDS relies on AD for user authentication. Through the Push connector for OpenText CARS, the users are also synchronized into OpenText CARS. So, the users created in AD are synchronized with OTDS and subsequently synchronized from OTDS into OpenText CARS (possibly with other Open Text products through their Push connectors).

### Synchronizing users from OTDS to AppWorks Platform

The Push connector for AppWorks Platform synchronizes users, their details, and group membership from OTDS to AppWorks Platform.

Making a user a member of a group in OTDS implies that the user is assigned a role in AppWorks Platform. If the group is part of a specific OTDS package role partition, a package role is assigned to the user; otherwise, an organizational role is assigned.

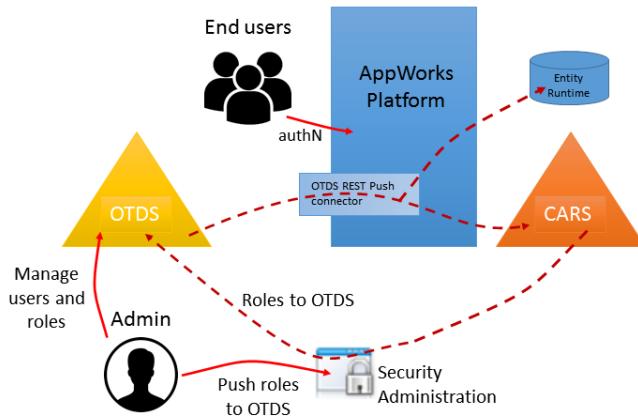


### Synchronizing roles from AppWorks Platform to OTDS

OTDS supports the concept of groups. A user can be a member of a group and a group can be a member of another group. This allows users in a group to become a part of another group. Using the Admin user interface in OTDS, you can create groups from AppWorks Platform or from an external identity provider, such as AD.

The Security Administration task in AppWorks Platform allows to synchronize roles from AppWorks Platform to OTDS. See [Synchronizing roles](#). All functional and application package roles are synchronized to a specific partition in OTDS and are made available as groups. Adding a user to one of these groups results in assigning a corresponding role to the user in AppWorks Platform. The Push roles to OTDS functionality must be used manually after the deployment of new packages that contain roles. Running the tool ensures that these roles are available in OTDS. The roles follow the pattern <package-name>#<role-name>. The

hash ('#') is used as a separator between the package name and role name, for example, Cordys@Work#Developer. Any role that contains a hash is considered to be a package role.



You must synchronize roles from AppWorks Platform to OTDS in the following scenarios:

- When a package with a functional or application role is deployed in AppWorks Platform.
- When an updated package is deployed and the package has new, renamed, or updated roles. A role is updated when either the role or role membership is changed.
- When AppWorks Platform is upgraded to a newer version or a fix pack, which might contain new or updated roles.
- When a package is undeployed from AppWorks Platform.

**Note:** The Push roles to OTDS functionality cannot detect a role name change. Therefore, when a role is renamed, the user and group membership for that role is lost. You must map the role to the user and group membership again.

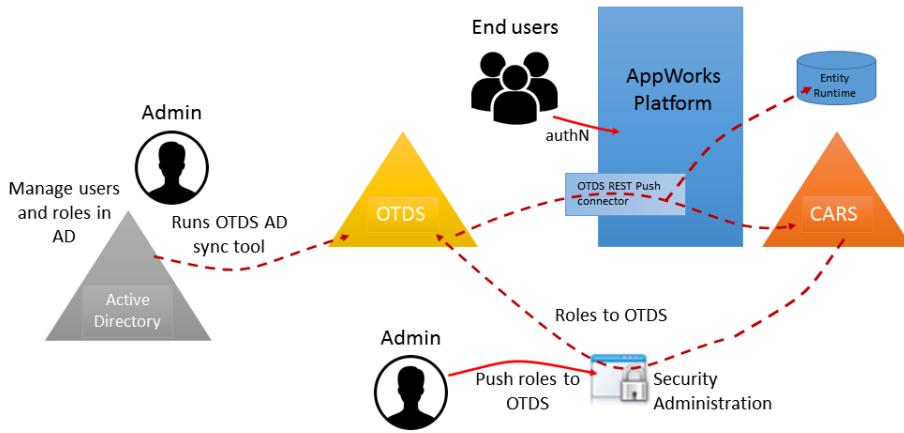
#### Restrictions when using OTDS for user management

Users and group memberships are synchronized ONLY from OTDS to OpenText CARS; they cannot be synchronized from OpenText CARS to OTDS. Therefore, you must not use the User Manager task in AppWorks Platform to modify the user-role assignments or user details such as changing name, description, or email address; these changes will be overwritten by the changes in OTDS.

#### Using Active Directory for user management

OTDS can be integrated with Active Directory (AD). You can synchronize users and groups in AD with OTDS. This setup allows administrators to manage users in AD and provide users with access to applications in AppWorks Platform. The users and group memberships in AD

are synchronized with OTDS and synchronized from OTDS into OpenText CARS subsequently. Therefore, when a user is created in AD, a corresponding user is created in OTDS and OpenText CARS. When a user is added as a member of a specific group, the user gets a role in AppWorks Platform.



This setup requires all functional and application package roles to be available in both OTDS and AD. However, some AD administrators might not allow such fine-grained configuration of their groups. In such scenarios, you cannot use this configuration.

**Note:** OTDS does not support synchronizing groups in OTDS with AD.

#### Restrictions when using AD for user management

While you can synchronize the users and group memberships in AD with OTDS, you CANNOT synchronize the user and group memberships in OTDS with AD. Therefore, if you make any changes to the user details in OTDS, they will be lost when you synchronize the data with AD.

**Note:** When using Active Directory for user management, do not edit users or group membership in OTDS.

#### When to use the User Manager task in AppWorks Platform

You may have to use the User Manager task in AppWorks Platform in certain scenarios such as:

- Configuring task assignments
- Using organization roles

- Using AppWorks Platform authentication. For instance, when you use AppWorks Platform authentication, even if you use OTDS for user management, user passwords are validated in AppWorks Platform only; hence, they must be available in AppWorks Platform. As user passwords are not synced from OTDS to OpenText CARS you have to use the User Manager task in AppWorks Platform to set or change the user passwords when using AppWorks Platform authentication.

## **Using OTDS with SSL**

AppWorks Platform and OTDS share sensitive information such as secret keys. To prevent others from intercepting such information, OpenText recommends that you enable Secure Sockets Layer (SSL), that is `https://`, on AppWorks Platform and OTDS.

Communication from AppWorks Platform to OTDS occurs while:

- Activating a resource - See [Managing an OTDS resource](#)
- Verifying the user identity for inbound OTDS SOAP requests - See [OTAuthentication SOAP header](#)

Communication from OTDS to AppWorks Platform occurs while:

- Consolidating users and groups - See [Configuring OTDS Push connector](#).

## **Securing communication from AppWorks Platform to OTDS**

If the OTDS Server is protected by SSL, AppWorks Platform must trust the SSL certificate.

If the certificate authority that is used to sign the SSL certificate, is not trusted by AppWorks Platform, or if the certificate is self-signed, the certificate must be added to the AppWorks Platform trust store. Use the Security Administration task to perform this action. See [Adding a new certificate](#).

## **Securing communication from OTDS to AppWorks Platform**

If the AppWorks Platform Web server or load balancer is protected by SSL, the OTDS Server must trust the SSL certificate. See [Configuring access URLs](#).

OTDS uses the standard Java trust store. Therefore, adding the certificate to that trust store configures the trust between OTDS and AppWorks Platform.

Follow the [instructions of the JVM vendor](#) to add a certificate to the trust store of the JVM.

For example:

```
keytool -import -trustcacerts -keystore "<<JAVA_HOME>>\jre\lib\security\cacerts" -  
storepass changeit -noprompt -alias mycert -file "<<full path to certificate file>>"
```

## Using OTDS tickets

AppWorks Platform supports the use of OTDS tickets for user authentication, that is, a client can send a SOAP request to AppWorks Platform with an OTDS ticket as an authentication token in the SOAP Header.

This topic explains the procedure to retrieve a ticket from OTDS and use it in the SOAP requests to AppWorks Platform.

### Configuring OTDS with AppWorks Platform

#### To configure OTDS with AppWorks Platform:

1. Access OTDS and create a resource in OTDS and link it through an access role to a Partition that contains users.
2. Access AppWorks Platform and configure the Platform resource with the resource ID of the resource in OTDS. See [OTDS resources and trust](#) for more details.
3. Optional: Configure the OTDS Push connector to synchronize users and roles to OpenText CARS and Identity components. See [Configuring OTDS Push connector](#).

**Note:** Ensure that the users intended for OTDS ticket authentication are available in both OTDS and AppWorks Platform.

### Obtaining an OTDS ticket

You can use the REST services in OTDS to obtain an OTDS ticket for a user. OTDS supports resource bound tickets and resource unbound tickets. The OTDS REST APIs can be accessed from `https://<otds-server>/otdsws/api/index.html`. The `/authentication/credentials` REST service can be used to obtain the OTDS ticket, which requires `userName`, `password`, and an optional `targetResourceId` as parameters. The `userName` and `password` are the credentials of the user in OTDS. The `targetResourceId` is the Platform Resource ID in AppWorks Platform. When the `targetResourceId` is provided, OTDS generates resource bound tickets. When `targetResourceID` is not provided, then OTDS generates a resource unbound ticket (OTDSSSO).

#### Resource bound ticket

Resource bound tickets are valid for a specific resource only and can be validated only once. The `/authentication/credentials` REST service requires `userName`, `password`, and `targetResourceId` as parameters.

#### To obtain a resource bound ticket:

1. Navigate to `https://<otds-server>/otdsws/api/index.html#/authentication/authenticateWithPassword`. A list of the available REST services is displayed.
2. Click **authentication/credentials** REST service.
3. Type the user name, password, and target resource ID as parameters in the body text box, where `userName` and `password` are the credentials of the user in OTDS and `targetResourceId` is the Platform Resource ID in AppWorks Platform. The

`targetResourceId` is mandatory for resource bound tickets.  
A sample request is displayed below.

```
{
  "userName": "jdoe@acme.com",
  "password": "Mypa$$w0rd",
  "targetResourceId": "9bf779ea-b34e-4b76-8f8a-4d439d85e0ae"
}
```

#### 4. Execute the request.

A response with the ticket information is displayed in the Response body text box as shown in the sample code below. The ticket value starts with `*VER2*`.

```
{
  "token": "6F7464735F73657373696F6E5F6B6579",
  "userId": "jdoe@acme.com",
  "ticket": "*VER2*ABTOd-_erjH-dR3KAKCFF80155w7AgAQX973eZYvkS5sEeH69C5U9QJANmsY_MX5CzZ4QgMZpphIZSProES2RYNRhtEEM17WW8jgkzSVuIro3dtMKZ2L22zbHUGJe1051Hu4CmcAhQTsv7nzYKIO-rqOmyxwq3U1dh5ff6fZGnOcgC76PBga3gKFM7dc1KBPXgBYzrrA-c4mlaq37m2sfhrWtVqE6EnEd_eZrkCp9dtuUgYj4fpGMT2Pb6gFp5Ap3NTQmAHH7tHYjaD6NCd5JJ_Y8F8eFwGvvNuJdmsfAWoQ6WC019ENzioTMoAa6aJGhx3oKvDVBww23N1_TB2Q3wt2F3KuFLsqyBG119142bDt5b_7k2TTpnB_OB81brYf6s-vphZzms1gUp8ntVnWRsOJYnVP8tLex-4Z3UrqPVDoN4EJce9co1EcZdjA2P6_nHJG3iC2FtLUwNikdetQmLRzytrhy3ft5b15l1DcZOT-dJI_gAxg4PWG1dNJSrdH7esD2ZJ03SQn1_Xk57jUqRUpFPTpkLLfY4umFZ_DzgDSQmlSwfXOcBQPvIpuML_V1SIL8qe4zLG0B1dZtgShZdMFL-sTXJYpO5z1TNx9yk-mNENICI333Rd6S31hX_NX4eBkk7omkvUUfy27wuwzk2t_-sIXhlc9Dzs-rnutfDuflvvyyTdaJRgAZhnjLwhHJu11I111G16-4WHAFCVjmpJtxL-8KzYV07pT95jxLg_ccsq5PfZ_qnAnDiFY4GfH4BdBrfNGNO_tQNMwycUDVTOZGtCo4voCEYypbmA9xcxyX26m_Ikfcm",
  "resourceID": "9bf779ea-b34e-4b76-8f8a-4d439d85e0ae",
  "failureReason": null,
  "passwordExpirationTime": 0,
  "continuation": false,
  "continuationContext": null,
  "continuationData": null
}
```

## Resource unbound ticket

Resource unbound tickets can be used with any resource and can be validated multiple times until they expire.

1. Navigate to `https://<otds-server>/otdsws/api/index.html#/authentication/authenticateWithPassword`. A list of the available REST services is displayed.
2. Click **authentication/credentials** REST service.
3. Type the user name, password as parameters in the body text box, where `userName` and `password` are the credentials of the user in OTDS. The `targetResourceId` is not required for resource unbound tickets.  
A sample request is displayed below.

```
{
  "userName": "jdoe@acme.com",
  "password": "Mypa$$w0rd"
}
```

#### 4. Execute the request.

A response with the ticket information is displayed in the Response body text box as shown in the sample below. The ticket value starts with \*OTDSSO\*.

```
{
  "token": "6F7464735F73657373696F6E5F6B6579",
  "userId": "jdoe@acme.com",
  "ticket": "*OTDSSO*AWRBQ1FpS2xMdmlqc1ZaOV16Y21DNDI3WmZQREc2NXdBUTdHM2V0UENzNzkwFRHdld0N05
OVVFEZ0FjQmJNcjh5MVZBbXdxVURFN01RQXJuYzEwb2NGNjdXN2pQQT11TGtydWpKb1Z0LXctrDRZRVRy
UGt1ZnhxNmU0QuC1YTdZQ18xTEtISE1EamVXOXc0bTJvZ1RNcUVON1NpWT1RUXFZX3czdnoyZUVsN1Yyd
GRTUW9VLW8yZzFqTWVOYWyU3FWbj12bWR6eXR3a01IY0x4RVRneTB0SzNoTm84T11ES0pnRDQ1SXdYVj
dWVURNZ3p5VkJYs1JEaUVpXzMzN05LYVBuczhORVp5SjZ1bG1XQ1M5Qm9CbFprd1JfV1IycEVTR2NTRjR
HWG5aWEdwVW5uZXFkZV93eHdvUEJPQXhsY253aWFyZUFTZ2w1SU5LbF1WR0hHdDVYc19SWk1XNVBtaXZR
KgBOAEoAFGeDI3Pj3AfJ4wuMQxIFvyN_
Wu0dABAZghus779RwFbcp8oeuy8YACBKOSb0Ja0HNhmlyzA6X0a3hj6FJUMMO7pGJPWckFS0UwAA",
  "resourceID": null,
  "failureReason": null,
  "passwordExpirationTime": 0,
  "continuation": false,
  "continuationContext": null,
  "continuationData": null
}
```

### Using the OTDS ticket

This OTDS ticket can be used in a SOAP request to AppWorks Platform. The ticket information must be provided inside the <SOAP Header>/<OTAuthentication> tags enclosed in the AuthenticationToken element. See [OTAAuthentication SOAP header](#).

The urn:api.bpm.opentext.com namespace must be as OTAuthentication as shown in the sample below.

```
<SOAP:Envelope xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP:Header>
    <OTAuthentication xmlns="urn:api.bpm.opentext.com">

      <AuthenticationToken>*OTDSSO*AWRBQ1NETmtrUk9IcDJKaWlvRldmaG5JMU1EeDc5N1FBUXVGyVdoVz
      BQOxP2Yz1jVERUeU1RLUFEZ3FCclhTdnRCWnp4dUhoNWNvX0ZhdVVQeUo0bVFTYVNPan1PX0hNVDN2bmRMal
      RaTm9ySWJDaF9ITTk4OU9rLvdDOFBaV1JKS1o4QXd2SUJfafz13Z3p5azRKNGVybVlpMW1WeFBpSGFXT19VT0
      1nbDRoWULoWULoYW1vTEZSTHc2LU5iNGlheW83MXc0UHoTGVPUg8xWVUwaDI5VEs3ejh2TzRpWG9wd1JkNF
    </AuthenticationToken>
  </OTAuthentication>
</SOAP:Header>
</SOAP:Envelope>
```

```

Ziv3dud3FmM3NnNmpidE42VmwytcTk1UHA1VWRMd0pnd0FBdzRRZ1FCdlZhak1YZTN0dk5yQVU2cHpaeeUJJaj
g2d1c4Y0p5cGdWeTFRM1JHNTBVMDJKSTJGZmltT200ZXFaS2JSV0JRZ3FEeVNvYmdIOU0wZDlhUm1DSVNFV0
UwKgBOAEoAFPQt6DkuJjczzHT8jDi4GvoM3pqUABB_4vAEZGEBoNdi_X2SE4BOACBj2HdBdsxG9_
98BvMaV14Tx7rw8m9gLbdF7RHivFNKwAA</AuthenticationToken>
    </OTAuthentication>
</SOAP:Header>
<SOAP:Body>
    <samlp:Request xmlns:samlp="urn:oasis:names:tc:SAML:1.0:protocol">
        MajorVersion="1" MinorVersion="1"
        IssueInstant="2018-11-07T16:47:13.359Z" RequestID="a5470c392e-264e-jopl-56ac-
4397b1b416d">
            <samlp:AuthenticationQuery>
                <saml:Subject xmlns:saml="urn:oasis:names:tc:SAML:1.0:assertion">
                    <saml:NameIdentifier Format="urn:oasis:names:tc:SAML:1.1:nameid-
format:unspecified"/>
                </saml:Subject>
            </samlp:AuthenticationQuery>
        </samlp:Request>
    </SOAP:Body>
</SOAP:Envelope>

```

The request returns a SAML artifact that can be used in subsequent requests to AppWorks Platform. For more details on how to use a SAML artifact, see [Using SAML artifacts](#). While the OTDS tickets can be used to get a SAML artifact, they can also be used with any other SOAP request. A typical use case of an OTDSSO ticket is to retrieve it from OTDS and use it directly in subsequent Web service requests to AppWorks Platform.

### Common mistakes

Few common mistakes that occur when using the OTDS tickets and their solutions are as follows:

- the user for which the OTDS ticket is retrieved does not exist in AppWorks Platform or the user ID is different than in OTDS. For example, `jdoe` vs `jdoe@users`. Open the User Manager task in AppWorks Platform and verify that the user ID of the user matches the user ID provided by OTDS.
- a resource bound OTDS ticket is used more than once in a SOAP request sent to AppWorks Platform. As a resource bound OTDS ticket can be validated only once, it can be used only once in a request to AppWorks Platform.
- the SOAP request is executed as the `anonymous` user. This can occur when a wrong namespace is used. For older releases until AppWorks Platform 16.4, the namespace for the `OTAuthentication` element must be `urn:api.ecm.opentext.com`. From AppWorks Platform 16.5 onward, the namespace must be `urn:api.bpm.opentext.com`.

### OTDS resources and trust

OpenText Directory Services (OTDS) is a product for identity management and user authentication across OpenText products. OTDS provides the single sign-on feature for the user across OpenText products. Users can be managed in OTDS, and information about the

identity of a user can be exchanged between products that support OTDS. In OTDS, a user can have a different account for each product. OTDS has a functionality that can map user accounts between different products. With this, OTDS can provide authentication proof for a user account in one product based on the authentication proof of the same user in another product.

All products must be registered in the same OTDS server. This registration is called a resource and is identified by a resource ID. To create a trust relation between a product and the OTDS server, the resource ID must be activated. In the activation process, OTDS provides a shared secret. This shared secret is used to secure the communication between the product and the OTDS server. It is an expression of trust.

The trust relation between the OTDS server and the AppWorks Platform is used for user authentication, inbound request authentication, and single sign-on integration with other products. See [Configuring OTDS authenticator for AppWorks Platform](#) for user authentication. See [OTAuthentication SOAP header](#) for inbound request authentication.

To integrate with other products, AppWorks Platform must know the resource ID of that product. For example, to send a request to Content Server, AppWorks Platform requests the OTDS server for an authentication proof for the Content Server resource ID on behalf of the current user.

This is not required for Web-based single sign-on, since the interaction is done in the browser with redirects to the OTDS server.

## **Creating a trust relation between the OTDS server and AppWorks Platform**

### **To create a trust relation:**

1. Create a new Resource in the OTDS server and copy the new resource identifier.
2. Create the trust relation in AppWorks Platform using the OTDS server URL and the resource identifier. See [Managing OTDS platform resources](#).

## **Registering other products for use with the AppWorks Platform**

### **To register the resource of another product:**

1. Search the product Resource in the OTDS server and copy the new resource identifier.
2. Create a new Resource in the AppWorks Platform using the OTDS server URL and the resource identifier. See [Managing an OTDS resource](#).

## **Managing OTDS platform resources**

To enable authentication, inbound trust and single sign on with multiple products, an OTDS resource must be configured and activated, this is called the platform resource. This expresses the trust relation between AppWorks Platform and the OTDS server.

### **Before you begin**

To configure a platform resource to either level, you must have one of the following roles:

Space	Role
Shared	Security System Administrator
Organization	Security Administrator

**Note:** The platform resource in the shared space is applicable for all the organizations and the one in the organization space is applicable only for that organization. Hence, a specific role is required to configure the platform resource in the shared space.

### Opening the OTDS platform resources configuration panel

#### To open the OTDS platform resources panel:

1. On the Welcome page, click  (Security Administration).  
The Security Administration window opens and the **Certificates** tab is displayed by default.
2. Click the **OTDS Resources** tab.
3. Click the **Platform** tab.

#### Creating the trust relation

#### To create the trust relation:

1. Configure the platform resource.
  - a. Provide the OTDS server URL to access the OTDS server. This must be a valid URL, starting with either `http` or `https`. There can be a port number in the URL.  
To configure the trusted certificates between AppWorks Platform and OTDS, see [Using OTDS with SSL](#).

**Note:** Example: `https://otds-server:8443/`.

Do not add `/otds-admin/` in the URL. In case an OTDS tenant is used then provide the URL including `/otdstenant/<otds-tenant-name>`, for example: `https://otds-server:8443/otdstenant/acme/`.

1. In **Resource ID**, specify the unique ID of the AppWorks Platform resource configured in OTDS.
  - c. Click  to store the resource in the repository.
2. Activate the platform resource.
  - a. Click **Activate**.

**Note:** It is only possible to activate resources that are stored in the repository. If **Activate** is grayed-out, click first.

The platform resource is configured and a trust relation is created.

**Note:** The platform resource for the shared space can only be configured by the system administrator and therefore only in the system organization. This will be the default platform resource for all the organizations. If an organization administrator wants to

configure his own platform resource, he can overload the shared platform resource by configuring a platform resource for the organization configuration.

### Removing the trust relation

#### To remove the Platform Resource:

1. Click **Clear Resource**.
2. Click  to remove the resource from the repository.

### Auditing modifications

#### To enable auditing modifications to the Platform Resources:

1. On the Welcome page, click  (Audit Configuration).
2. Enable the artifact type **OTDS Resource**.

The OTDS resources are enabled for modifications.

### Managing an OTDS resource

An OTDS resource can be configured either at the shared space level or the organization space level. Based on the required level, you must have the following roles to add, delete, or modify the resources:

- Shared: Security System Administrator
- Organization: Security Administrator
- You can view and manage OTDS resources from the Security Administration task. OTDS resources are used in different components such as the Document Store Service Container and Metastorm BPM (MBPM) configurations.

This topic includes:

- [Adding a resource](#)
- [Modifying a resource](#)
- [Deleting a resource](#)
- [Activating a resource](#)
- [Enabling audit for a resource](#)

### Adding a resource

#### To add a resource:

1. On the **Welcome** page > **My Applications**, click  (Security Administration).  
The **Security Administration** window opens.
2. Click the **OTDS Resources** tab.  
The OTDS resources configuration panel displays.

3. On the **Other** tab, click .

An empty editable **Resource Details** pane displays at the bottom.

4. Do the following:

Field	Description
Resource Name	Type any name to identify the OTDS resource.
Space	Select the space for which the OTDS resource is active. <b>Note:</b> The OTDS resources in the shared space can be used across all the organizations. However, an OTDS resource in the organization space can be used in the current organization only.
OTDS Server URL	Type the URL of the OTDS server to be accessed. This must be the same as the effective OTDS platform resource URL. See <a href="#">Managing OTDS platform resources</a> for more information.
Resource ID	Type the unique ID for the OTDS resource configured in OTDS.

5. Click  to store the resource in the repository.

## Modifying a resource

### To modify a resource:

1. On the **OTDS Resources** tab, click the required resource to be modified. An editable **Resource Details** pane displays at the bottom.
2. Update the values of the **OTDS Server URL** and **Resource ID** fields as applicable.  
**Note:** The **Resource Name** and **Space** fields cannot be modified as they form the unique key.
3. Click  to store the resource in the repository.

## Deleting a resource

### To delete a resource:

1. On the **OTDS Resources** tab, click the check box of the resource to be deleted.
2. Click .  
The selected resource is removed from the table.
3. Click  to remove the resource from the repository.

**Note:** You can perform the add, update, and delete actions simultaneously before storing it in the repository.

## Activating a resource

You can activate only those resources that are stored in the repository.

### To activate a resource:

1. On the **OTDS Resources** tab, click the resource to be activated.  
An editable **Resource Details** pane displays at the bottom.
2. Click **Activate**.  
The selected resource is activated.

**Note:** If **Activate** is disabled, save the resource to enable it. Click  (Save), and then click **Activate**.

### Enabling audit for a resource

#### To enable auditing for OTDS resource:

1. On the **Welcome** page > **My Applications**, click  (Audit Configuration).  
The **Audit Configuration** page opens.
2. Select the check box for the OTDS resource.
3. Click .

Auditing is enabled for the OTDS resource.

### OTAuthentication SOAP header

When an external application sends a SOAP request to AppWorks Platform, authentication information must also be included with the SOAP request. OTDS ticket is one of the authentication information types that AppWorks Platform supports. After the administrator configures OTDS Resources and Trust, AppWorks Platform accepts inbound SOAP requests with OTDS tickets as the proof of authentication. See [OTDS resources and trust](#).

The OTDS ticket information must be available in an `AuthenticationToken` node, which is wrapped in the `OTAuthentication` node. The `OTAuthentication` node is a child of the SOAP header. The `OTAuthentication` node can be sent along with any SOAP request and is not limited to the Authentication Request.

When the AppWorks Platform receives a SOAP request with an OTDS ticket in the `OTAuthentication` SOAP header, it validates that ticket against the trusted OTDS server. After the validation of the OTDS ticket, the OTDS server returns the user ID of the corresponding user. AppWorks Platform then searches for an authenticated user and an organizational user based on the user ID and trusts that user ID when performing the actual request.

OTDS supports two types of OTDS tickets:

- Resource bound tickets - the resource bound OTDS tickets can only be used with a specific OTDS resource. An OTDS Resource ID is provided in the request sent to OTDS, ensuring only the users configured with this specific resource in OTDS are used. For example, any user who has access to an application through Resource A cannot access a system or application through Resource B. The resource bound ticket can only be validated once. Therefore, when using the resource bound ticket type, it is recommended to use the `OTAuthentication` header authentication to request an

AppWorks Platform specific SAML artifact for use in subsequent calls.

- Resource unbound tickets - also known as OTDSSSO tickets, the resource unbound ticket is retrieved from OTDS without specifying a targetResource ID. It is not bound to a specific OTDS resource but can be used with any OTDS resource. It can be validated repeatedly until its validity expires.

### Example SOAP request

The following is an example of a SOAP request to retrieve SAML assertions from AppWorks Platform where the authentication is done based on the OTDS ticket provided in the SOAP header.

### Sample SOAP request with OTDS authentication in the header

```
<SOAP:Envelope xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/">
<SOAP:Header>
<OTAAuthentication xmlns="urn:api.bpm.acme.com">
    <AuthenticationToken>*VER2*ABQz0QagVDQY_3lxtdzT7VwUFEEEowaQ-F0AfX6hCKdIX
fYekJvDbwHgsfB0dDWUOcgcyCUo3I3whFbQIy99AEUAtmgaim91m-
OtLXyg65S4dieHxqI2EvZjPcuE9v0ucS1kDu09MigB
m5Lq0Pwx8kqB9cQ2-nOyqY1T9rpfn7bBOA2v1ZZb1ZE0OvmhCZNOSKI_
z642BdDnOGcLJLpcGEYoTd9qYdfrH6CQI5TOE-1
_SLe4xZWTQ1R6BOC_isFGmK93QuHLWxHwNbNGy5yPqN9ZIRFzPI76jF5MbEeY8I0rmGEDsZseVuRd_
h6fEXjf9ogUPvOX4f
K31U_mGDU_FK1VdeD9ryMC6AK-jbSdRONunFXvrZFa-
mP4ZVqpRyKGi1HXebuYvAn1ESDddJIsid4usztr8WX3igUMTDk5B
nj-aQs9r10puydB44iNub9pqmWtYUbT8igieIp3_NfXFpJga5C5AX27iJWhi_4Z3mHVz437Rj2A-
Ov31ITgt74IiFzmmwX
hJvk_h3ZBmcHgMyImKfGUOPZkd-tU-kpSCA3qQoPMh9Lmn1E4dAxbbp098SBthN1Z1VeZvHo_
wEVoAc38TptuASZP32XQD2
FW4HYS64t9SW-TzBS3w27EJhUSDxZYDxZ51y1R2p11IXu9QuRwlSEzthkZj6JW1DUkt7fwUql6Yy4TmW_
    </AuthenticationToken>
</OTAAuthentication>
</SOAP:Header>
<SOAP:Body>
    <samlp:Request xmlns:samlp="urn:oasis:names:tc:SAML:1.0:protocol"
MajorVersion="1" MinorVersion="1" IssueInstant="2014-05-20T15:29:49.156Z"
RequestID="a5470c392e-264e-9537-56ac-4397b1b416d">
        <samlp:AuthenticationQuery>
            <saml:Subject xmlns:saml="urn:oasis:names:tc:SAML:1.0:assertion">
                <saml:NameIdentifier Format="urn:oasis:names:tc:SAML:1.1:nameid-
format:unspecified">
                    </saml:NameIdentifier>
                </saml:Subject>
            </samlp:AuthenticationQuery>
        </samlp:Request>
    </SOAP:Body>
</SOAP:Envelope>
```

## Example SOAP response

```

<SOAP:Envelope xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP:Body>
    <samlp:Response xmlns:samlp="urn:oasis:names:tc:SAML:1.0:protocol"
      ResponseID="A0050568B-0048-11E3-FC06-71F9A6615F04" MajorVersion="1" MinorVersion="1"
      IssueInstant="2014-05-20T15:29:49.199Z" InResponseTo="a5470c392e-264e-9537-56ac-
      4397b1b416d">
      <Signature xmlns:samlp="urn:oasis:names:tc:SAML:1.0:protocol"
        xmlns="http://www.w3.org/2000/09/xmldsig#">
        <SignedInfo>
          <CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
          <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
          <Reference URI="#A0050568B-0048-11E3-FC06-71F9A6617F04">
            <Transforms>
              <Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
            </Transforms>
            <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
            <DigestValue>OwNCVXqfWIqTl8mWYI70JRGtHww=</DigestValue>
          </Reference>
        </SignedInfo>
        <SignatureValue>
          CPM3IK/UtNCZhFpIGx/REBiJ9wJ0NR5T2s0faNQW5sJhiKSTRbkplaqiQzt7m6brJetic3dPVzg
          Bwi4t3j19gE/TW1CgQODJwNKZFjatx5t+mG3PmOczA5Kwb/dp2fZSyC/Hb90IuIN5SKkk8hlcbB
          ymZOdVOjfOA6+pugpob00=
        </SignatureValue>
        <KeyInfo>
          <X509Data>
            <X509Certificate>
              MIICXTCCAUWgAwIBAgIPUFaLAEgR4/v31DmtE/8EMA0GCSqGSIb3DQEBBQUAMDAXDzANBgNVBA
              oTBnN5c3RlbTEdMBsGA1UEAwuBw9uaXRvckBzcnYtbmwtY3JkMzgwHhcNMTQwNTIwMDEzMjQ3
              WhcNMjQwNTE3MDEzMjQ3WjAqMRcwFQYDVQQDEw5zaW5nbGUgc2lnbi1vbjEPMA0GA1UEChMGc3
              1zdGVtMIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQC8hMDOMYbgkRd8FFIIWpnVaPF37WrH
              PNbaeo589AnZ1F/Hz0G3N7PK1Tu2C2horcxMI913S0SwzbMY0U6GBeGmQnpQO5vtV13tfvTKf
              TkANHXfvnj76okCQm200LAeUgAHNPDrHlk2DYqhd1HgJW8XAz6B6uoLWvOeelZhKYmGwIDAQAB
              MA0GCSqGSIb3DQEBBQUAA4IBAQBLpYQt98NCGUKOUUC3+yrRD1orBrRh2romOkZGMwQHwE/n5+
              ZczlHCiKQoQ8vkL3tp/Y/53sLXHQTp/Qki84DgGUQfe8JdBPOK/jXlgrLxX1h5/1LjlZRfP+z
              Kd0u44X9vtTfqXzt/BJS6pofBVL9k5Jxee0lm0qfyjWG9bR8zb61jdWgsP36cV2cCPHbPMoOSM
              117CW9g9v0AoVMyGK3wgLXcaeOLRITbMDKpicyWK0qc94LPujs/qBd9i1eI2GuKD0FmoGm4c7v
              S/SBidK+UnqXJCB+QcMBqEVKVBHFEcoJNzcTZJ+tag8RFDjfGP2oSTBz2eati2RkL5N/AjNJ
            </X509Certificate>
          </X509Data>
        </KeyInfo>
      </Signature>
      <samlp:Status xmlns:samlp="urn:oasis:names:tc:SAML:1.0:protocol">
        <samlp:StatusCode Value="samlp:Success" />
      </samlp:Status>
      <saml:Assertion xmlns:samlp="urn:oasis:names:tc:SAML:1.0:protocol"
        xmlns:saml="urn:oasis:names:tc:SAML:1.0:assertion"
        AssertionID="A0050568B-0048-11E3-FC06-71F9A6617F04"
        MajorVersion="1" MinorVersion="1" IssueInstant="2014-05-20T15:29:49.200Z"
        Issuer="https://www.acme.com/SSO">
    
```

```

<saml:Conditions xmlns:saml="urn:oasis:names:tc:SAML:1.0:assertion"
NotBefore="2014-05-20T15:24:49.200Z" NotOnOrAfter="2014-05-20T23:29:49.200Z"/>
  <saml:AuthenticationStatement xmlns:saml="urn:oasis:names:
  tc:SAML:1.0:assertion"
  AuthenticationMethod="urn:oasis:names:tc:SAML:1.0:am:password"
  AuthenticationInstant="2014-05-20T15:29:49.200Z">
    <saml:Subject xmlns:saml="urn:oasis:names:tc:SAML:1.0:assertion">
      <saml:NameIdentifier Format="urn:oasis:names:tc:SAML:1.1:nameid-
format:unspecified">jopl
      </saml:NameIdentifier>
    </saml:Subject>
    </saml:AuthenticationStatement>
  </saml:Assertion>
  <samlp:AssertionArtifact xmlns:samlp="urn:oasis:names:tc:SAML:1.0:protocol">
MDGeMioCNVSvx176ZD5hEmc56SbrNva5JH7ESkQZ6lZE6Bt1DCjDqkIt
</samlp:AssertionArtifact>
  </samlp:Response>
</SOAP:Body>
</SOAP:Envelope>

```

## Using SAMLart as the HTTP header

In subsequent requests, the value in the `AssertionArtifact` tag can be provided as the authentication proof. This value can be sent through the HTTP header, `SAMLart`.

Sample SOAP request:

```

POST https://testbop.acme.com/home/acmeNL/com.eibus.web.soap.Gateway.wcp HTTP/1.1
Host: testbop.acme.com
Connection: keep-alive
Content-Length: 181
Content-Type: text/xml; charset=UTF-8
SAMLart: MDGeMioCNVSvx176ZD5hEmc56SbrNva5JH7ESkQZ6lZE6Bt1DCjDqkIt
<SOAP:Envelope xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP:Body>
    <GetUserDetails xmlns="http://schemas.acme.com/1.0/ldap"/>
  </SOAP:Body>
</SOAP:Envelope>

```

## XSD schema for OTAuthentication SOAP header

The XSD schema for the `OTAuthentication` SOAP header is as follows:

```

<xsd:schema attributeFormDefault="unqualified" elementFormDefault="qualified"
targetNamespace="urn:api.bpm.acme.com"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <xsd:element name="OTAuthentication">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="AuthenticationToken" type="xsd:string"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
</xsd:schema>

```

```

</xsd:complexType>
</xsd:element>
</xsd:schema>

```

## OTDS OAuth integration

A web-service may require specific information from the client for an authenticated user-context. This can be done using different, standard, or proprietary methods. A standard method is OAuth. OAuth describes a specific set of tokens to be used. One of its advantages is that the client application does not need to know or store the user credentials, making it more secure.

### OAuth support in AppWorks Platform

When an external application sends a SOAP or REST web-service request to AppWorks Platform, authentication information must be included with that request. AppWorks Platform supports different types of authentication information, such as OTDS OAuth access tokens. After the Administrator configures [OTDS resources and trust](#), AppWorks Platform accepts inbound requests with OTDS OAuth access tokens as the proof of authentication.

The OTDS access token must be available for AppWorks Platform in the **Authorization HTTP header** field of the request.

When AppWorks Platform receives a request with an OTDS OAuth access token, it validates that token with the help of the configured trusted OTDS server. The access token contains a standard JSON Web Token (JWT) which is validated by the platform. This JWT contains the user ID of the user the token is issued for. AppWorks Platform uses this ID to find an authenticated and organizational user, and continues running the actual request.

### Using the OAuth access token

To use an OAuth access token in a web-service call to the AppWorks Platform:

1. In OTDS, create an OAuth client. To configure an OAuth Client, see OAuth Clients in the OTDS documentation.
2. Create an OAuth access token for a user in the AppWorks Platform partition. For details, see the OAuth-based Integration document.

**Note:** An OAuth client can get access and refresh tokens using any of the OAuth2 grants specified in RFC 6749.

3. Use the OAuth access token to authenticate a web-service request to AppWorks Platform by adding the **Authorization** HTTP header with the value **Bearer + access token**.

### Example:

```
POST http://srv-nl-jopl:8080/home/system/com.eibus.web.soap.Gateway.wcp HTTP/1.1
```

```

Host: srv-nl-jopl:8080
Content-Length: 181
Content-Type: text/xml; charset=UTF-8
Authorization: Bearer ABSRBksSXb01ifFJUiGW8wAQ...oOctYLwQncrMeoq1NXfxo-L2VpqMp
<SOAP:Envelope xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/">
    <SOAP:Body>
        <GetUserDetails xmlns="http://schemas.cordys.com/1.0/ldap"/>
    </SOAP:Body>
</SOAP:Envelope>

```

When an OAuth access token expires, a 401 HTTP status code is returned and a new access token must be retrieved from OTDS.

## **Configuring OTDS authenticator for AppWorks Platform**

### **Before you begin**

Create a trust relation between AppWorks Platform and the OTDS server. See [OTDS resources and trust](#).

AppWorks Platform can integrate with OpenText Directory Services (OTDS) for user authentication. This integration provides Single Sign-on behavior to the users, that is, users need to sign in to OTDS only once, instead of signing in for each product separately.

AppWorks Platform must validate the username and password of the logging in user. With an authenticator, that task is delegated to an external system such as OTDS. AppWorks Platform supports two types of OTDS authenticators:

- OTDS - for authenticating applications using OTDS. By using an OTDS-typed authenticator, Single Sign-on (SSO) can be achieved between different OpenText product instances. When OTDS is configured on the Active Directory, the SSO experience might be present across other applications as well.
- OAuth - for applications that require users to sign rule actions by forcing the users to confirm their identity again. To use an OAuth authenticator, you must have configured an OAuth client in OTDS. The OAuth client must have a redirect URL that allows OTDS to redirect to AppWorks Platform. As the redirect URL, provide the public server URL of AppWorks Platform, for example <https://myapp.acme.com>. For details, see OAuth Clients documentation in OTDS.

To set up AppWorks Platform to rely on OTDS for its user interface-based authentication, the administrator must create an authenticator.

### **To create an authenticator:**

1. On the Welcome page, go to **Security Administration > Authenticators**, and do one of the following:

- To create an authenticator for all the organizations, click  on the Shared Authenticators grid.
- To create an authenticator for all the users in a specific organization, click  on the Organizational Authenticators grid.

The Authenticator Properties window opens.

## 2. For Authenticator:

Field	Action
ID	Enter a unique identifier for each authenticator
Type	Select <b>OTDS</b> or <b>OAuth</b> from the list. Based on the selection, the fields on the Properties tab vary.
Description	Enter a meaningful description for the authenticator.
Default	Select <b>Default</b> to make this authenticator as default. When accessing AppWorks Platform using the browser, it uses a URL pointing to the instance. A URL parameter called <code>authID</code> is used to ensure the use of a specific authenticator. If this authID URL parameter is not specified, the default authenticator is used.
Test only	Select <b>Test only</b> to test this authenticator before setting it to default.
Test URL	Enter the URL to access the AppWorks Platform instance with this active authenticator configuration.
Use as default for signing	This option is available only when OAuth is selected as the authenticator type. When users need to confirm their identity to sign a rule action, this authenticator is used as the default authenticator.

## 3. For Frame Properties:

Property	Action
No Frame	Select <b>No Frame</b> if the Authentication form does not require a frame. This option reloads the page and results in losing the browser context. With this option, you can view the complete URL of the external identity provider, which ensures more trust. You can validate going to a trusted site before providing your username and password. This is the default and most secure setting.
Maximize	Select <b>Maximize</b> to display the Authentication form in a maximized frame.

Property	Action
Width	Enter the width of the frame that displays the Authentication form.
Height	Enter the height of the frame that displays the Authentication form.
Target Frame	This setting is not needed in combination with your application.

4. Click the Properties tab and do the following:

Property	Description
OTDS Server URL	Displays the OTDS Server URL as configured for the Platform resource.
Public OTDS login URL	<p>Optional.</p> <p>Enter the Public OTDS login URL only if this is different from the OTDS server URL. The OTDS server URL is the URL to access the OTDS server from the internal network.</p> <p>If not on the internal network, the user must access the OTDS server and then specify the publicly accessible URL of OTDS. This Public OTDS server URL is used from the browser.</p> <p>Access the OTDS server, and then enter the publicly accessible URL of OTDS if you are not on external network. This Public OTDS server URL is used from the browser. The value specified in the Public OTDS login URL is used as specified. A typical OTDS login URL ends with /otdsws/login.</p> <p><b>Note:</b> Ensure that you specify a complete login URL.</p> <p><b>For example:</b> <a href="https://otds.acme.com/otdsws/login">https://otds.acme.com/otdsws/login</a></p>
Resource ID	Displays the ID of the resource as configured for the Platform resource.
Client ID (displayed only when OAuth is selected as the authenticator type.)	Enter the ID of the OAuth client that is configured in OTDS.

5. Click **Save**.

The OTDS Authenticator is configured and added to either the Shared Authenticators list or the Organizational Authenticators list.

## Using the built-in authenticator

If accessing the default OTDS authenticator results in an unrecoverable error, the administrator can sign in and correct the configuration using the AppWorks Platform's built-in authenticator.

To access the built-in authenticator, use the `authID` URL parameter with the value **CordysBuiltIn**. This bypasses any default configured authenticator and displays the built-in AppWorks Platform login page.

Example:

`https://www.acme.com/home/myorg/?authID=CordysBuiltIn`

## Configuring OTDS for signed user action

The signed user action uses OTDS to reauthenticate the current user. To do this, a URL to sign-in must be opened on the configured OTDS server where the user can approve or cancel the signing action. OTDS displays a specific OTDS sign user interface or will redirect users to a configured external Identity Provider. For details on using the signed user action feature, see *A user triggers an action button > This action must be signed by the user* section in the AppWorks Platform Low-Code Design guide.

AppWorks Platform uses a specific OAuth flow when communicating with OTDS. To enable this, an OAuth client must be created in OTDS and an OAuth authenticator must be configured in the Security Administration task in AppWorks Platform.

## Configuring OAuth client in OTDS

### To add an OAuth client in OTDS:

1. Connect to OTDS web administration page using the following URL:  
`https://<hostname>:<port no>/otds-admin`, and sign in with admin credentials.  
The Web administration menu displays.
2. Click the **OAuth Clients** tab.
3. On the OAuth Clients page, click **Add**.
4. On the General page, enter a **Client ID**.  
Make a note of this ID as you have to provide it while configuring the OTDS authenticator for AppWorks Platform.
5. Click **Next** until you reach the Redirect URLs page.
6. On the Redirect URLs page, click **Add** and enter the AppWorks Platform URL as the redirect URL in the following format - `<protocol>://<server domain>:<port number, if any>`. For example, `https://awp.acme.com/`

**Note:** Do not add `/home` or `/home/<organization>`.

The OAuth client is added in OTDS.

## Configuring OAuth authenticator in AppWorks Platform

### To create an OAuth authenticator:

1. Access AppWorks Platform.
2. On the Welcome page, go to **Security Administration > OTDS Resources**.
3. On the Platform tab, enter the following details as provided when configuring the OTDS resource.
  - a. OTDS Server URL
  - b. Resource ID
4. Save the resource and click **Activate**.
5. Click the **Authenticators** tab.
6. Click  to add an authenticator.
7. For Id, enter a unique identifier for the authenticator.
8. From the Type list, select **OAuth** as the authenticator type.
9. Select **Use as default for signing**.
10. On the Properties tab, for Client ID, enter the ID of the OAuth client that is configured in the section above.
11. Click .

OAuth is configured in OTDS and AppWorks Platform for the signed user action.

## Configuring authentication handler for a repository

### Before you begin

AppWorks Platform must be configured as an OAuth Client in the OTDS of the corresponding repositories such as OpenText Core and OpenText Documentum. See [Configuring OTDS authenticator for AppWorks Platform](#)

## Configuring the authentication handler for the repository

### To configure authentication handler for repository:

1. Connect to OTDS web administration page using the following URL:  
`https://<hostname>:<port no>/otds-admin`, and sign in with admin credentials.  
The Web administration menu displays.
2. Click **Authentication Handlers**.  
The Authentication Handlers dialog box opens displaying a list of all defined authentication handlers.
3. Click **Add**.  
The New Authentication Handler assistant guides you through the steps.

4. On the General page, do the following:
  - a. In **Authentication Handler name**, type a name for the authentication handler.
  - b. In **Authentication Handler type**, select **OAuth 2.0/OpenID Connect Authentication Handler**.
  - c. In **Description**, the description of the authentication handler is displayed by the Directory Services based on your selection of the authentication handler type.
  - d. Click **Next**.  
The User Partition dialog box opens.
5. On the **User Partition** page, select **Global**, and then click **Next**.
6. On the **Parameters** page, enter the following.

Parameter	Value
Provider Name	Enter a name for the authentication provider. This name is displayed on the sign in page.
Active By Default	true
Enable cred validation	true
Enable token validation	true
Client ID	The client ID of AppWorks Platform that is registered in the content repository's OTDS.
Client Secret	The client password of AppWorks Platform that is registered in the content repository's OTDS.
Authorization Endpoint	<code>http://&lt;otds login endpoint of the content repository&gt;/otdssws/login.</code> <b>For example -</b> <code>http://&lt;host name&gt;:&lt;port number&gt;/otdssws/login</code>
Token Endpoint	<code>http://&lt;otds login endpoint of the content repository&gt;/otdssws/login</code> <b>For example -</b> <code>http://&lt;host name&gt;:&lt;port number&gt;/otdssws/login</code>
User Info Endpoint	<b>Core</b> - <code>https://&lt;FQDN of Core server&gt;/api/v1/users/self</code> <b>Documentum</b> - <code>http://&lt;otds login endpoint of the content repository&gt;/otdssws/oauth2/userinfo</code>
Send Access Token in Header	true
User Identifier Parameter	email

7. Click **Next**.

8. On the **Configuration** page, do the following:
  - a. From the list, select **mail**.
  - b. Click **Add**.The **mail** attribute is added to the **Authentication principal attribute** box.
9. Click **Save**.
10. From the Web administration menu, select the **Authentication Handlers** tab.
11. Search for `http.negotiate`, click **Actions**, and then select **Disable**.

### Enabling auto-provisioning of accounts

#### To enable auto-provisioning of accounts:

1. From the Web administration menu, click **System Config** tab.  
The list of System Attributes is displayed.
2. Navigate to the `directory.auth.AutoProvisionAccounts` system attribute and click the Value column.
3. Type **true**, and then click **Save**.

The authentication handler is now configured for the selected repository.

## Managing authorization

Authorization is the process of determining the types of activities that are permitted. It refers to the access rights granted to a user, program, or process. When a user tries to access an object or a resource, the authorization process checks the access privileges of that user. Permissions are generally defined by the administrator in the form of the Access Control List (ACL). Users must only be granted permissions to objects they can access and tasks they can perform.

After the system identifies a user through authentication, it uses authorization to determine what the user can do in it. This is defined in ACLs, where an administrator can set permissions for various resources. Role-based permissions and the privileges of an administrator are examples of configuring permissions.

ACLs are used to control access over the securable objects in AppWorks Platform. These lists define the ability to permit or deny the use of a particular resource by a user.

AppWorks Platform offers a robust information security framework based on ACLs.

Administrators can create an ACL to define which Web services can be invoked by specific users. Administrators can choose between fine-grained access control, that is Web service operation level or data field level, or a higher level access control, for example, database table level.

The Web-based ACL editor of AppWorks Platform also eliminates the manual effort of creating and managing the ACL.

The information about ACLs is stored in LDAP. The ACL entries for a particular organization can be modified only by the administrators.

This section contains the following topics:

Configuring AppWorks Platform Security Policy: You can configure security policies in AppWorks Platform through the Security Properties page of Management Console. See [Configuring AppWorks Platform security policy](#).

Managing ACL: You can manage authorization by regulating ACLs at various levels, such as users, Web services, service groups, metadata and LDAP object, and XML store objects. See [Managing an access control list](#).

## Configuring AppWorks Platform security policy

The security properties feature of the management console facilitates the configuration of security policies in AppWorks Platform. These policies are stored in the `securitypolicy.xml` file of the `<AppWorks Platform_installdir>/config` folder. This feature enables administrators to manage the security policies in AppWorks Platform for resource access management. Resources can be service groups, Web service interfaces, or Web service operations, for which access permissions can be configured for selected users.

To configure the AppWorks Platform security policy using the management console:

1. Depending on your operating system, do one of the following:

Operating system	Procedure
Windows	Click <b>Start &gt; Programs &gt; OpenText AppWorks Platform &lt;Version&gt; &gt; &lt;Instance Name&gt; &gt; Tools &gt; Management Console.</b>
Linux	<ol style="list-style-type: none"> <li>1. Launch Terminal.</li> <li>2. CD <code>&lt;AppWorks Platform_installdir&gt;/bin</code></li> <li>3. Run <code>./cmc.sh</code></li> </ol>

The Management Console window opens.

2. In the Management Console window click **Security Properties**.  
The Security Properties window opens.
3. In the Security Properties window, click **Security Policy**.
4. From the **Select Authenticated User** list, select the user for whom the access or deny permissions have to be set.
5. To grant or deny access permissions for the selected user to selected service groups, follow the procedure in [Setting permissions for service groups](#).
6. To grant or deny access permissions for the selected user to selected Web service Interfaces and Operations, follow the procedure in [Setting permissions for Web service interfaces and operations](#).
7. Click  (Save).  
The AppWorks Platform Security policy properties are saved.

## ***Setting permissions for service groups***

Use the management console to limit access to service groups in your application.

1. Based on your operating system, do the following:

<b>Windows</b>	Click Start > Programs > OpenText AppWorks Platform <Version> > <Instance Name> > Tools > Management Console
<b>Linux</b>	a. Launch Terminal. b. CD <AppWorks Platform_installdir>/bin c. Run ./cmc.sh

The Management Console window opens.

2. In the management console, click **Security Properties**.  
The Security Properties window opens.
3. Click the **Others** tab.
4. Select **Enable sandbox feature** and click  (Save).
5. Go to **SecurityPolicy** and from the Select Authenticated User list, select the user for whom the access or deny permissions need to be set.
6. To grant permissions to specific service groups:
  - a. Click **Grant Permission > Service Groups**.  
 (Add Service Group) is enabled on the toolbar.
  - b. Click , select the required service group from **Add Service Groups**, and click **Add**.

The selected service groups are added to the Grant Permissions list.

7. To deny permissions to specific service groups:
  - a. Click **Deny > Service Groups**.  
 (Add Service groups) is enabled on the toolbar.
  - b. Click , select the required service group from **Add Service Groups**, and click **Add**.
8. The selected service groups are added to the Deny Permissions list.
8. Click  (Save) on the Security Policy toolbar to save the policy changes.  
The grant or deny permissions are set for the service groups.

## ***Setting permissions for Web service interfaces and operations***

You may not want all users accessing all the Web service interfaces and Web service operations in your application.

### To restrict the access of Web service interfaces and operations:

- Based on your operating system, do the following:

<b>Windows</b>	Click Start > Programs > OpenText AppWorks Platform <Version> > <Instance Name> > Tools > Management Console
<b>Linux</b>	<ol style="list-style-type: none"> <li>Launch Terminal</li> <li>CD &lt;AppWorks Platform_installdir&gt;/bin</li> <li>Execute ./cmc.sh</li> </ol>

The Management Console window displays.

- Click **Security Properties**.

The Security Properties window displays.

- Click **Others**.

- Select the Enable sandbox feature check box and click .

- Click the **SecurityPolicy** tab and from the Select Authenticated User list, select the user for whom the access or deny permissions need to be set.

- To grant permissions to a Web service interface:

- Click **Grant Permission** > **Web Service Interfaces**.

- Click  on the toolbar, select the required Web service interface from the Add Web Service Interfaces window, and then click **Add**.The selected Web service interfaces are added to the Grant Permissions list.

- To grant permissions to a Web service operation:

- Click **Grant Permissions** > **Web Service Interfaces** > <**Web service Interface**>.

- Click  on the toolbar, select the required Web service operation from the Add Operations window, and then click **Add**.The selected Web service operations are added to the Grant Permissions list.

- To deny permissions to a Web service interface:

- Click **Deny** > **Web Service Interfaces**.

- Click  on the toolbar, select the required Web service interface from the Web Service Interfaces window, and then click **Add**.The selected Web service interfaces are added to the Deny Permissions list.

- To deny permissions to a Web service operation:

- Click **Grant Permissions** > **Web Service Interfaces** > <**Web service Interface**>.

- Click  on the toolbar, select the required Web service operation from the Add Operations window, and then click **Add**.The selected Web service operations are added to the Deny Permissions list.

10. Click on the Security Policy toolbar to save the policy changes.
11. Restart the Web services.

## Managing an access control list

Access Control Lists (ACLs) play a vital role in all administrative activities of AppWorks Platform. An ACL defines permissions deciding which user is allowed to see what information. It is used to control access over secure AppWorks Platform objects. It is not desirable for all users to have access to all objects in AppWorks Platform. Hence, access to certain objects is restricted to specific users in various organizations. If an ACL is not set, all users can access these objects by default.

ACL is meant for authorization, and not for authentication. Therefore, ACL settings can be configured for users and roles within an organization, but cannot be configured for authenticated users. The ACL setting is of two types:

- Unconditional ACL - See [Unconditional ACL](#).
- Conditional ACL - See Conditional ACL in the *AppWorks Platform API Reference Guide*.

The ACL settings in AppWorks Platform allow defining all the ACL objects in a single tree for a particular user or role. See ACL definitions in the *AppWorks Platform Advanced Development Guide* for more information.

AppWorks Platform provides default ACL settings for roles, users, service containers, XML store objects, and LDAP objects. Apart from these, an administrator can change ACL entries or set ACL settings for these items at the organization level. Administrators can set an access control list on the basis of ACL parameters. See ACL parameters in the *AppWorks Platform Advanced Development Guide*.

You can set ACL at the following levels:

- [Configuring ACL for service groups](#)
- [Configuring ACL for database metadata](#)
- [Configuring ACL for XMLStore objects](#)
- [Configuring ACL for Web service interfaces and operations](#)
- [Configuring ACL for LDAP objects](#)
- [Configuring ACL for users](#)
- [Configuring ACL for roles](#)

### Configuring ACL for service groups

You can configure ACL for a service group using the System Resource Manager task and authorize the access permissions on the service groups. For example, if you provide deny permissions to a user on the XForms service group, the user cannot access any XForm-based applications. You can provide the allow or block permissions to users and roles.

**Before you begin:**

You must have the role of systemAdmin or Administrator.

**To configure ACL for service groups:**

1. On the **Welcome** page > **My Applications**, click  (System Resource Manager).  
The System Resource Manager window opens.
2. Click  (Service Groups) on the toolbar.  
The Service Groups App Palette appears.
3. Right-click the required service group and select **Security**.  
The Security - <service group name> dialog box appears and displays a list of users and roles with the ACLs set.

**Note:** If ACLs are not set on a service group, the Security dialog box does not display any data.

4. Click **Add** button to select a user or role.  
The Organizational Users/Roles dialog box appears and displays a list of users and roles.
5. Select the required user or role to assign and click **OK**.  
The security window displays a list of available permissions for the users and roles.
6. To block the access permissions for a user or a role, clear the **Allow** check box.

**Note:** To remove an ACL entry from the **Security - <service group name>** dialog box, select it and click **Remove**.

7. Click **Apply**, and then click **OK**.

ACL is created and set for the service group.

### **Configuring ACL for database metadata**

This feature provides administrators with an option at run time to set or modify Access Control Lists (ACLs) on the database metadata objects such as tables, views, and procedures. For tables and views, you can set access control at various levels such as Read, Insert, Update, and Delete. For procedures, you can set only one type of access control, that is Allow.

You also have the option to set Conditional ACL on each metadata object (applicable only to Tables and Views). See Conditional ACL in the *AppWorks Platform API Reference Guide*. This option also provides the administrators with a facility to override the ACL set on the Database Metadata while developing the application. See Setting access control on tables and views of a database in the *AppWorks Platform Advanced Development Guide*.

1. On the **Welcome** page > **My Applications**, click  (Database Metadata Manager).  
The Database Metadata Manager window displays the contents of the configured WS-AppServer service.
2. Expand the contents of the WS-AppServer service too see tables, procedures, and views, and their contents.

3. Based on your requirement, right-click a **Table**, **Column**, **View**, or **Procedure** and select **Security**.

The Security App Palette opens on the right-side of the Metadata Explorer, displaying the name of the selected metadata object on which you can set Read, Update, Insert, and Delete permissions for users or roles.

4. On the Security App Palette, click **Add** to select the user or role for whom you want to set the ACL.

The Organizational Users/Roles - Users/Roles dialog box appears with a list of names, with the type and description for each user or role.

5. Select a user or role and click **OK**.

The selected user or role appears on the Security App Palette with permission granted by default for all the levels such as Read, Update, Insert, and Delete. There is a check mark for each level. For procedures, there is a check mark under Allow.

**Note:** For tables and views, the permission is hierarchical in nature, Read being the lowest level of permission and Delete being the highest level. So, if you deny Read permission, the denial applies to the remaining levels, and if you permit Delete, you grant permission to all the levels.

6. Do one of the following:

- Based on the level of access you want the particular user or role to have on the selected metadata object, select or clear the check box for a permission type.
- Set Conditional ACL on the metadata object (only tables or views) for the selected user or role. See Conditional ACL in the *AppWorks Platform API Reference Guide*.

**To switch to Conditional ACL:**

- a. Select the **Condition** check box.

A confirmation message to move to Conditional ACL displays.

- b. Click **Yes** on the message box.

The message box closes and **RUID** (Read, Update, Insert, and Delete) is displayed next to the Condition check box.

- c. Click **RUID**.

The Conditional ACL dialog box opens.

- d. Select an **Allow condition** (Read, Update, Insert, and Delete) from the options, provide the condition content in the text area and click **OK**.

The Conditional ACL dialog box closes and the RUID highlights one of the alphabets in red indicating the selection. For instance, if you selected Read, R is shown in red.

7. Click **Apply**, and then click **OK**.

The permission level on that metadata object is defined for the selected user or role.

## Configuring ACL for XMLStore objects

You can define ACLs for the XMLStore objects using the XMLStore Explorer and authorize the access permissions on applications, organizations, and users available in the XMLStore. You can provide allow or block permissions to users and roles for accessing the XMLStore objects.

**Before you begin:**

You must have the role of systemAdmin or Administrator.

**To configure ACL for XMLStore objects:**

1. On the **Welcome** page > **My Applications**, click  (XMLStore Explorer).  
The XMLStore Explorer window opens and displays the Collection folder containing all the files and directories in the XML Store.
2. Expand the **Collection** folder.  
The expanded tree list displays various folders containing applications and other objects.
3. Expand the tree, right-click the required XML object, and select **Security**.  
The Security dialog box displays a list of permissions for the users or roles.
4. Click **Add**.  
The Organizational Users/Roles dialog box displays a list of users and roles.
5. Select the user or role for which ACL is to be configured and click **OK**.  
The Security dialog box displays a list of permissions for the users or roles.  
**Note:** If there are no existing users or roles, a blank window opens.
6. To block the access permissions for a user or role, clear the **Allow** check box, which is selected by default for all new users and roles.  
**Note:** To remove an ACL entry in the Security - <service group name> dialog box, select it and click **Remove**.
7. Click the **Apply**, and then click **OK**.  
Setting ACLs for XML objects involves an additional property called version that indicates the version for which ACL is being set. If the object is a folder under the XMLStore Explorer, the versions are displayed in a dropdown list. See Versioning in XMLStore in the *AppWorks Platform Advanced Development Guide*.

### **Configuring ACL for Web service interfaces and operations**

You can configure the access control settings on Web service operations to restrict the user from invoking the Web service interface or operation. This controls the rights of a user to run the Web service. You can set Access Control List (ACL) permissions for Web service interfaces and Web service operations from the **Web Service Interface Explorer**. Setting security permissions on Web service interface or operation enables access controls on the namespaces associated with them.

**Note:** These ACL settings are the runtime modifications and they override the ACLs set during design-time. See Setting access control on Web service interface and Web service operations in the *AppWorks Platform Advanced Development Guide*.

**Before you begin:**

You must have the role of systemAdmin or organizationalAdmin.

### To configure ACL for Web service interfaces and operations:

1. On the **Welcome** page > **My Applications**, click  (Web service Interface Explorer). Alternatively, click  on the toolbar of the System Resource Manager window. The Web Service Interface Explorer window opens with default search query options. See [Searching for Web services in Web service Interface Explorer](#).
2. In **Keyword**, type the required Web service interface or Web service operation you want to secure and click **Find**. The names of the Web service interfaces or Web service operations display in the Search Results group box.
3. Right-click a Web service interface or Web service operation and select **Properties**. The Web Service Interface Properties - <Name of the Web Service Interface> dialog box opens. This dialog box displays the Web service interfaces and their operations in corresponding App palettes.
4. Do one of the following:

To configure ACL on	Procedure
Web service interface	In the Web Service Interface App Palette, click  (Extend Menu) and select <b>Security</b> .
Web service operation	In the Web Service Operation App Palette, right-click the required operation and select <b>Security</b> .

The Security dialog box displays a list of users and roles to which ACL has been set.

5. Click **Add** to set access controls for a new user or role. The Organizational Users / Roles dialog box displays a list of users or roles.
 

**Note:** While invoking a Web service interface or operation, the ACLs configured at the user level take higher priority than the ACLs at the role level.
6. Select a user or role and click **OK**. The selected roles or users are populated in the Security dialog box.
 

**Note:** To block access to a Web service, clear the Allow check box against a user or role.
7. To remove ACLs associated with a user or role, select the required user or role and click **Remove**.
8. Click **Apply** and **OK**.

## Configuring ACL for LDAP objects

You can define Access Control Lists (ACLs) for Lightweight Directory Access Protocol (LDAP) objects using the LDAP Explorer and authorize the access permissions on various system resources. You can provide or block permissions for users and roles for accessing the LDAP objects. Only the user, or the user with the role on which the ACLs are defined experience the impact of the ACL.

### Before you begin:

You must have the role of systemAdmin or organizationAdmin.

1. On the **Welcome** page > **My Applications**, click  (LDAP Explorer).  
The LDAP Store Explorer window displays the list of LDAP objects such as organizations, authenticated users, and applications, with their content.
  2. Expand the tree and select the required LDAP object to set an ACL.
  3. Right-click the **<object>** and select **Security**.  
The Security dialog box displays the list of users or roles on which ACL have been set.
  4. Click **Add** to set ACL for a user or role.  
The Organizational Users/Roles dialog box displays the list of users or roles.
  5. Select the required user or role and click **OK**.  
The Security dialog box displays the list of permissions for users or roles.
- Note:** If there are no existing users or roles, the window is blank.
6. Select one of the required options:
    - Read
    - Update
    - Insert
    - Delete
  7. Click **Apply**, and then **OK**.

The ACL is set at the LDAP object level.

## Configuring ACL for users

ACLs can be configured to enable or restrict user from accessing applications.

### Before you begin:

You must have the role of systemAdmin or organizationalAdmin.

1. On the **Welcome** page > **My Applications**, click  (User Manager).  
The User Manager window opens.
2. Select the **Users - Roles** or **Users - Tasks** view in the User Manager window. See [Users - Roles view](#) and [Users - Tasks view](#).
3. In the Users pane, right-click the user for which ACL must be set and select **Security**.  
The ACL Explorer dialog box displays the existing ACLs.

4. Click  (Add).  
The Access Control Set - New Access Control Set dialog box opens.
5. Type the required details and click  (Save).  
See ACL explorer interface in the *AppWorks Platform Advanced Development Guide* for more details.

**Note:** To delete an ACL, right-click the required ACL, and then select **Delete**. Alternatively, you can click  in the Access Control Set window.

### Configuring ACL for roles

By configuring Access Control Lists (ACLs) for a role, you can define the privileges for that role. If this role is assigned to a user, based on the restrictions associated with the role, the user is granted or denied access to the applications on AppWorks Platform. See [Assigning roles to users](#). You cannot set ACL on application roles. See [Managing roles](#).

#### Before you begin:

You must have the role of systemAdmin or organizationalAdmin.

1. On the **Welcome** page > **My Applications**, click  (User Manager).  
The User Manager window opens.
2. Select one of the following views depending on your ACL requirements:
  - Roles - Roles - See [Roles - Roles view](#)
  - Roles - Users - See [Roles - Users view](#).
  - Roles - Tasks - See [Roles - Tasks view](#).
3. In the Roles pane, right-click the required role and select **Security**.  
The ACL Explorer dialog box displays the existing ACLs.
4. Click  (Add).  
The Access Control Set - New Access Control Set dialog box opens.
5. Type the required details and click  (Save).  
See ACL configuration interface in the *AppWorks Platform Advanced Development Guide* for more details.

**Note:** To delete an ACL, right-click the required ACL, and then select **Delete**. Alternatively, you can click  in the Access Control Set window.

## Digital certificates and trust stores

Digital certificates are used for identifying an entity. Apart from identification, certificates can be used for signing software files and messages.

When two parties want to interact using digital certificates, they must approach a Certificate Authority (CA) that issues certificates. The CA is a third-party organization that both the parties trust. Once the certificates are issued by the CA to an entity, the entity can use the

certificate to identify itself. An entity can have more than one certificate, and each certificate can be used for various purposes. A digital certificate contains the name and other identification information of the certificate holder, such as a serial number, expiration date, a copy of the public key of the certificate holder, and the digital signature of the certificate issuing authority. A certificate is issued for a limited period of time. After that period, the certificate expires and is no longer valid. If you suspect that a certificate is compromised, it can be revoked. In these cases also the certificate is invalid. The life cycle of the certificate begins when the certificate is issued and it ends when the certificate expires or is revoked. See [Certificate validation](#) to understand how certificate validation is handled in AppWorks Platform.

AppWorks Platform uses digital certificates for various purposes. For information about the use of digital certificates in AppWorks Platform, see [Using a digital certificate](#). All the certificates of an organization are placed in a repository called certificate store. This repository is managed with the security administration task. The certificates that are trusted by an organization are configured in the certificate store of that organization. For each organization, a fallback repository level called platform is also available to manage the certificates. The certificates in the fallback repository are trusted by the organization.

In AppWorks Platform, certificates are used for signing applications. See [Application signing](#) for more information. For Secure Sockets Layer (SSL), trust anchors must be configured in the security administration task. See [Managing a certificate](#) for more information. SSL/Transport Layer Security (TLS) is widely used for digital certificates. In AppWorks Platform, the outbound SSL/TLS connections through UDDI are managed with security administration. See [Using SSL in Platform connectors](#).

## Managing a certificate

All certificates that are trusted by an organization are placed in a repository called the certificate store. The certificate store can be found on the first tab of the Security Administration task. Certificates that are used for signing applications, signing messages exchanged between service groups, or for validating SSL connections must be registered. A certificate is only valid in scenarios wherein the complete chain is available, the root of the certificate chain is registered in the certificate store, the certificate has not expired or revoked. See [Troubleshooting Certificate Status and Revocation](#) or [Certificate Revocation List](#) or [Certificate types](#) for more information.

The **View certificates trusted on** list on the **Certificates** tab of the Security Administration task is the switch between the platform-level certificate store and the organization-level certificate store. Any update done when switched to the organization level influences your organization. Similarly, any update done at the platform level influences only the platform, which can only be managed by the system administrator.

The AppWorks Platform certificate store contains the default Java certificates that are located in the cacerts keystore of the Java installation, and additional certificates added by administrators at an organization or platform level. The Java default store supports default trust.

By default, the revocation checking of certificates is disabled. To turn it on, see [Security administration properties](#).

You can perform the following tasks on the **Certificates** tab of the Security Administration task:

- [Adding a new certificate](#)
- [Validating a certificate](#)
- [Viewing certificate details](#)

### **Certificate validation**

The Security Administration service validates Certificate Chains against the own Certificates store. The certificates are signed with a digital signature of a Certificate Authority (CA). The certificate of that CA is needed, henceforth till the root CA. A root CA is trusted only when it appears in a selective list of trust anchors. This list is managed in the Certificates store of the Security Administration task. For more information, see [Managing a certificate](#).

A CA can revoke a published certificate by publishing it in a Certificate Revocation List (CRL). A CRL contains a list of revoked certificates, and is protected with expiry and digital signature means. A certificate can contain a CRL location. The Security Administration service downloads the CRL (optional), and checks the revocation status of the certificates. The Security Administration service will mark a certificate as invalid when the CRL is not available. This is done to prevent the DoS attacks on CRL servers that give attackers an advantage.

**Note:** Online CRL retrieval might result in runtime stability issues. See Deployment considerations below for more information.

For third party information on this subject, see [Troubleshooting Certificate Status and Revocation](#), [Certificate Revocation List](#) and [Certificate types](#).

### **Deployment considerations**

For complete certificate validation it is necessary to have access to up-to-date CRLs. Normally the CRLs have an expiry period of few weeks and online access is also needed. Failing to download a CRL causes the certificate validation to report an invalid certificate; hence, the UDDI request is blocked as a valid SSL connection is needed. This setup makes the functioning of the AppWorks Platform solution dependent on third-party servers over the internet. This might result in the following:

- When AppWorks Platform does not have internet access, this causes failures
- Internet access interruptions causes failures
- Unresponsiveness of CRL servers causes failures

Because of above scenarios, CRL checking is disabled by default.

### **Enabling CRL checking**

CRL checking can be enabled by setting the following Security Administration property:  
`certificatemanager.validation.revocation.enabled=true`

A more manageable way of CRL checking can be done through [Online Certificate Status Protocol \(OCSP\)](#). With OCSP the Security Administration service connects to a single (on-premise) server to fetch the certificate status. With such a setup, the AppWorks Platform solution is not dependent anymore on a third-party services as it leverages the complete SSL security.

OCSP can be enabled by setting the following Security Administration property:

```
ocsp.responderURL=<ocsp responder endpoint>
```

### **Third party examples of OCSP responder products**

RSA Validation manager: <http://www.rsa.com/>

Active Directory Certificate Services: <http://technet.microsoft.com/en-us/windowsserver/dd448615.aspx>

### ***Adding a new certificate***

Certificates at the shared level are applicable to all the organizations and certificates at the organization level are applicable to that specific organization only. Based on the target location of the certificates, you must have the following roles to add, delete, and update the certificates.

- Shared level - Security System Administrator
- Organization level - Security Administrator

Adding a Digital Certificate registers the certificate in AppWorks Platform.

#### **Note:**

- Certificates from AppWorks Platform installation and the Java Runtime Environment (JRE) are available in the Certificates tab by default (shared level).
- All the other certificates that you add, based on the level they are added, are available in their corresponding (shared or organization) levels.

#### **To add a certificate:**

1. On the Welcome page, click  (Security Administration).  
The Security Administration window opens and the Certificates tab is displayed by default. It contains a table showing all the trusted certificates from the AppWorks Platform keystore and the default Java keystore (cacerts).
2. From the View certificates trusted on list on the Certificates tab, select the required level to add the certificate.
3. Click .  
The Load New Certificate window opens.
4. Upload or paste the relevant certificate.
5. Optionally, you can change the alias under which the certificate is stored.
6. Click **OK** to import the certificate.

The certificate is added.

## Validating a certificate

Certificates must be validated before they are used. Digital certificates can be revoked if they are compromised. Expired certificates are not valid. You can also validate if the certificate is meant for a certain purpose. For example, a certificate used for signing applications must be validated to check if the Key Usage field in the certificate is for code signing.

### Before you begin:

- Ensure that details of the certificate, such as root certificate and Certificate Revocation List (CRL) details, are configured in the trust store.
- Ensure that you have the security administrator role to validate the certificate details.

### To validate a certificate:

1. On the **Welcome page > My Applications**, click  (Security Administration). The **Security Administration** window opens and the **Certificates** tab is displayed by default.
2. Click **Certificate test tool** on the top right of the table. The **Validate Certificate** dialog box opens.
3. In the **Select Mode** list, click one of the following options:
  - **Upload certificate:** Uploads the certificate.
  - **Paste base64 encoded certificate content:** Pastes the contents of an existing certificate. Paste the certificate details in the Base64 encoded certificate text area and skip the next step.
4. Click **Browse** next to the **Certificate** box, select the certificate to be validated from the relevant location, and click **Open**.  
The certificate name appears.
5. In the **Key Usage** list, select the type of key usage you want to validate for the certificate.
6. In the **Extended Key Usage** list, select the type of extended key usage of the certificate.
7. Click **Validate**.  
The validity status of the certificate is displayed in the **Validation Status** box.

**Note:** To clear the options selected in the **Validate Certificate** dialog box, click **Reset**.

## Viewing certificate details

After adding a certificate to the certificate store, you can view the details of the certificate.

1. On the **Welcome page > My Applications**, click  (Security Administration). The Security Administration window opens and the Certificates tab is displayed by default.

2. Double-click the required certificate.

The Certificate Details window with the following details opens:

- Name of the certificate.
- Validation status (this is retrieved asynchronously, because it involves a chain validation).
- Certificate chain of the certificate as far as it is stored in the certificate store.
- All attributes of the certificate in the Details panel.
- X509 data in the Details panel that can be used to paste the certificate in the import dialog or in a .cer file.

## ***Security administration properties***

The following table describes the various properties related to certificate validation:

<b>Component</b>	<b>Property name</b>	<b>Default value</b>	<b>Description</b>
Security Administration	certificatemanager.keystore.location	\$JAVA_HOME/lib/security/cacerts	Location of the Java keystore file.
	certificatemanager.keystore.java.enabled	true	Enables the Java keystore.
	certificatemanager.keystore.expiry	3600 (1 hour)	Keystore is reloaded when <b>expiry time</b> is passed.
	certificatemanager.validation.revocation.enabled	false	Enables AppWorks Platform revocation checking of certificates.

<b>Component</b>	<b>Property name</b>	<b>Default value</b>	<b>Description</b>
	com.ibm.jsse2.checkRevocation	true	Enables revocation checking. <b>Note:</b> Applicable only when IBM Java security provider is used.
	ocsp.enable	true	Enables OCSP protocol to be used for revocation check.
	com.ibm.security.enableCRLDP	true	Enables CRL retrieval through certificate distribution points. <b>Note:</b> Applicable for IBM security provider only.
	com.sun.security.enableCRLDP	true	Enables CRL

<b>Component</b>	<b>Property name</b>	<b>Default value</b>	<b>Description</b>
			<p>retrieval through certificate distribution points.</p> <p><b>Note:</b> Applicable for SUN security provider only.</p>
	ocsp.responderURL		Sets default OCSP responder URL; overrules individual responder URLs found inside certificates.
	com.eibus.security.x509.validCertificateCache	1000	Cache size for validated certificate chains.
	com.eibus.security.x509.validCertificateCache.expirytime	8 hours	Time after which validate

<b>Component</b>	<b>Property name</b>	<b>Default value</b>	<b>Description</b>
			d certifica te chains are remove d from the cache.
Java Secure Socket Extension (JSSE)	All		JSSE Referen ce Guide.

## Application signing

When an application is developed, there are high chances of the application being tampered before deployment. Therefore, to prevent such tampering, it must be signed. A signed application indicates that the package:

- is genuine
- has not been tampered with
- is from a trusted party

The provision of working with signed applications assures the AppWorks Platform Administrator that no tampered application can be installed in AppWorks Platform. A certificate can be used to trace the entity that signed it. Therefore, signing an application ensures:

- Integrity of the application - assurance that the application is not tampered with.
- Identity of the author - assurance that the application is from the entity that it is supposed to be from.

AppWorks Platform provides a Package Signer tool to sign your applications. See [Signing packages](#) for more information on using the tool.

When you install a signed application into AppWorks Platform, the certificate used to sign the application is verified and validated. Verification is done to check if the certificate is indeed from the party that signed the application. The application must be signed by a trusted publisher. Certificates of trusted publishers are registered in the Code Signing tab of the Security Administration task.

In some cases, the applications you install in AppWorks Platform might be tampered with, might not be signed, or their certificates might not be available in the trust store. It might also be that the certificate used to sign an application might not be meant for signing applications. As an Administrator, you are required to manage such cases. AppWorks Platform provides you with an interface to manage such scenarios. For information on using security settings while installing applications, see [Configuring security settings for application installer](#).

## ***Configuring security settings for application installer***

Signed applications ensure integrity of applications and also assure the identity of the signing authority. To prevent installation of tampered applications, AppWorks Platform provides some default security settings. You can change the default security settings as follows:

### **Before you begin:**

You must have the role of a systemAdmin to configure security settings.

### **To configure the security settings for application installer:**

1. On the **Welcome** page > **My Applications**, click  (Security Administration). The Security Administration window opens.
2. Click the **Code Signing** tab.  
For information on the fields in this tab, see [Security settings and options](#). It is recommended to leave the settings as Disallow.
3. In the **Validity of Application Signature** area, select the settings for the applications that must be installed.

**Note:** The settings for the unsigned applications depends on the settings for the tampered applications.

**Note:** The following table provides the options available for the unsigned applications against each setting of the tampered applications:

<b>Setting for tampered applications</b>	<b>Setting for unsigned applications</b>
Disallow	<ul style="list-style-type: none"> <li>■ Disallow</li> </ul>
Prompt	<ul style="list-style-type: none"> <li>■ Disallow</li> <li>■ Prompt</li> </ul>
Allow	<ul style="list-style-type: none"> <li>■ Disallow</li> <li>■ Prompt</li> <li>■ Allow</li> </ul>

4. In the **Validity of Signing Certificate** area, select the required settings for the signing certificates.

5. Click .

The security settings are saved.

## Managing SAML trust

WS-Security SAML Token Profile is a method of user authentication that AppWorks Platform uses. It enables external applications or components to send a SOAP request with a SAML 2 assertion in the WS-Security SOAP header. The AppWorks Platform identity framework uses the user identity available in the SAML assertion to resolve the actual AppWorks Platform user. Typically, a SAML assertion is digitally signed. This allows the receiver to validate the origin of the SAML assertion. To validate the SAML assertion, a trust relation has to be configured between the sender and receiver. After the trust relation is configured, the AppWorks Platform can validate the digital signature of the SAML assertion.

The SAML assertion also contains additional conditions, such as audience restriction and the time-frame of the SAML assertion validity. See [Example of a SAML assertion](#) for more information.

### Managing trust

To trust external SAML assertions provided in the AppWorks Platform:

- Add the certificate that is used to sign the SAML assertion to the SAML trust store. The actual certificate that is used to sign the SAML assertion must be added to the SAML trust store. Only the certificates present in the SAML trust store are used for SAML assertion validation.
- Add the complete certificate chain of the signing certificate to the generic trust store. All certificates in the chain of the signing certificate must be available in the generic trust store.

The signing certificate and the certificates in the certificate chain can be added at a shared or organization level. Certificates added to the shared trust store can be used for the validation of SAML assertions of all organizations. Certificates added to the organization trust store can only be used within that organization.

See [Adding a certificate to SAML trust store](#) for the procedure to add a certificate to the SAML trust. The changes are applied automatically without the need to restart any service containers.

See [Adding a new certificate](#) for the procedure to add certificates from the certificate chain.

Removing a certificate from the SAML Trust store also invalidates the trust relation with the SAML assertion provider.

### Audience restriction

SAML assertions can contain conditions. One condition is audience restriction, that enables the receiver to confirm the SAML assertion provided is actually meant for them.

The default value expected for an AppWorks Platform instance is its instance name or the value of the wcp property, `AppWorks Platform.instance.name`.

You can configure the expected audience restriction value by setting the property `cordys.inboundsaml2.audiencerestriction.template` in the `wcp.properties` file to the desired value or pattern. The following properties can be used in the pattern:

Variable name	Value
<code>\$INSTANCE</code>	Value of the WCP property <code>AppWorks Platform.instance.name</code> . This is the default value. For example, <code>defaultInst</code> .
<code>\$ORGANIZATION</code>	The name part of the current organization content. For example, <code>system</code> or <code>test</code> .

The following are the samples of the audience restriction property values:

```
cordys.inboundsaml2.audiencerestriction.template=testSP
cordys.inboundsaml2.audiencerestriction.template=$INSTANCE
cordys.inboundsaml2.audiencerestriction.template=test$instance
cordys.inboundsaml2.audiencerestriction.template=$ORGANIZATION_$INSTANCE
```

If the property value is empty, the audience restriction check cannot be performed.

The following is a sample SAML assertion condition:

```
<saml2:Conditions NotBefore="2014-02-04T14:26:30Z" NotOnOrAfter="2014-02-04T14:46:30Z">
  <saml2:AudienceRestriction>
    <saml2:Audience>defaultInst</saml2:Audience>
  </saml2:AudienceRestriction>
</saml2:Conditions>
```

## Using SSL in Platform connectors

Using plain Hypertext Transfer Protocol (HTTP) for communication over the Internet allows third parties to listen to or tamper with the data being sent. To secure the communication between a client and the server, use HTTP Secure (HTTPS). The Secure Sockets Layer (SSL) and Transport Layer Security (TLS) protocols bring the security and privacy while communicating through HTTPS.

TLS is the successor of SSL. SSL is not secure, hence in practice only the TLS protocol is used.

**Note:** Whenever SSL is used below, TLS is also implied.

### Components

There are a couple of Platform connectors that have a specific AppWorks Platform library that handles the validation of the SSL protocol. This library requests the security

administration trust store to validate the SSL certificate, and verifies the `hostname`. The SSL certificates are maintained in the AppWorks Platform Trust store.

The following connectors use this specific SSL socket library:

Connector	Description
FTP connector Web services	See <a href="#">Creating a configuration profile for an FTP server</a> .
HTTP connector	See <a href="#">Developing web services using the HTTP connector in the AppWorks Platform Advanced Development Guide</a> .
UDDI connector	See <a href="#">Working With External Web services in the AppWorks Platform Advanced Development Guide</a> .

## SSL/TLS validation

### To validate SSL/TLS for these connectors:

1. Check the certificate validity.  
See [Certificate validation](#) for more information. The SSL protocol passes the entire chain of certificates to the AppWorks Platform client. This chain is passed to the security administration service to enable validation, even if the chain is not fully known within AppWorks Platform. An integrity check on the chain is part of the certificate validation algorithm.
2. Verify the requested server host name with the content of the certificate, using the subject or subjectAltName attributes.

### UDDI SSL specific configuration settings

- For backward compatible properties related to UDDI, see [UDDI connector properties](#).
- For backward compatibility and to disable `hostname` verification against the SSL certificate in the UDDI, use: `uddi.http.connection.verifyhostname=false`
- To disable the validation of the SSL connection, use the configuration flag `Ignore Certificate Validation`. See the [UDDI connector interface](#) for more information. As a result, the certificates and hostnames are not verified.

### Client authentication with SSL

2-way SSL, also called client authentication or mutual authenticated connections, is based on certificates provided by both sides of the connection. Both the client and the server provide a certificate to the other party to be validated.

When a connection is set up with 2-way SSL, the client validates the certificate by the server. When no client authentication is done, validation is done against the Platform Trust store.

After the client validates the server certificate, it sends the client certificate to the server for validation. The client certificate is taken from the configured key store. The connectors need access to the private key of the client certificate to authenticate themselves with the external service. This information must be available in a Java Key Store (JKS) file on the server.

The following properties are used to configure the use of the JKS. They are put in the `wcp.properties` file or used as JVM property (-D..). See Service container properties interface in the *AppWorks Platform Advanced Development Guide* for more information. A JVM property setting overwrites the setting provided in `wcp.properties`.

The properties are checked in the following order:

Property	Description
<code>cordys.net.ssl.keyStore</code>	Path to JKS.  Example: <code>cordys.organization.acme.net.ssl.keyStore=\\opt\\pp108\\certificates\\keystore\\uddi_external_ws.jks</code>
<code>cordys.net.ssl.keyStorePassword</code>	Plain-text password to access the key store and the key in the key store.
<code>javax.net.ssl.keyStore</code>	Path to JKS.  Example: <code>cordys.organization.acme.net.ssl.keyStore=\\opt\\pp108\\certificates\\keystore\\uddi_external_ws.jks</code>
<code>javax.net.ssl.keyStorePassword</code>	Plain-text password to access the key store and the key in the key store.

### To use a Java key store for 2-way SSL:

1. Create or get a Java Key Store (JKS) with the client certificate and private key, and place it on the AppWorks Platform server.
2. Use the properties described to specify the path to JKS as key store and password.
3. Restart the connector.

**Note:** Each Java key store has only one client certificate.

### Backward compatibility

For backward compatibility, use the `uddi.keystore` property. The [AppWorks Developer community](#) provides additional resources that you may find helpful.

**Note:** This forces the UDDI service to work in the backward compatibility mode, where no integration with the security administration is used.

## Managing gateway with security options

Transport Level Security (TLS), the successor of SSL, is used on Web applications to protect sensitive information through the HTTPS protocol. AppWorks Platform Web Gateway can be configured to use TLS to secure access to AppWorks Platform applications.

Users can also sign in to AppWorks Platform applications using digital certificates. The AppWorks Platform gateway running in the Web server can be configured to enable digital certificate-based logging.

# Chapter 4

## Managing users and roles

User and role management describes activities that relate to users and roles. User and role management can also be called user provisioning, which refers to creation, maintenance, and deactivation of user objects and user attributes. The User Manager provides a way to manage users, roles, and associated attributes. It facilitates assigning of privileges to AppWorks Platform users. As a user, you have permissions through various roles to work in an organization. You can also assign tasks and teams to users and govern their operation.

**To access User Manager:**

1. On the Welcome page, click  (User Manager).  
The User Manager window opens.
2. Select one of the following views:

View	Description
Users - Roles	See <a href="#">Users - Roles view</a>
Roles - Users	See <a href="#">Roles - Users view</a>
Roles - Roles	See <a href="#">Roles - Roles view</a>
Roles - Tasks	See <a href="#">Roles - Tasks view</a>
Users - Tasks	See <a href="#">Users - Tasks view</a>

For information about the tasks that can be performed from these views, see [User Manager](#).

This section includes the following topics:

- [Managing users](#)
- [Managing roles](#)

**Note:** OpenText recommends not using the same role for administration activities and application development. Create new users with application-specific roles. This customization prevents security leaks.

For example, the System Admin role must be used for administrative activities, such as installing an application and not to develop or access applications.

## User Manager

User Manager facilitates different views to manage users and roles. The following table provides a mapping between the tasks and the corresponding views to perform them.

Tasks	View
Creating users	<ul style="list-style-type: none"> <li>■ Users - Roles</li> <li>■ Users - Tasks</li> </ul>
Creating multiple users	<ul style="list-style-type: none"> <li>■ Users - Roles</li> <li>■ Users - Tasks</li> </ul>
Creating a Role	<ul style="list-style-type: none"> <li>■ Roles - Roles</li> <li>■ Roles - Users</li> <li>■ Roles - Tasks</li> </ul>
Setting ACLs on Users	<ul style="list-style-type: none"> <li>■ Users - Roles</li> <li>■ Users - Tasks</li> </ul>
Setting ACLs on Roles	<ul style="list-style-type: none"> <li>■ Roles - Roles</li> <li>■ Roles - Users</li> <li>■ Roles - Tasks</li> </ul>
Deleting a User	<ul style="list-style-type: none"> <li>■ Users - Roles</li> <li>■ Users - Tasks</li> </ul>
Deleting a Role	<ul style="list-style-type: none"> <li>■ Roles - Roles</li> <li>■ Roles - Users</li> <li>■ Roles - Tasks</li> </ul>
Cloning a User	<ul style="list-style-type: none"> <li>■ Users - Roles</li> <li>■ Users - Tasks</li> </ul>
<ul style="list-style-type: none"> <li>■ Assigning Roles to Users</li> <li>■ Revoking Roles from Users</li> </ul>	<ul style="list-style-type: none"> <li>■ Users - Roles</li> <li>■ Roles - Users</li> </ul>
<ul style="list-style-type: none"> <li>■ Assigning Sub-Roles</li> <li>■ Revoking Sub-Roles</li> </ul>	Roles - Roles
<ul style="list-style-type: none"> <li>■ Assigning Tasks to Users</li> <li>■ Revoking Tasks from Users</li> </ul>	Users - Tasks
<ul style="list-style-type: none"> <li>■ Assigning Tasks to Roles</li> <li>■ Revoking Tasks from Roles</li> </ul>	Roles - Tasks

## Users - Roles view

To access the roles view:

On the **Welcome page**, click  (User Manager) and select **Users - Roles view** from the Select View list.

### Features

This is the default view of the User Manager window. Some features:

- The left pane displays the users and the right pane displays the roles.
- Use the **Find** feature to find users or roles. Click anywhere on the pane to focus the cursor and press **CTRL+F**. The Find dialog box opens at the bottom of the pane. Enter a letter (with which the name or ID starts) and the relevant results are highlighted.
- A series of letters is listed as buttons adjacent to the left pane. Click a button and the associated names or IDs are displayed.
- For names starting with characters other than letters and numbers, click  (Button), for example, '( )'.
- The **User Name**, **User ID**, and the **number of roles** associated with the user are indicated in the **Users** pane.
- Select a user, and the associated roles are highlighted in the **Roles** pane.
- Select multiple users, and the roles common to the selected users are highlighted.

## Users - Tasks view

### Accessing the view

On the **Welcome page**, click  (User Manager) and select **Users - Tasks view** from the Select View list.

### Features

Some features of the tasks view are:

- The left pane displays the users and the right pane displays the tasks.
- Use the **Find** feature to find users or tasks. Click anywhere on the pane to focus the cursor and press **CTRL+F**. The Find dialog box opens at the bottom of the pane. Enter a letter (with which the name or ID starts) and the relevant results are highlighted.
- A series of letters is listed as buttons adjacent to the left pane. Click a button and the associated names or IDs are displayed.
- For names starting with characters other than letters and numbers, click  (Button), for example, '( )'.

- The **User Name**, **User ID**, and the **number of roles** associated with the user are indicated in the **Users** pane.
- Select a user, and the associated roles are highlighted in the **Tasks** pane.
- Select multiple users, and the tasks common to the selected users are highlighted.

## Managing users

A user refers to someone who accesses AppWorks Platform and performs administrative operations or develops applications. Users must identify themselves with a User ID to access AppWorks Platform. User ID governs the authorization of a user to perform any action. Once the user logs into the AppWorks Platform environment, the web server validates these credentials and provides access to the user.

Users can access applications based on the roles assigned to them. These roles determine the scope of user actions. As an administrator, you can assign the necessary roles and govern the user behavior.

### User types

The following types of users can be created in AppWorks Platform. Each represents an authentication type of user.

#### External users

External users are created in AppWorks Platform, but are authenticated by external components or identity providers, such as Windows Authentication or Web server authentication. SAML2 supports identity providers like OpenSSO. External users do not need a password because they are authenticated outside AppWorks Platform. For example, `ntdom/gharry`, where `ntdom` is the domain and `gharry` is the login ID.

#### AppWorks Platform users

AppWorks Platform users are created and authenticated in AppWorks Platform based on their user ID and password. AppWorks Platform authenticates these users using their login credentials.

When users are authenticated, the AppWorks Platform SSO component provides a signed SAML assertion and a cookie-based security artifact. See [Security Assertion Markup Language](#). When a user requests for services, this cookie-based security artifact is validated and used to identify the user. To login as custom users, you must configure anonymous access in the web application server. See [Web server authentication](#). For example, `gharry/Pswd`, where `gharry` is the login ID and `Pswd` is the password provided.

**Note:** An AppWorks Platform user and an external user cannot sign in AppWorks Platform at the same time. Configuring anonymous access in the Web server allows the AppWorks Platform users to log into AppWorks Platform environment. However, this change in the context of authentication will not recognize the External Users. Hence, you

must switch the context and sign in as an AppWorks Platform user or an external user at a time.

**Note:** If a user is created with the same ID of an existing authenticated user, the user can be mapped to that authenticated user, if both the users belong to the same type of users. For example, let us assume that you have already created a domain user A1 and if you try to create a custom user called A1, you cannot map it to the pre-existing domain user A1 because they do not belong to the same category of users. However, if you have created a custom user A1 and are trying to create another custom user A1, you can connect them. In addition, you can map the same type of users with the same IDs only in different organizations.

## Working with User Manager

The administrator can perform the following tasks in User Manager:

- [Creating users](#)
- [Cloning users](#)
- [Disabling users](#)
- [Deleting users](#)
- [Assigning roles to users](#)
- [Revoking roles from users](#)
- [Assigning tasks to users](#)
- [Revoking tasks from users](#)
- [Viewing the profile of a user](#)
- [Changing the default organization of a user](#)

### User Manager views

The User Manager view is split into two panes: the left pane and right pane. Depending on the selected view, the User Manager artifacts are displayed in the corresponding panes. There are seven different views in User Manager:

View	Description
Users - Roles	Displays users list on the left pane and roles on the right pane.
Roles - Roles	Displays roles and associated roles.
Roles - Users	Displays roles on the left pane and users list on the right pane.
Roles - Tasks	Displays list of roles and associated tasks.
Users - Tasks	Displays list of users and associated tasks to the selected user.

<b>View</b>	<b>Description</b>
Teams - Users	Displays list of teams and associated users of the teams.
Users - Teams	Displays the users list and the teams they are associated with.

## Show All

The **Show All** feature is applicable only to the artifacts in the left pane of the User Manager. When you click **Show All**, based on the view selected, all the corresponding artifacts are displayed in the left pane.

## Search for users

The default view in the User Manager is **Users - Roles** which displays only the users who are currently logged. The **Search** feature enables you to search for an artifact in the left pane of the User manager.

- To search for an artifact, type any of the characters in the name of the artifact. For example, to search for the user with the name John, you can enter the characters J, Joh or Jon.
- The list of the artifacts matching with the search criteria is displayed. To view all the artifacts, leave the search field empty and click on the **search** button.

All the artifacts are displayed in the left pane according to the selected view.

## Creating users

To access AppWorks Platform features, users must be part of an organization as a user. During user creation, define the login credentials for the user. Login credentials are validated when the user attempts to access AppWorks Platform.

User Manager provides the flexibility of creating more than one user at a time. The procedure is similar to creating a single user. However, it extends the functionality to create batch of users in the same window.

### Before you begin

You must have the role of a systemAdmin or organizationalAdmin to create a user.

### To create a user:

1. On the **Welcome** page > **My Applications**, click  **User Manager**.  
The *User Manager* window opens.
2. From the views list, select from:
  - **Users - Roles**
  - **Users - Tasks**

- Users - Teams
3. To add a user:
- Click  to add a single user.  
The Create User dialog box opens.
  - Click  to add multiple users.  
The Create Multiple Users dialog box opens.
4. Select the type of user you intend to create from **Authentication Type** options. Depending on the type chosen, corresponding fields appear as described in the below table. Other than these, the following fields appear for all types of users:
- **User Name**: Unique name of the user.
  - **User Full Name**: Description or the full name of the user.
  - **User ID**: Operating system identity of the user. This acts as a unique user identifier which is used to log into AppWorks Platform environment. If the **User ID** of an organizational user already exists, you can map it to the existing authenticated user.

If you select	Description	Then
<b>External</b>	This is the default setup where you map the existing domain login ID of the Cordys user.	Provide a <b>User Name</b> , <b>User Full Name</b> , and <b>User ID</b> in the respective fields. <b>Note:</b> If you do not enter a valid User ID, the user is created but is not allowed to access AppWorks Platform. This is because the login ID is not validated during creation and is verified only while accessing AppWorks Platform. Hence, the user with invalid login ID is restricted to log into the AppWorks Platform environment.
<b>Cordys</b>	You can authenticate a user with a customized username and password. To log in as a Cordys user, you must configure the web server. For more	1. Provide a <b>User Name</b> , <b>User Full Name</b> , and <b>User ID</b> in the respective fields. 2. Expand the <b>Assign</b>

If you select	Description	Then
	information on setting the web server, see <a href="#">Web server authentication</a> .	<b>Password</b> groupbox and provide the password in the respective fields. These credentials are validated whenever the user logs into AppWorks Platform environment. For more information on AppWorks Platform password storage mechanism, see <a href="#">AppWorks Platform password hashing mechanism through PBKDF2</a> .

5. If required, expand the **Contact Info** group box and fill the necessary details. See Contact information in the *AppWorks Platform Advanced Development Guide*.
  6. Click .
- A user is created.

**Note:** After creating a user, if you want to edit the details, right-click the user and select **Edit**.

## Cloning users

If you want to create a user with the same set of roles, tasks, teams, and default organization as an existing user, you can clone the user. By cloning, you avoid redoing the process of assigning roles, tasks, and teams.

### Before you begin:

You must have the role of systemAdmin or organizationalAdmin to clone users.

### To clone users:

1. On the **Welcome** page > **My Applications**, click  **User Manager** .  
The User Manager window is displayed.
2. From the view dropdown list, select from:
  - Users - Roles
  - Users - Tasks
  - Users - Teams
3. In the **Users** pane, right-click the user you intend to clone and select **Clone User**.  
The Clone User dialog box is displayed.

4. Provide the following details:
  - **User Name:** Unique name of the user.
  - **User Full Name:** Description or the full name of the user.

**Note:** By default, the roles and the organization of the original user are associated with the new user.
5. Perform the appropriate action:
  - To clone the teams of the user being cloned, select the **Clone Teams**.
  - To clone the tasks of the user being cloned, select **Clone Tasks**.

**Note:** When one of the teams has a lead role assigned, the team is not considered and all the other teams are processed for cloning.
6. In **Assign Password**, type the password for the new user.
7. In **Confirm**, type the password again.
8. Click **Save** icon.  
A cloned user is created.

## Deleting users

Deleting a user revokes the access rights for the user in an organization. You can delete single or multiple users.

### Before you begin:

You must have the role of an Administrator to delete users.

### To delete users:

On the Welcome page, click  User Manager and select the Users - Roles or Users - Tasks view in the User Manager window.

To	Steps
Delete a single user	<p><b>To delete a single user:</b></p> <ol style="list-style-type: none"> <li>1. In the <b>Users</b> pane, right-click the user and select <b>Delete</b>. A confirmation message appears.</li> <li>2. Click <b>Yes</b> to proceed.</li> </ol>
Delete multiple users	<p><b>To delete multiple users:</b></p> <ol style="list-style-type: none"> <li>1. In the <b>Users</b> pane, hold down <b>CTRL</b> and select multiple users, right-click, and then select <b>Delete</b>. A confirmation message appears.</li> <li>2. Click <b>Yes</b> to proceed.</li> </ol>

## Disabling users

Disabling a user deactivates the user temporarily. This prevents the user from accessing applications or features in an organization where the user is disabled.

### Before you begin:

You must have the role of a systemAdmin or Organizational Admin to disable users.

1. On the Welcome page, click  (User Manager).  
The User Manager window opens.
2. Select the **Users - Roles or Users - Tasks** view.
3. In the Users pane, right-click a user and select **Disable**. You can select multiple users by holding down **CTRL**.  
A user is disabled and  appears for the disabled user.

**Note:** To enable a disabled user, right-click the name of the user, and then select **Enable**.

## Assigning roles to users

Roles serve as means to regulate the authorization on tasks. By assigning roles to users, you can define the privileges that can be associated with a user. You can assign roles to users in multiple ways.

### Before you begin:

You must have the role of a systemAdmin or organizationalAdmin to assign roles to users.

On the **Welcome** page > **My Applications**, click  User Manager and select the **Users - Roles** view in the User Manager window.

### To assign a role to a user:

1. In the **Users** pane, select a user.
2. In the **Roles** pane, select a role.
3. Right click the role and click **Assign to Selected Users**.  
The role is assigned to the user.

### Tip:

- Alternatively, to assign the roles, drag the selected roles from the **Roles** pane to the selected users in the **Users** pane.
- Select the **Include internal roles** check box to view and assign the internal roles, such as everyone role.
- You can select multiple roles or users by holding down the **CTRL** key, and press **CTRL+A** to select all the roles.

**Note:** You can also assign roles to users in the Roles - Users view. See Assigning users to roles in the *AppWorks Platform Advanced Development Guide*.

**Caution:** You must assign only the required roles to a user. Addition of roles that are not required causes security leaks.

## Revoking roles from users

After assigning roles to users, to remove roles from users, perform the following procedures.

### Before you begin

You must have the role of systemAdmin or organizationalAdmin to revoke roles from users.

On the **Welcome** page > **My Applications**, click  **User Manager** and select the **Users - Roles** view in the User Manager window.

### To remove roles from a single user:

1. In the **Users** pane, select a user.  
The corresponding roles are highlighted in the **Roles** pane.
2. Select the required roles. Right-click the roles and click **Remove from Selected Users**.  
Unwanted roles are removed from the user.

### To remove roles from multiple users:

1. In the **Users** pane, select users.  
The corresponding common roles are highlighted in the **Roles** pane.
2. Select the required roles. Right-click the roles and click **Remove from Selected Users**.  
Unwanted roles are removed from the selected users.

**Note:** You can also revoke roles from users in the Roles - Users view.

### Tip:

- Select **Include internal roles** to revoke the internal roles, such as everyone role.
- You can select multiple roles or users by holding down the **CTRL** key.

## Assigning tasks to users

Tasks refer to the independent activities performed by a user. By assigning tasks to users, you can prescribe the set of tasks or activities that a user can perform on AppWorks Platform. You can assign tasks to the users in multiple forms in the following manner:

### Before you begin:

- Create a task - See Creating a User Interface Using a Web Page URL in the *AppWorks Platform Advanced Development Guide*.

- Create a user - See [Creating users](#)
- You must have the role of a systemAdmin or organizationalAdmin to assign tasks to users.

On the **Welcome** page > **My Applications**, click  **User Manager** and select [Users - Tasks view](#) in the User Manager window.

**To assign a single task to a single user:**

1. In the **Users** pane, select a user.
2. In the **Tasks** pane, select a task.
3. Right-click the task and click **Assign to Selected Users**.

**To assign multiple tasks to a single user:**

1. In the **Users** pane, select a user.
2. In the **Tasks** pane, select multiple tasks.
3. Right-click the tasks and click **Assign to Selected Users**.  
The required tasks are assigned to the selected users.

**Note:**

- A task might involve multiple subtasks or sub-activities to complete an activity. For such tasks, the **Configure TaskParts** dialog box displays a list of subtasks. Select the subtasks to associate with the user and click **OK**.
- To edit the subtask assignment for a user, right-click the task and select **Configure** to change your selection of the subtask.
- Subtasks are also known as **task parts**.

**Tip:**

- You can assign tasks to users by dragging the selected tasks from the **Tasks** pane to the selected users in the **Users** pane.
- You can select multiple tasks by holding down the **CTRL** key.

**Note:** To remove the assignment of tasks from users, see [Revoking tasks from users](#).

## Revoking tasks from users

After assigning tasks to users, to remove the tasks from users, follow this procedure.

**Before you begin**

You must have the role of systemAdmin or organizationalAdmin to revoke tasks from users.

1. On the **Welcome** page > **My Applications**, click  **User Manager** and select the [Users - Tasks](#) view in the User Manager window.
2. In the **Users** pane, select a user.  
The corresponding tasks are highlighted in the **Tasks** pane.

3. Select the required tasks. Right-click the tasks and click **Remove from Selected Users**.

The tasks are removed from the user.

**Note:**

- Role-based tasks are tasks that are configured to the roles of a user. These tasks are represented as . You cannot revoke such tasks from users.
- User-based tasks are tasks that are directly configured to the user. These tasks are represented as . You can revoke such tasks from users.

**Tip:** You can select multiple tasks or users by holding down the CTRL key.

## Viewing the profile of a user

You can use the User Manager to assign multiple roles and tasks to a user. As an administrator, you must manage all the users and roles in an organization. The User Profile feature provides a summary of the associated objects and enables you to view the roles and tasks associated with a user in a single view.

**Before you begin:**

You must have the role of systemAdmin or organizationalAdmin to view the user profile.

**To view the profile of a user:**

1. On Welcome page, click  **User Manager**.  
The User Manager window opens.
2. Select the **Users - Roles** or **Users - Tasks** view.  
The corresponding view is displayed.
3. Right-click a user and select **User Profile**.  
The User Profile - <UserID> window opens. The roles and tasks associated with the user are displayed in separate sections.

**Note:** In the **Tasks** section, the role to which the task has been assigned is displayed adjacent to the name of the task. A task is assigned to a role and this role is in turn assigned to a user. As a result, the task associated with that role is assigned to the user and is displayed as an associated task of the user.  
The roles and tasks assigned to the user are displayed.

## Changing the default organization of a user

The organization in which you create a user becomes the default organization for that user. See [Creating users](#) You can change this default context and configure another organization as the default organization for the user.

**Before you begin:**

- To configure the default organization for a user, you must have the role of a systemAdmin.
- You must be in the context of the system organization.

**To change the default organization of a user:**

1. In the User Manager window, select **Users - Roles** or **Users - Tasks** view and click . The Manage Default Organization dialog box appears listing the **Users** and **Organizations** panes.
2. Select an organization from the **Select Organization** list. The users available in that organization are displayed.
3. Select a user in the **Users** pane. The organizations with which the user is associated are displayed in the **Organizations** pane. The default organization of the user is highlighted.
4. Select the organization to be configured as default in the **Organizations** pane, right-click it, and then select **Assign to selected User(s)** from the shortcut menu. The selected organization is configured as the default organization for the user.
5. Close the dialog box. The default organization is changed for a user.

## Managing roles

Roles determine the access privileges and activities a user can perform in an organization. This is based on the Access Control List (ACL) settings for a role.

You can manage roles through the following tasks:

- [Creating a role](#)
- [Deleting roles](#)
- [Assigning sub-roles](#)
- [Revoking sub-roles from roles](#)
- [Assigning tasks to roles](#)
- [Revoking tasks from roles](#)

**Note:** OpenText recommends not to use the same role for administration activities and application development. Always create users with application-specific roles. This customization prevents security leaks. For example, do not use the System Admin role for developing or accessing applications as this role is meant for administration activities, such as installing an application.

**Important:** In the earlier versions of AppWorks Platform, role-based menus were available when the user launched Explorer. Simulating this concept of menus, AppWorks Platform offers role-based tasks, where some or all the applications are visible and accessible on the

OpenText AppWorks Platform User Welcome Page depending on the role of a user. For more information about tasks, see Overview of task in the *AppWorks Platform Overview Guide*.

## Creating a role

Creating a role helps to define the privileges that can be associated with a user. This enables the user to use features of AppWorks Platform to perform specific tasks.

### Before you begin:

You must have the role of a systemAdmin or organizationalAdmin to create a role.

### To create a role:

1. On the **Welcome** page > **My Applications**, click  **User Manager**.  
The User Manager window opens.
2. Select **Roles - Roles view** or **Roles - Users view** or **Roles - Tasks view** in the User Manager window and click  .  
The Create Role page opens.
3. In the **Name** field, type a name.
4. In the **Roles** pane, the roles available with the current organization and those in application package are displayed. Select the required roles and move them to the **Selected Roles** pane by clicking .

**Note:** The new role derives the privileges of the selected role. Roles can be removed from the **Selected Roles** pane by clicking .

5. Click **Save**.  
A role is created. After creating a role, if you want to edit the details of the role, right-click the role and select **Edit**.

**Note:** Select **Include internal roles** to assign internal roles, such as everyone role.

## Understanding roles

Roles determine the access privileges and activities a user can perform in an organization and regulate authorization on tasks. Roles are created in an organization or delivered within an application, and then are assigned to users. Users can deploy an application only if they have certain privileges that are offered through roles. The degree to which a user can access the content in an application depends upon the roles assigned to them.

For example, a user can have a customer service representative, supervisor, or manager role. A manager or a supervisor may only have the authority to approve or reject task results.

Roles can be nested, which means a role can extend from another role to enhance the privileges offered by the super role. They are classified as follows:

- Functional roles: High-level roles that can be created according to the needs of the organization. You can create functional roles in User Manager or configure them during

design time and package them. You can use functional roles with organizational units and worklists.

- Application roles: Create application roles during design time and provide access to a set of resources within one package. A package can contain several application roles. You can combine a group of application roles from different packages in a functional role and group the fine grained internal roles into a more logical role. Application roles cannot be used with organizational units and worklists.
- Internal roles: Internal role with specific access privileges that are required for an activity. For example, process administrator is an internal role, which is only used to administer business processes and is a part of the administrator role. There are also some common roles, such as the `everyoneIn` roles. These are visible as **everyoneIn<Application Name>**. The internal roles are not directly visible. To view the internal roles, in the User Manager window, select the **Include Internal Roles** check box.

Functional, application, and internal roles are handled in the same manner from a security perspective.

Depending on the activity, you can assign the following roles:

- Administrator
- Analyst
- Developer
- Deployer
- SetupUser
- SystemAdmin
- DocumentationUser

The following tables list all the roles in AppWorks Platform, their internal roles, and their functions.

Role	Purpose	Internal roles	Function of each internal role
Administrator	Groups the roles that are required to perform administrative activities.	Audit Administrator	Configure audit for components.
		Audit Viewer	View audit data.
		Process Administrator	<ul style="list-style-type: none"><li>■ Monitor and perform operations on business process instances.</li></ul>

<b>Role</b>	<b>Purpose</b>	<b>Internal roles</b>	<b>Function of each internal role</b>
			<ul style="list-style-type: none"> <li>■ View audit information of published processes.</li> <li>■ Archive and restore process instances.</li> </ul>
		CoBOC Administrator	<ul style="list-style-type: none"> <li>■ Configure the synchronization details for database cache.</li> <li>■ Configure synchronization for multiple CoBOC service containers.</li> <li>■ View the CoBOC transaction details in CoBOC Monitor.</li> </ul>
		Log Viewer Administrator	<ul style="list-style-type: none"> <li>■ Search Composite Application logs.</li> <li>■ View log messages.</li> </ul>
		Notification Administrator	<ul style="list-style-type: none"> <li>■ Set notification preferences.</li> <li>■ Update the profile of any user in the organization.</li> </ul>
		Orchestrator Administrator	Extend all the administrative roles of Rule, CoBOC, Schedule, Notification, and Data Transformation. These roles are deprecated and are required for backward compatibility when you migrate from C2.
		Rule Administrator	<ul style="list-style-type: none"> <li>■ Configure Rule Sync up details for synchronizing</li> </ul>

<b>Role</b>	<b>Purpose</b>	<b>Internal roles</b>	<b>Function of each internal role</b>
			<ul style="list-style-type: none"> <li>multiple Rule Repository Processors.</li> <li>■ Monitor rule execution through Admin Cockpit.</li> </ul>
		Schedule Administrator	<ul style="list-style-type: none"> <li>■ Deploy schedules.</li> <li>■ Undeploy schedules.</li> </ul>
		Security Administrator	<ul style="list-style-type: none"> <li>■ Create and manage trust relationship among groups of service containers.</li> <li>■ Manage Trust stores for signing applications.</li> <li>■ Manage certificates in the organization space.</li> <li>■ Manage OTDS resources in the organization space.</li> </ul>
		Security System Administrator	<ul style="list-style-type: none"> <li>■ Manage certificates in the shared space.</li> <li>■ Manage OTDS resources in the shared space.</li> </ul>
		UDDI Administrator	<ul style="list-style-type: none"> <li>■ Manage UDDI registries.</li> <li>■ Validate registries.</li> </ul>
		WS-AppServer Administrator	<ul style="list-style-type: none"> <li>■ Manage access control for Database Metadata.</li> <li>■ Configure auditing settings for Database Metadata.</li> </ul>
		Organizational	Manages the following

<b>Role</b>	<b>Purpose</b>	<b>Internal roles</b>	<b>Function of each internal role</b>
		Administrator	<p>artifacts and system resources within an organization:</p> <ul style="list-style-type: none"> <li>■ Users</li> <li>■ Organizations</li> <li>■ Roles</li> <li>■ ACLs</li> <li>■ Service groups, service containers, and connection points</li> <li>■ Web service interfaces and operations</li> <li>■ XML store objects</li> <li>■ Audit configurations for database tables</li> <li>■ LDAP content</li> <li>■ Gateway statistics monitoring</li> <li>■ Database configurations</li> </ul>
		FTP Administrator	<ul style="list-style-type: none"> <li>■ Create, update, and delete the FTP server configuration.</li> <li>■ Verify the FTP server configuration.</li> </ul>
		BAM Administrator	<ul style="list-style-type: none"> <li>■ Configure synchronization and upload of process data.</li> <li>■ Publish or unpublish Business Event, Process Monitoring Object, Business Measure, KPI, and Dashboard.</li> </ul>
		MDM Administrator	Manage the runtime

<b>Role</b>	<b>Purpose</b>	<b>Internal roles</b>	<b>Function of each internal role</b>
			settings for MDM environment.

<b>Role</b>	<b>Purpose</b>	<b>Internal roles</b>	<b>Function of each internal role</b>
Analyst	Responsible for the analysis and design of business process and the ensuing process improvements.	CWS User	<ul style="list-style-type: none"> <li>■ Modify models within an installed application package in organization staging.</li> <li>■ Add runtime reference to other models.</li> </ul>
		BAM Analyst	<ul style="list-style-type: none"> <li>■ Edit Business Event, Process Monitoring Object, Business Measure, KPI, and Dashboard in staging.</li> <li>■ Publish or unpublish Business Event, Process Monitoring Object, Business Measure, KPI, and Dashboard.</li> <li>■ Delete Business Event, Process Monitoring Object, Business Measure, KPI, and Dashboard.</li> <li>■ View custom and standard Dashboards.</li> </ul>
		MDM Data	Manage the runtime

<b>Role</b>	<b>Purpose</b>	<b>Internal roles</b>	<b>Function of each internal role</b>
		Steward	models (upload data, view dashboard, and logs).

<b>Role</b>	<b>Purpose</b>	<b>Internal roles</b>	<b>Function of each internal role</b>
Developer	Groups all the roles related to the design and development of an application.	Process Developer	<ul style="list-style-type: none"> <li>■ Validate a process model.</li> <li>■ Publish a process model.</li> <li>■ Run a process model.</li> <li>■ Debug a process model during process iterations.</li> <li>■ Generate Web services.</li> </ul>
		CoBOC Developer	<ul style="list-style-type: none"> <li>■ Create objects using CoBOC browser.</li> <li>■ Update objects using CoBOC browser.</li> <li>■ Delete objects using CoBOC browser.</li> </ul>
		CWS Developer	<ul style="list-style-type: none"> <li>■ Develop applications.</li> <li>■ Build and publish applications.</li> </ul>
		Data Transformation Developer	<ul style="list-style-type: none"> <li>■ Execute Data Transformations.</li> <li>■ Generate Web services on Data Transformations.</li> </ul>
		Notification Developer	<ul style="list-style-type: none"> <li>■ Create, update, delete, and customize Inbox and Email models.</li> </ul>

<b>Role</b>	<b>Purpose</b>	<b>Internal roles</b>	<b>Function of each internal role</b>
			<ul style="list-style-type: none"> <li>■ Register Dispatch Logic implementations for use in custom routing of tasks.</li> <li>■ Publish notification models such as work lists, Inbox models, and email models.</li> </ul>
		Orchestrator Developer	Extend all the developer roles of Rule, CoBOC, Schedule, Notification, and Data Transformation. These roles are deprecated and are required for backward compatibility when user migrates from C2.
		Rule Developer	Test the rule execution using the Rule Test Tool.
		Schedule Developer	<ul style="list-style-type: none"> <li>■ Create schedule templates.</li> <li>■ Update schedule templates.</li> <li>■ Delete schedule templates.</li> </ul>
		Translator	<ul style="list-style-type: none"> <li>■ Add, delete, and modify labels in the Translation Repository.</li> <li>■ Add, delete, and modify the supported translation languages.</li> <li>■ Add, delete, and modify translation</li> </ul>

<b>Role</b>	<b>Purpose</b>	<b>Internal roles</b>	<b>Function of each internal role</b>
			<p>contexts</p> <ul style="list-style-type: none"> <li>■ Translate labels in the Translation Repository.</li> <li>■ Approve translations submitted by other users.</li> </ul>
		UDDI Developer	<ul style="list-style-type: none"> <li>■ Work with UDDI Explorer and publish to external registries.</li> <li>■ Manage registries and customize Web service access.</li> </ul>
		WS-Appserver Developer	<ul style="list-style-type: none"> <li>■ Create Database Metadata</li> <li>■ Create WS-AppServer package and data models from the database metadata.</li> <li>■ Generate Java code and Webservices on WS-AppServer models.</li> <li>■ Generate custom business logic.</li> </ul>
		BAM Developer	<ul style="list-style-type: none"> <li>■ Create Business Event, Process Monitoring Object, Business Measure, KPI, and Dashboard.</li> <li>■ Edit Business Event, Process Monitoring Object, Business Measure, KPI, and Dashboard.</li> <li>■ Delete Business</li> </ul>

<b>Role</b>	<b>Purpose</b>	<b>Internal roles</b>	<b>Function of each internal role</b>
			Event, Process Monitoring Object, Business Measure, KPI, and Dashboard.

<b>Role</b>	<b>Purpose</b>	<b>Internal roles</b>	<b>Function of each internal role</b>
Deployer	Responsible for customizing an installed application package.	CWS Application Administrator	<ul style="list-style-type: none"> <li>■ Modify documents within an installed application package in organization staging.</li> <li>■ Publish contents of an application package from organization staging to the runtime.</li> </ul>
Setup User			This role contains all the roles that are available during the installation of AppWorks Platform (baseline and application package) and is automatically assigned to the user who installed AppWorks Platform. You can assign this role to a user to extend the installation-related privileges to that user. This option is useful when you intend to have multiple administrators with the same privileges as that

<b>Role</b>	<b>Purpose</b>	<b>Internal roles</b>	<b>Function of each internal role</b>
			of the installation user. If you upgrade AppWorks Platform or install new applications, the roles within those applications are automatically assigned to this role and not to the user who is performing the activity.
SystemAdmin			This role has authority over the AppWorks Platform setup and is entitled to create and manage users, organizations, roles, and system resources spanning across AppWorks Platform components. Apart from these, the SystemAdmin role is used to configure security-related settings, maintain audit data, and license-related information.
DocumentationUser			This role is used to provide access to documentation only on a need basis. By default, users with the Administrator, Deployer, Analyst, and Developer roles can only view the product documentation. Users without these roles

<b>Role</b>	<b>Purpose</b>	<b>Internal roles</b>	<b>Function of each internal role</b>
			cannot view or access documentation. You must assign the documentationUser role to such users.

## Deleting roles

You can delete the custom roles or organizational roles only. The Application and systemAdmin roles cannot be deleted. You also cannot delete any other roles that are selected with the application and systemAdmin roles.

### Before you begin:

- You must have the role of systemAdmin or organizationalAdmin to delete roles.

### To delete roles:

1. On the Welcome page > click  (User Manager).  
The User Manager window opens.
2. Select the [Roles - Roles view](#) or [Roles - Users view](#) or [Roles - Tasks view](#).

<b>To</b>	<b>Steps</b>
Delete a single role	<b>To delete a single role:</b> <ol style="list-style-type: none"> <li>a. Right-click a role and select <b>Delete</b>. A message appears.</li> <li>b. Click <b>Yes</b> to proceed.</li> </ol>
Delete multiple roles	<b>To delete multiple roles:</b> <ol style="list-style-type: none"> <li>a. Hold down <b>CTRL</b> and select multiple roles, right-click, and then select <b>Delete</b>. A message appears.</li> <li>b. Click <b>OK</b> to proceed.</li> </ol>

## Assigning sub-roles

You can extend the functionality of a role by assigning other roles to it. The roles that are assigned are referred to as sub-roles. By assigning sub-roles, you are deriving and grouping a set of privileges of the existing roles and assigning them to another role. This sub-role assignment is applicable only to the custom or organizational roles. You can assign sub-roles in multiple forms.

**Before you begin:**

You must have the role of systemAdmin or organizationalAdmin to assign sub-roles to a role.

On the **Welcome** page > **My Applications**, click  **User Manager** and select **Roles - Roles view** in the User Manager window.

**To assign single sub-role to a single role:**

1. In the **Roles - <(alphabets)>** pane, select a role.
2. In the **Roles** pane, select a sub-role.
3. Right click the sub-role and click **Assign to Selected Role(s)**.  
The required sub-role is assigned to the selected role.

**To assign a single sub-role to multiple roles:**

1. In the **Roles - <(alphabets)>** pane, select multiple roles by holding down the **CTRL** key.
2. In the **Roles** pane, select a sub-role. Right-click the sub-role and click **Assign to Selected Role(s)**.  
The required sub-role is assigned to multiple roles.

**To assign multiple sub-roles to a single role:**

1. In the **Roles - <(alphabets)>** pane, select a role.
2. In the **Roles** pane, select multiple sub-roles. Right-click the sub-roles and click **Assign to Selected Role(s)**.  
The required sub-roles are assigned to a role.

**To assign multiple sub-roles to multiple roles:**

1. In the **Roles - <(alphabets)>** pane, select multiple roles by holding down the **CTRL** key.
2. In the **Roles** pane, select multiple sub-roles. Right-click the sub-roles, and click **Assign to Selected Role(s)**.  
Multiple sub-roles are assigned to multiple roles.

**Tip:** Alternatively, drag the roles from the **Roles** pane to the selected role(s) in the **Roles - <(alphabets)>** pane.

**Important:**

- A role cannot be a sub-role for itself.
- You cannot assign or revoke sub-roles for the Application and systemAdmin roles.
- You cannot assign or revoke sub-roles for a role if it is selected along with an Application role or systemAdmin role.

**Note:** See [Revoking sub-roles from roles](#) to remove the assignment of sub-roles from roles.

## Revoking sub-roles from roles

To remove sub-roles after assigning them, perform these steps.

### Before you begin

You must have the role of systemAdmin or organizationalAdmin to revoke sub-roles from roles.

### To remove sub-roles from roles:

1. On the **Welcome** page > **My Applications**, click  **User Manager**, and select the **Roles - Roles** view in the User Manager window.
2. In the **Roles - <(alphabets)>** pane, select the roles.  
The corresponding sub-roles are highlighted in the **Roles** pane.
3. Select the required sub-roles. Right-click these sub-roles and click **Remove from Selected Roles**.  
The sub-roles are removed from the selected roles.

**Tip:** You can select multiple roles or sub-roles by holding down the **CTRL** key.

## Assigning tasks to roles

Tasks refer to independent activities. You can assign one or more tasks to a role. By assigning tasks, you can prescribe the set of activities that you can associate with the role. After being assigned to a user, this role governs the activities that the user can perform. Therefore, the user actions are bound to the tasks that are assigned to a role. Alternatively, tasks can be assigned to users directly. See [Assigning tasks to users](#). You can assign tasks to roles in multiple forms.

### Before you begin:

- You must have the role of a systemAdmin or organizationalAdmin.
- Create a task - See Creating a User Interface Using a Web Page URL in the *AppWorks Platform Advanced Development Guide*.

On the **Welcome** page > **My Applications**, click  **User Manager**, and select the **Roles - Tasks** view in the User Manager window.

### To assign tasks to single role:

1. In the **Roles** pane, select a role.
2. In the **Task** pane, select the required tasks.
3. Right-click the tasks and click **Assign to Selected Roles**.  
The required tasks are assigned to the selected role.

### Note:

- A task might involve multiple subtasks or sub-activities to complete an activity. For such tasks, a list of subtasks is displayed in the Configure TaskParts dialog box. Select the subtasks to associate with the role and click **OK**.
- To edit the subtask assignment for a role, right-click the task and select **Configure** to change your selection of subtasks.
- Subtasks are also known as task parts.
- You cannot assign tasks to application roles or the systemAdmin role because these roles cannot be modified.

**Tip:**

- You can assign tasks to roles by dragging the selected tasks in the **Tasks** pane to the selected roles in the **Roles** pane.
- Select the **Include internal roles** option to view internal roles such as everyone role.

**Important:**

In the previous versions of AppWorks Platform, the role-based menus were available when the user launched the Explorer. Simulating this concept of menus, AppWorks Platform offers role-based tasks, where some or all the applications are displayed and are accessible on the Welcome page depending on the role of a user. For more information about tasks, see Overview of tasks in the *AppWorks Platform Overview Guide*.

## Revoking tasks from roles

After assigning tasks to roles, to remove these tasks from the roles, perform the following:

**Before you begin**

You must have the role of a systemAdmin or organizationalAdmin to revoke tasks from roles.

**To revoke tasks from roles:**

1. On the **Welcome** page > **My Applications**, click  **User Manager**, and select the **Roles - Tasks** view in the User Manager window.
2. In the Roles pane, select a role.  
The corresponding tasks are highlighted in the **Tasks** pane.
3. Select the required task. Right-click the task and click **Remove from Selected Roles**.  
The tasks are removed from the role.

**Tip:** You can select multiple roles or tasks by holding down the **CTRL** key.

## Revoking users from roles

Revoking users from roles results in invalidates their privileges. After assigning users to roles, you can remove the unwanted users from roles.

### **Before you begin:**

- You must have the role of systemAdmin or organizationalAdmin.

### **To revoke users from roles:**

- On the Welcome page, click  (User Manager) and then select **Roles - Users view** in the User Manager window.

### **To remove users from a single role:**

1. In the Roles pane, select a role.  
The corresponding users are highlighted in the Users pane.
2. Select the required user(s), right-click them, and then click **Remove from Selected Role(s)**.  
The selected users are removed from the role.

### **To remove users from multiple roles:**

1. In the Roles pane, select the roles.  
The corresponding users are highlighted in the Users pane.
2. Select the required user(s), right-click them, and then click **Remove from Selected Role(s)**.  
The selected users are removed from the selected roles.

**Tip:** You can select multiple roles or users by holding down the CTRL key as you click each role or user.

## ***Roles - Users view***

### **To access the view:**

On the Welcome page, click  (User Manager) and select the Roles - Users view from the Select View list in the User Manager window.

### **Features**

- The left pane displays roles and the right pane displays users.
- You can use the Find feature to find roles or users. Click anywhere on the pane to get the cursor focus and press CTRL+F. The Find dialog box opens at the bottom of the pane. Enter a letter (with which the name or ID starts) and relevant results are highlighted.
- Series of letters are listed as buttons adjacent to the left pane. Click a button and the associated names or IDs (starting with one of the letters on the button) are displayed.
  - For names starting with characters other than letters and numbers, click  ... . For example: ( )
  - Select a role and the number indicating the users associated with it is displayed beside the role name or ID.
  - Select a role and the associated users are highlighted in the Users pane.

- The Users pane indicates the User Name and User ID of the user.
- If you select multiple roles, users who are common to all the selected roles are highlighted.

## **Roles - Roles view**

### **To access the view:**

On the Welcome page, click  (User Manager) and select **Roles - Roles view** from the Select View list in the User Manager window.

### **Features**

- Both panes display functional and application roles. Select **Include Internal Roles** to view the internal roles also.
- You can use the Find feature to find roles. Click anywhere on the pane to get the cursor focus and press CTRL+F. The Find dialog box opens at the bottom of the pane. Enter a letter (with which the Roles start) and relevant results are highlighted.
- Series of letters are listed as buttons adjacent to the left pane. Click a button and the associated names or IDs (starting with one of the letters on the button) are displayed.
- For names starting with characters other than letters and numbers, click . For example: ( )
- When you select a role, a number indicating the sub-roles associated with it is displayed beside the role name or ID.
- Select a role in the Roles - <(alphabets)> pane (Left pane) and the associated sub-roles are highlighted in the Roles pane on the right.
- If you select multiple roles, sub-roles that are common to all the selected roles are highlighted.

## **Roles - Tasks view**

### **To access the view:**

On the Welcome page, click  (User Manager) and select the Roles - Tasks view from the Select View list in the User Manager window.

### **Features**

- The left pane displays roles and the right pane displays tasks.
- You can use the Find feature to find roles or tasks. Click anywhere on the pane to get the cursor focus and press CTRL+F. The Find dialog box opens at the bottom of the pane. Enter a letter (with which the name or ID starts) and relevant results are highlighted.
- Series of letters are listed as buttons adjacent to the left pane. Click a button and the associated names or IDs (starting with one of the letters on the button) are displayed.
- For names starting with characters other than letters and numbers, click . For example: ( )

- Select a role and the number indicating the tasks associated with it is displayed beside the role name or ID.
- Select a role and the associated tasks are highlighted in the Tasks pane.
- If you select multiple roles, tasks that are common to all the selected roles are highlighted.

# Chapter 5

# Managing system resources

AppWorks Platform establishes backend connections to execute SOAP requests. System resources, such as service groups and service containers, play a vital role in fulfilling a SOAP request. They communicate to and from applications and databases to fetch data and execute tasks.

The following system resources are important in the SOAP request cycle.

## **Service groups**

A service group is a logical entity to which SOAP messages can be sent. It can also receive SOAP messages. When AppWorks Platform is installed, it creates several service groups and corresponding service containers.

When a SOAP request is sent to AppWorks Platform, it searches in LDAP for a service group that can execute this request. The decision about whether a service group can execute a request is based on the namespace and the Web service operation that is requested. If the request cannot be made, the request is transferred to another service group. The back and forth movement of SOAP requests between the service groups is monitored by the AppWorks Platform Web gateway. For the service group to execute the request, you must attach the appropriate Web service interfaces.

**Note:** A Web service operation usually contains a query or another kind of task that determines which data is being requested or which action must be taken. A group of Web service operations form a Web service interface.

## **Service containers**

Service containers are the Java virtual machines (JVMs) that run on the operating system. You configure one or more service containers for a service group. By attaching multiple service containers to a service group, you can distribute the number of requests across multiple service containers to achieve load balancing.

## **OS process**

Multiple service containers can be combined in one OS process to reduce memory usage and optimize performance. The request is sent from one service container to another in the same OS process.

## Application connectors

A service container includes one or more application connectors and connection points. An application connector is responsible for translating a SOAP Request (XML) to a format that an application or API understands. An application connector also translates the response from the application or API back into a SOAP message (XML). The application connector is attached to a service container and loads when the service container starts.

### Connection point

A connection point is a generic label of a middleware address in AppWorks Platform. It contains administrative information about the middleware, such as the Uniform Resource Identifier (URI) of the socket. Connection points are available for TCP/IP. Service containers have at least one connection point but can be equipped with multiple connection points. Depending on how the SOAP request involves multiple connection points, connection points are grouped into a connection point group.

When a service container starts, it initiates an instance of the JVM and loads the service container. In turn, the service container loads the application connector and opens the connection point for communication.

This section contains the following topics:

- [Managing service groups](#)
- [Managing service containers](#)
- [Managing connection points](#)
- [Managing client connection point groups](#)
- [Working with Web Service Interface Explorer](#)
- [Managing database configurations](#)
- [Monitoring Web gateway](#)
- [Transaction configuration](#)

## Managing service groups

A service group is a logical entity that contains a set of related service containers. AppWorks Platform enables administrators to manage service groups.

Monitor Service Group involves activating service containers and restoring crashed service containers. It also helps when installing and uninstalling applications.

For more information about creating and modifying service groups, see:

- [Creating a service group](#)
- [Modifying service group configuration](#)
- [Modifying a monitor service group](#)
- [Mapping service groups and web service interfaces](#)

## Creating a service group

This topic describes the procedure to create a service group.

### Before you begin:

- You must have the role of systemAdmin or organizationalAdmin to create a service group in an organization.
- A service group is a logical grouping of service containers. A service group helps send, reply, relay, and process SOAP requests. To process a SOAP request for your business operations over backend applications such as databases, you must setup connectivity using a service group configuration.

### To create a service group:

1. On the **Welcome** page > **My Applications**, click  (System Resource Manager). The System Resource Manager window opens.
2. In the toolbar, click  **Show All Service groups**. The Service Groups App Palette appears.
3. Click  to display the New Service Group wizard.
4. Select one or more application connectors.

**Note:** To select multiple application connectors, press **CTRL** and click the connectors.

5. Click **Next**.  
The Provide Service Group Details page opens.

**Note:** By selecting one or more application connectors, the service container processes all the requests of those connectors. For more information, see Application Connectors in the *AppWorks Platform Advanced Development Guide*.

6. Specify the details about the service group.
7. Click **Next**.  
The Provide Service Container Details page appears. For more information, see Service Group Configuration Interface in the *AppWorks Platform Advanced Development Guide*.
8. Specify the required details.
9. Click **Next**.  
The Provide Details page opens. For more information, see Service Container Configuration Interface in the *AppWorks Platform Advanced Development Guide*.
10. Specify the required details for the application connectors and click **Next**.  
The Provide Connection Point Details page opens. For more information, see [Configuration parameters for application connectors](#).
11. Specify the required details and click **Finish**.  
For more information on configuring the connection point, see Connection Point Configuration Interface in the *AppWorks Platform Advanced Development Guide*.

**Note:** To delete a service group, select the service group in **Service Groups App Palette** > **Context menu** > click  (Delete).  
The service group is deleted.

## Modifying service group configuration

When modifying a service group, administrators can change its configuration details and attach or detach web service interfaces. See [Attaching web service interfaces to service groups](#).

### Before you begin

You must have the role of systemAdmin or organizationalAdmin to modify a service group in an organization.

### To modify a service group configuration:

1. On the **Welcome** page > **My Applications**, click  (System Resource Manager).
2. In the toolbar, click  **Show All Service Groups**.  
The Service Groups App Palette opens.
3. Select the Service Group to modify and double-click it.  
The service group's properties are displayed in its App Palette at the bottom of the page.  
For more information, see Service Group Configuration Interface in the *AppWorks Platform Advanced Development Guide*.
4. Modify the service group configuration, and click .  
The selected service group is modified.

**Note:** For more information about attaching web service interfaces to service groups, see [Attaching web service interfaces to service groups](#).

## Attaching web service interfaces to service groups

A logical group of web service operations form a Web service interface. When a web service interface is attached to a service group, the service containers beneath the service group fulfill the SOAP request, by running the operations associated with the web service interface.

You can attach the web service interfaces either during the creation of service groups or later, using the **ServiceGroup Properties - <Service Group Name>** App Palette.

### Before you begin:

You must have the role of systemAdmin or organizationalAdmin to attach web service interfaces to a service group.

### To attach a web interface to a service group:

1. Click  in the Web Service Interfaces group box.  
The Attach Web Service Interfaces dialog box displays a list of existing web service interfaces.

2. Expand the **Organization/Package group box**, and select an Application Package or Organization name from the list.  
The Web service interfaces are listed.
3. Select the web service interface check box and click **Add**.

**Note:** In the list of web service interfaces, the  icon indicates that the corresponding web service interface cannot be implemented by any of the application connectors associated with the service group. To resolve it, either clear the selection of the web service interface or add the corresponding application connector that implements the web service interface. The web service interfaces are added to the existing list.

4. Click **Done**.  
The web service interfaces are displayed.

**Note:** If you are attaching web service interfaces in the Provide Service Group Details page of the Service Group creation wizard, select the check boxes to attach them to the service group. If you are attaching in the ServiceGroup Properties - <Service Group Name> App Palette, you do not have to select the check boxes. The selected web service interfaces are attached to the service group.

**Tip:** To detach a Web service interface, select its check box and click .

## Modifying a monitor service group

### Before you begin

You must have the role of systemAdmin or organizationalAdmin to modify a Monitor Service Group.

### To modify a monitor service group:

1. On the **Welcome** page > My Applications, click  (System Resource Manager).  
The System Resource Manager opens.
2. In the toolbar, click  **Show All Service Groups**.  
The Service Groups App Palette opens.
3. Select **Monitor Service Group** and double-click.  
The corresponding properties are displayed in the ServiceGroup Properties - <Service Group Name> App Palette. For more information, see Service group configuration interface in the *AppWorks Platform Advanced Development Guide*.

**Note:** Monitor Service Container is displayed in the Service Containers App Palette. To modify, double-click and change the configuration details in the Monitor tab of the Services Properties - <Service Name> App Palette. For more information, see [Monitor service group configuration interface](#).

4. Modify the Monitor Service Group configuration details, and click .  
The Monitor Service Group details are modified.

## Enabling payload validation

If you disabled payload validation for deploying an application using Application Deployer, you must enable payload validation.

### To enable payload validation using Management Console:

1. Set the following properties to true in the `wcp.properties` file using Management Console:
  - `cordys.gateway.protocol.validation`
  - `cordys.gateway.request.validation`
  - `cordys.gateway.response.validation`
2. Restart the Web server.

### To enable payload validation through a SOAP request:

1. On the Welcome page of AppWorks Platform, click **System Resource Manager**.  
The System Resource Manager window opens.
2. Click **Show > All Service Groups**.  
All the service groups are displayed.
3. Double-click the required service group for which payload validation is disabled.  
The service group details are displayed.
4. In Service Group Properties, select the **Protocol** and **Payload** check boxes, and then click **Save**.

## Disabling payload validation

Disable payload validation if it is enabled.

### To disable payload validation using Management Console:

1. Set the following properties to false in the `wcp.properties` file using Management Console:
  - `cordys.gateway.protocol.validation`
  - `cordys.gateway.request.validation`
  - `cordys.gateway.response.validation`
2. Restart the Web server.

### To disable payload validation through a SOAP request:

1. On the Welcome page of AppWorks Platform, click **System Resource Manager**.  
The System Resource Manager window opens.
2. Click **Show > All Service Groups**.  
All the service groups are displayed.

3. Double-click the required service group for which payload validation is enabled.  
The service group details are displayed.
4. In Service Group Properties, clear the **Protocol** and **Payload** options, and then click **Save**.

## Mapping service groups and web service interfaces

This topic describes the mapping of various service groups (that can be created for the available Application Connectors) with the Web service interfaces that must be attached to these service groups during configuration. Web service interfaces are an aggregation of Web service operations that usually contain a query or a task that determines which data is being requested or which action must be taken, and are associated with service groups. When a Web service interface is attached to a service group, the service containers beneath the service group fulfill the SOAP request, by executing the operations associated with the Web service interface.

This table displays the default service groups and basic Web service interfaces that are required by the corresponding service groups to run. They must be mapped as is.

For a list of public Web services, see SOAP APIs in the *AppWorks Platform API Reference Guide*.

Service Group	Application	Web service interfaces
Auditing	Cordys Audit Service	Method Set ArtifactsAudit 1.0
LDAP	Cordys LDAP Connector	Method Set LDAP 1.0
	Cordys LDAP Connector	Method Set LDAP 1.1
	Cordys LDAP Connector	Method Set VirtualConsortium
	Cordys LDAP Connector	Method Set Certificate User
Event Handling	Cordys Event Service	Method Set Event Service 1.0
FTP Service	Cordys FTP Connector	Method Set FTP 1.1
	Cordys FTP Connector	Method Set FTP
Monitor	Cordys ESBServer	Method Set Monitor
	AppWorks Platform DB Connection	Method Set DSOConfig
E-mail	Cordys E-mail Connector	Method Set E-mail
ISV Packages	Cordys ESBServer	Method Set ISV Packages
Click Choice	AppWorks Platform WCP 1.3	Method Set ClickChoice
	AppWorks Platform WCP 1.4	Method Set ClickChoice

<b>Service Group</b>	<b>Application</b>	<b>Web service interfaces</b>
Collaborative Workspace	Cordys CWS Core	Method Set CWS Repository 1.0
	Cordys CWS Core	Method Set CWS Build 1.0
	Cordys CWS Core	Method Set CWS Deploy 1.0
	Cordys CWS Core	Method Set CWS Package 1.0
	Cordys CWS Core	Method Set CWS Operation 1.0
	Cordys CWS Core	Method Set CWS Synchronize 1.0
	Cordys CWS Core	Method Set CWS Team Development 1.0
	Cordys CWS Core	Method Set CWS Upload 1.0
UDDI	Cordys UDDI Connector	Method Set UDDI
Data Transformation	Cordys Data Transformation	Method Set DataTransformation 4. 2
		Parser Method Set
MDM Hub Publisher	Cordys MDM	MDM Common Methodset
		MDM Publisher Methodset
MDM Service	Cordys MDM	MDM Common Methodset
		MDM Service Methodset
		MDM LogViewer Methodset
Notification	Cordys Notification	Notification Method Set
Rule Engine	Cordys Rule Engine	Method Set RuleAdmin 4.2
		Method Set RuleEngine 4.2
Repository	Cordys Tag Server	Method Set TagServer 1.0
	Cordys XMLStore	Method Set XMLStore
		Method Set MenuStore
		Method Set StyleStore
	Cordys Task Server	Method Set Task Design Time
		Method Set Task Run Time
	Cordys Document Store	Method Set Document Store
Security Administration	Cordys Security Admin Core	Method Set SecAdmin
		Method Set Certificate Manager
		Method Set Verification
Single Sign-On	Cordys Single Sign-On	SAMLProtocol
Scheduler	Cordys Scheduler	Method Set Scheduler 4.2
		Method Set Scheduler 1.0
CoBOC	Cordys CoBOC	COBOC Method Set 4.2
		COBOC Method Set
		Admin Method Set

<b>Service Group</b>	<b>Application</b>	<b>Web service interfaces</b>
XForms	Cordys XForms Runtime	Method Set XForm
Translation	AppWorks Platform Translation	Method Set Translation 1.0
Business Process Management	Cordys Business Process Engine	Method Set Admin and Monitoring 4.2
		Method Set Process Instance Archive 4.2
		Method Set Process Deployment 4.2
		Method Set Process Modeling 4.2
		Method Set Process Execution 4.2
		Method Set Process Authorization 4.2
		Method Set Process Schemas 4.2
		Method Set Query Process Instance Data 4.2
		Method Set Business Identifier 4.2
WS-AppServer	Cordys WS-AppServer	WS-AppServer Development Method Set
		WS-AppServer Runtime Method Set
		Method Set DBMetadata 1.0
		Method Set DB Schema
		Method Set JDBCSCHEMA
		Method Set OleDBSchema 1.0
		Method Set OleDBSchema 1.1
		Method Set Audit 1.0
		Method Set DataSourceInfo
		Method Set DBCommands 1.0
		Method Set DBConnector Schemas 1.0
		Method Set Log Viewer
User Management	OpenText AppWorks Platform User Management	Method Set User Management Global
		Method Set User Management Global
		Method Set User Management User

## Managing service containers

A service container is an instance of a Java Virtual Machine (JVM) that has application connectors and connection points attached to it. Several service containers can run within a single JVM. A service container loads the application connectors and opens the connection points from which to read.

A service container processes the request or response SOAP messages according to the conventions defined by SOAP. Service containers are responsible for enforcing the rules that govern the exchange of SOAP messages. They access the service containers provided by the underlying protocols through SOAP bindings.

Users with SystemAdmin role can monitor, stop, and start the service containers in their organizations.

A service container can be started in the following ways:

- **Automatic:** The service container starts automatically after the AppWorks Platform (<instance name>) service container starts.
- **Manual:** The service container can be started manually by using the System Resource Manager.

**Note:** Multiple service containers can run on a single machine. By default, System Resource Manager displays the available service containers in a grid view. Click on the toolbar of the service containers App Palette to change the mode in which the service containers are displayed.

**Note:** On VMware or low processing speed machines, CoBOC, Business Process Management, and Rule Management service containers might consume high CPU capacity during startup. To reduce CPU usage, set the `com.cordys.cluster.timers.factor` property to 2 in `wcp.properties`.

**Note:** For more information about service containers, see:

- [Creating a service container](#)
- [Modifying a service container](#)
- [Configuring OS processes for a service container](#)
- [Monitoring a service container](#)
- [Viewing memory status details](#)

## Understanding service containers

This topic provides a short description of each service container used in AppWorks Platform.

Name of the service container in AppWorks Platform	Function
Auditing	<ul style="list-style-type: none"><li>■ View the Artifacts Audit Information.</li><li>■ Record the Artifacts Audit Information when the mode is selected as SOAP in the Audit Configuration page.</li></ul>
Business Process Management	<p>Manage the following:</p> <ul style="list-style-type: none"><li>■ Features of business process model debugging, execution, and deployment</li></ul>

<b>Name of the service container in AppWorks Platform</b>	<b>Function</b>
	<ul style="list-style-type: none"> <li>■ Features of PIM and Archival</li> </ul>
CoBOC	<ul style="list-style-type: none"> <li>■ Set security restrictions.</li> <li>■ Manage CoBOC Browser and Admin Cockpit.</li> </ul>
Repository	<p>Acts as a repository for AppWorks Platform objects. Contains the following service containers:</p> <ul style="list-style-type: none"> <li>■ Tag Server: Manage tags.</li> <li>■ Task Server: Retrieve and store tasks into XDS.</li> <li>■ XML Store: Read XML objects from a persistent storage and write into it.</li> </ul>
Collaborative Workspace	<ul style="list-style-type: none"> <li>■ Create workspace, project, solution, and documents.</li> <li>■ Synchronize content in workspace with content in the file system of the local computer.</li> <li>■ Build and publish documents and projects.</li> <li>■ Version control by integrating with a source control management system.</li> <li>■ Package application content.</li> </ul>
Data Transformation	Manage data transformation.
Document Store	Managing Content Management Servers (CMS) in AppWorks Platform.
Email	<ul style="list-style-type: none"> <li>■ Send and receive emails.</li> <li>■ Delete received emails.</li> </ul>
Event Handling	Publish all events that occur in AppWorks Platform to the event's subscribers.
FTP	<ul style="list-style-type: none"> <li>■ Upload files to an FTP server.</li> <li>■ Download files from an FTP server.</li> <li>■ Delete files from an FTP server.</li> <li>■ Get the list of files and folders in the FTP server.</li> </ul>

Name of the service container in AppWorks Platform	Function
LDAP	<ul style="list-style-type: none"> <li>■ Provide access to an LDAP directory.</li> <li>■ Query and modify directory services running over TCP/IP.</li> </ul>
Logging	Log and retrieve log messages.
monitor@<machine name>	<p>Provide the following service containers to manage the lifecycle of service containers (start, stop, restart, and reset):</p> <ul style="list-style-type: none"> <li>■ License</li> <li>■ Application Package</li> <li>■ Monitor</li> <li>■ Database Configuration: Manage Database Configurations</li> </ul>
Notification	<p>Manage:</p> <ul style="list-style-type: none"> <li>■ Dispatch logic registration</li> <li>■ Participant preferences</li> </ul>
Rule Management	<ul style="list-style-type: none"> <li>■ Create rules.</li> <li>■ Update rules.</li> <li>■ Delete rules.</li> <li>■ Manage the Rule Test Tool.</li> </ul>
Scheduling	<p>Manage:</p> <ul style="list-style-type: none"> <li>■ Deploy and remove schedules</li> <li>■ Security</li> </ul>
Security Administration	Manage the security settings (mainly the trust stores) for the AppWorks Platform environment.
Single Sign-On	Provide Single Sign-On services based on the SAML protocol.
UDDI Service	<ul style="list-style-type: none"> <li>■ Create a registry.</li> <li>■ Publish methods and edit them.</li> <li>■ Search for a service using the UDDI Browser.</li> </ul>
WS-AppServer	<ul style="list-style-type: none"> <li>■ Create database metadata on all leading</li> </ul>

<b>Name of the service container in AppWorks Platform</b>	<b>Function</b>
	<p>database vendors and build data models.</p> <ul style="list-style-type: none"> <li>■ Support generation of Java code for both standard and custom data models.</li> <li>■ Support generation of Web service operations for data models and Java sources.</li> <li>■ Supports running SQL queries on the data models.</li> </ul>
XForms	<ul style="list-style-type: none"> <li>■ Handle all requests and responses related to XForms content.</li> <li>■ Retrieve Property sheets, create preview, and publish (design time).</li> <li>■ Retrieve Form content, cache, and compress.</li> <li>■ Configure expiry settings (runtime).</li> </ul>
MDM Hub Publisher	<ul style="list-style-type: none"> <li>■ Publish the changes at the Hub to all the subscribing spoke systems.</li> <li>■ Handle the Publish To Run-time process (PTR) of MDM Models.</li> </ul>
MDM Service	<ul style="list-style-type: none"> <li>■ Receive the changes from all the spoke systems and update the Hub data store.</li> <li>■ Perform bulk uploads.</li> <li>■ Handle requests for log viewer to view the upload and the synchronization logs.</li> </ul>
Business Activity Monitor	<ul style="list-style-type: none"> <li>■ Manage BAM configurations such as business process and data filtering.</li> <li>■ Publish and identify the KPIs and perform trend analysis.</li> <li>■ Publish and fill monitoring objects that form the building block for monitoring business processes.</li> <li>■ Publish and run the business measures that help in measuring any aspect of a business process or Enterprise Data Object or external Web services.</li> <li>■ Publish and run the business events that</li> </ul>

Name of the service container in AppWorks Platform	Function
	<p>monitor business critical events in that particular business process.</p> <ul style="list-style-type: none"> <li>■ Support composite controls that are the medium for visual representation of BAM runtime data.</li> </ul>

## Creating a service container

There are two ways to create a service container:

- Creating a new service group
- Attaching a service container to an existing service group

This procedure describes how to add a service container to an existing service group. To create a service container when creating a new service group, follow the instructions in [Creating a service group](#).

**Note:** OpenText recommends creating similar types of service containers under a service group. For example, service container A and service container B can be considered as similar, if they configure the same application connector.

### Before you begin:

You must have the role of systemAdmin or organizationalAdmin to create a service container in an organization.

### To create a service container:

1. On the **Welcome** page > **My Applications**, click  (System Resource Manager). The System Resource Manager window lists the available service containers in the Service Containers App Palette.
2. In the toolbar, click > **Show > Show All Service Groups**. The Service Groups apps palette is displayed on the left.
3. To add a service container to an existing service group , do the following:
  - a. Select a service group and right-click it.
  - b. Select **New Service** to display the New Service Container wizard.
  - c. Select an application connector from **Application Connectors**, and click **Next** to display the Provide Service Container Details page.

**Note:** A service container created at system level can process requests from other organizations (if the service container does not exist in those organizations). There is no need to replicate the same service containers for different organizations.

**Important:** If you add a WS-AppServer service container to an existing WS-AppServer service group, ensure that it points to the same database as the default service

container. If service containers point to different databases, they might cause problems with Web service generation.

4. Specify the required service container details and click **Next**.

Provide Details for the <selected application connector> appears. For example, if you selected WS-AppServer connector, the Provide Details for the Ws-Appserver Connector page appears. For more information, see Service container configuration interface in the *AppWorks Platform Advanced Development Guide*.

5. Specify the details for the application connector and click **Next**.

The Provide Connection Point Details page appears. For more information, see [Configuration parameters for application connectors](#).

6. Specify the connection point details and click **Finish**.

The service container is created. For more information, see Connection Point Configuration Interface in the *AppWorks Platform Advanced Development Guide*.

## **Configuration parameters for application connectors**

The following topics contain the configuration options for application connectors:

- Auditing - See Auditing service connection parameters interface in the *AppWorks Platform Advanced Development Guide*.
- Business Process Management - See Business process management service properties interface in the *AppWorks Platform Advanced Development Guide*.
- CoBOC Service - See [CoBOC Services Configuration interface](#).
- Collaborative Workspace - See Collaborative workspace service connection interface in the *AppWorks Platform Advanced Development Guide*.
- Data Transformation Service - See Data transformation service interface in the *AppWorks Platform API Reference Guide*.
- E-Mail - See [E-mail connector interface](#).
- FTP - See [FTP Service Connection Parameters interface](#).
- LDAP - See [LDAP service connection parameters interface](#).
- MDM Publisher - See [MDM publisher connector configuration interface](#).
- MDM Service - See [MDM service connector configuration interface](#).
- Notification Service - See [Notification Service configuration interface](#).
- Rule Repository Service - See [Rule Repository service interface](#).
- Scheduler Service - See Scheduler service interface in the *AppWorks Platform Advanced Development Guide*.
- Security Administration - See [Security Administration Configuration interface](#).
- Single Sign-On - See Single sign-on connection parameters interface in the *AppWorks Platform Advanced Development Guide*.
- Repository Service - See [Repository service connection parameters interface](#).  
Contains the following services:

- Tag Service
  - Task Service
  - XML Store
  - Document Store
- UDDI Service - See [UDDI connector interface](#).
- WS-AppServer - See [WS-AppServer service connection parameters interface](#).
- XForms - See [Configuring the XForms service container](#).
- HTTP - See [Configuring HTTP connector](#).
- JMS - See [Configuring the JMS connector](#).
- Other (Custom Application Connectors) - See Custom connector configuration interface in the *AppWorks Platform Advanced Development Guide*.

The following application connectors do not need to be configured. They are attached to the corresponding service containers by default.

Application connector	Service container
Database Configuration	See <a href="#">Monitor service group configuration interface</a> .
Event Service	Attached to the Event Handling service container. There are no parameters specific to event handling.
Foundation Framework	See <a href="#">Configuring the XForms service container</a> .
Web Logger	See <a href="#">Logging configuration interface</a> .

### Notification Service configuration interface

The Notification Service enables communication of event-related information to appropriate users or roles. The Provide Details for Notification Service Connector section of the New Service Group wizard helps you configure a notification service.

#### Notification Engine Database tab

The following table describes the fields of the Notification Engine Database tab.

Field	Description	User action
Select Database Configuration	Create a new database configuration	Select <b>New Database Configuration</b> from the drop-down list and provide the required information in the dialog box that opens. See <a href="#">Creating a database configuration</a> .
	Continue with an	Select the database configuration from the drop-

Field	Description	User action
	existing database configuration	down list. The associated fields are automatically filled.
Advanced Options		<p>Expand the group box and provide the necessary details. See Advanced properties in the AppWorks Platform Advanced Development Guide.</p> <p><b>Note:</b> Setting the cursor cache size to a high value results in high memory consumption. Based on the application usage, the administrator should set an optimal value. This is also applicable to the Query Cache size. For example. the optimal cursor cache size can be between 50 - 100, depending on to the application usage.</p>

## Notification Engine tab

The following table describes the fields of the Notification Engine tab.

Field	Description
Number of Threads	The number of messages that can be triggered simultaneously. The default value is five. If you increase the number of threads, the number of messages that can be triggered simultaneously also increases, but at the expense of memory.
Default Dispatch Timeout (in seconds)	If the default dispatch logic is used for delivering mail, specify the timeout period (in seconds) for that dispatch logic. The default value is zero.
Email Connector Service	To specify a mail connector, click  , select the required mail connector from the Choose E-mail Service dialog box, and click <b>OK</b> .
URL Prefix	If you configured the Notification service container to use the E-mail service, specify the URL prefix in the URL Prefix text box. The URL prefix is used to send mail. The syntax for URL Prefix is <code>http:// hostname: portnumber</code> .
Default Email ID	The default email address. If you do not provide a sender's email address while sending a notification, the system checks for the email address in the participant preferences. If the email address is not supplied in the participant preferences page, the system uses the default email ID that you provide in the service container configuration page.
Default Mail Content	The default message to send with the notification.
Generic Notification	

Field	Description
Service Container Properties	
Start Task Mandatorily before Complete	Select this check box to require the user to start the task before completing it. By default, this check box is cleared.

## Inbox Preferences tab

The following table describes the fields on the Inbox Preferences tab.

Field	Description
Inbox View Mode	<p>Specify how to check for new messages.</p> <ul style="list-style-type: none"> <li>■ To check for new messages (in the Inbox), click <b>Refresh</b>. This option is selected by default.</li> <li>■ To be alerted automatically when you receive a new message, click <b>Push</b>, select the event processor from the Choose Target window, and then click <b>OK</b>.</li> </ul>
Inbox Worklist Preferences	<p>Specify the work lists to show in the left tab of the Inbox. You can also specify whether to automatically claim a task when a user opens it.</p> <p><b>Automatically Claim / Revoke Claim in Inbox</b> - Enables a user to claim the task as soon as it opens. If the user opens and claims the task without any action, the task reverts to the New state automatically. By default, this check box is selected.</p> <p><b>Show Empty Work Lists</b> - Displays the work lists that do not contain tasks. By default, this check box is selected.</p> <p><b>Show Notifications</b> - Displays the Notifications option in My Inbox. By default, this check box is selected.</p> <p><b>Show Memo As Tooltip</b> - Enables showing the memo associated with a task as tooltip. Select the required task in the Inbox and position the pointing device on the Activity column to view the memo. The tooltip contains the name of the user who added the memo, the time when the memo was added, and the memo content. You can customize this option from Inbox Preferences. See Inbox preferences in the AppWorks Platform Advanced Development Guide. By default, this check box is cleared. Selecting the check box provides this feature for all users.</p> <p><b>Note:</b> In Refresh mode, the tooltip information that is associated with the selected task is cached. The cache is updated only when the Inbox is refreshed.</p>

<b>Field</b>	<b>Description</b>
Case Inbox Preferences	<p>Specify the types of case models to display in the Case(s) tab of My Inbox</p> <p><b>Hide Sub-Case Models in Case Inbox</b> - Select this check box to hide the sub-case models in the Case tab of My Inbox. By default, this check box is cleared and all the sub-case models belonging to a case are displayed in the Inbox.</p>
Task Toolbar Preferences	<p>Specify the task operations that can be shown in the task tool bar displayed in the following toolbars:</p> <ul style="list-style-type: none"> <li>■ Inbox Grid - Shows the tool bar button on the tool bar displayed above My Inbox Grid. This option also determines whether to show or hide this option as a context menu when you click any row in My Inbox grid.</li> <li>■ Task View - Shows the tool bar button on the tool bar displayed in the task view.</li> </ul> <p>Using this configuration, you can determine which of the following buttons to show on the tool bar of My Inbox grid and Task View.</p> <ul style="list-style-type: none"> <li>■ Claim</li> <li>■ Assign</li> <li>■ Revoke</li> <li>■ Start</li> <li>■ Stop</li> <li>■ Pause</li> <li>■ Resume</li> <li>■ Complete</li> <li>■ Suspend</li> <li>■ Skip</li> <li>■ Delegate</li> <li>■ Forward</li> <li>■ Modify System Attributes</li> <li>■ Add Memo</li> <li>■ Add Attachments</li> <li>■ Add Reminder</li> <li>■ Add Tags</li> </ul> <p><b>Default options</b></p> <p>By default, all the above buttons except Complete are shown in the My Inbox grid View.</p>

Field	Description
	<p>By default, all the above buttons are shown in Task View. The Add Tags button is not available in the tool bar of the task view so you cannot configure this button to be shown in Task view.</p> <p><b>Caution:</b></p> <ul style="list-style-type: none"> <li>■ Be sure that the Notification service container is restarted when its properties are modified. The new settings are applicable when users reopen My Inbox.</li> <li>■ Some configurations may not enable users to work on the tasks either from the My Inbox Grid or from Task view. The administrator should avoid such configurations.</li> <li>■ When more than one service of notification is created in a service group, the administrator should replicate the same configuration in all the services.</li> </ul>

### Repository service connection parameters interface

The Repository service container acts as a repository for AppWorks Platform objects. This service container contains the following:

- The Tag service manages tags.
- The Task service retrieves and stores tasks to the Repository
- The XML Store reads and writes XML objects to a persistent storage
- The Document Store provides a facility to work with any content repository

The following table describes the fields on the XML Store, Task Service, and Tag Service tabs of the Properties - AppWorks Platform Repository service container App Palette in the System Resources Manager window. For information on the fields that appear on the Document Store tab, see [Document Store Configuration interface](#).

Field	Description
Document Cache Size	The number of documents kept in cache per Repository service.
Network Configuration	<p>To select a free port number to enable cache coherence manually, you can select the Manual check box and provide an unused free port number in the Port Number field below. By default, the Manual field is not selected and the framework automatically assigns an unused free port number.</p> <p><b>Note:</b> You must restart the service container to apply the change.</p>
Select Database Configuration	To create a database configuration, select the <b>New Database Configuration</b> option from the drop-down list and provide the required information in the dialog box that opens. See <a href="#">Creating a</a>

Field	Description
	<p><a href="#">database configuration</a>.</p> <p>To continue with an existing database configuration, select it from the drop-down list. The associated fields are automatically filled.</p>
Advanced Options	Expand the group box and provide the necessary details. See Advanced properties in the AppWorks Platform Advanced Development Guide.

### Notification Preferences interface

The following table describes the fields on the Notification Preferences interface.

Field	Description
Email	Specify the email ID to receive the tasks or notifications.
Send task or notification to this e-mail	If selected, sends emails to the respective e-mail ID.

### FTP Service Connection Parameters interface

To perform file transfer operations through a proxy server instead of a local host, enter the following details.

Field	Description
Use Proxy	Enables routing of FTP requests through a proxy server.
Proxy Type	Type of proxy protocol: HTTP, SOCKS4, and SOCKS5.
Proxy Host	Name of the proxy server.
Proxy Port	Port of the proxy server.
Proxy Username and Proxy Password	Authentication details required to access the proxy server.

**Note:** In AppWorks Platform, proxy server support is available only for SFTP operations.

### LDAP service connection parameters interface

There are no properties to set on the LDAP Connector interface. To configure the LDAP service, open Management Console > Platform Properties tab, and then edit wcp.properties. See [LDAP related properties](#).

### MDM publisher connector configuration interface

The MDM Publisher application connector is available while creating a service container. For information see [Creating a service container](#).

The following table describes the fields on the Provide Details for MDM Publisher Connector Interface:

<b>Field name</b>	<b>Description</b>
Hub Publisher	Select this check box to mark the service container as a Hub publisher. If you do not select this check box, the service connector is configured as a spoke publisher.
Machine (name/IP)	IP address of the computer on which you are creating the service container is displayed by default. Specify either the machine name or IP address.
Port	Port number for connecting the service group. Specify a valid port number (greater than 1024). The machine IP address and port are used together to form a syncup ring between all the service containers within this service group, for exchanging cache information, load balancing, and failover messages.
Use different database for application	Select this check box if you do not want to place the application content in the same database as the MDM content. You must provide the database details for the application content in the Application tab that is displayed.

In the **Repository** tab, you must provide the details of the database where you want the MDM content to be placed. Similarly, in the **Application** tab, you must provide the details of the database where you want the application content to be placed.

The following table describes the fields on the Repository or the Application tab:

<b>Field name</b>	<b>Description</b>
Advanced Properties	Click  to view and enter the Advanced Properties in the <i>AppWorks Platform Advanced Development Guide</i> .
Select Database Configuration	Select one of the existing database configurations that have been created. The details of the selected database configuration appear in multiple fields within the same window. To create a new database configuration, select the New Database Configuration option. For details, see <a href="#">Creating a database configuration</a> .

### Security Administration Configuration interface

While configuring the security administration connector, to continue with an existing database configuration, select it from the Select Database Configuration list.

Following are the fields on the Security Administration tab.

<b>Field</b>	<b>Description</b>
Select Database Configuration	<p>To create a database configuration, select <b>New Database Configuration</b> from the list and enter the required information in the dialog box that opens. See <a href="#">Creating a database configuration</a>.</p> <p>To continue with an existing database configuration, select it from the list. The associated fields are automatically filled.</p>
Advanced Options	<p>Expand the group box and enter the necessary details. See Advanced properties in the AppWorks Platform Advanced Development Guide.</p>

### MDM service connector configuration interface

The following table describes the fields in the MDM Service Connector:

<b>Field name</b>	<b>Description</b>
Allow Parallel Processing	Select this property if the hub data store has more than one entity. This enables the hub service container to process the requests in parallel for different entities, resulting in an improved performance.
MDM Hub Publisher Service Group	Distinguished Name (DN) of the MDM Hub Publisher must be provided. Click  (Lookup Record) to select the DN of the required publisher.

### WS-AppServer service connection parameters interface

This topic describes the fields on the WS-AppServer service connection parameters interface.

The following table contains the WS-AppServer connector parameters:

<b>Field</b>	<b>Description</b>
Configure Database	Maps the WS-AppServer processor to the tables of a database to which AppWorks Platform is connected. This is selected by default. If this check box is cleared, you do not see any <b>Database Configuration</b> related fields and options. In that case, you can create the service container by providing the <b>Initializer Class</b>

Field	Description
	<p><b>Name.</b></p> <p><b>Note:</b> WS-AppServer service container can process the requests with implementation types that start with Bsf such as BsfJavaCall, BsfUpdate, BsfValidate, BsfMetaData, and BsfProcedure, only when the service container is configured with <b>Configure Database</b> option enabled.</p>
Auto Cleanup	<p>Free WS-AppServer NOM memory by removing object data associated with BusObjects immediately after the transaction is committed. It ensures availability of sufficient memory for the WS-AppServer. When the check box is clear, the Auto Cleanup function is inactive, and Java objects in the WS-AppServer memory are available even after the transaction is committed. They are cleaned up by garbage collection. The objects can be retrieved if required, before they are cleaned up. When you select this check box, the Auto Cleanup function is activated. It removes Java objects after every transaction, thus freeing the WS-AppServer memory. In this scenario, if you want WS-AppServer to retain some objects after the transaction is committed, use the <code>keepAlive()</code> method on those objects. For more information, see the <code>BusObject</code> class in Java APIs. For more information on the Auto Cleanup functionality, see Memory Management in WS-AppServer in the <i>AppWorks Platform Advanced Development Guide</i>.</p>
Initializer Class Name	<p>Class that implements the <code>iOnBsfConnector</code> interface. This helps you implement custom logic that runs whenever WS-AppServer starts or stops. This option is useful when you are not creating the WS-AppServer Service based on a database. For more information on the interface details, see the Java API documentation.</p> <p><b>Note:</b> The event listener <code>iOnBsfConnector</code> is valid and applicable only when WS-AppServer is run in service mode using the WS-AppServer service. It is not operational if WS-AppServer is</p>

Field	Description
	used in the absence of a WS-AppServer service in an embedded mode.
Application Initializer Class Name	<p>Implements the <code>IApplicationInitializeListener</code> interface. It helps you implement custom logic that is activated whenever the WS-AppServer starts. The associated event listener <code>ApplicationInitialize</code> can be used when WS-AppServer runs in service mode using the WS-AppServer service, as well as in embedded mode (in absence of a WS-AppServer service).</p> <p><b>Note:</b> Use this class to connect to several databases. However, the class is initialized only once for each database. Ensure that you place the class in the relevant classpath. For more information in implementing the class, see the Java API documentation related to the interface.</p>
Enable Rules	<p>Enables the <b>Rule Repository</b> tab where you can specify the following Rule Engine related information:</p> <ul style="list-style-type: none"> <li>■ <b>Organization:</b> Organization where the service is being created.</li> <li>■ <b>Select Database Configuration:</b> Select one of the existing database configurations on which the rule repository service will be based. The details of the selected database configuration appear in various fields.</li> </ul> <p><b>Note:</b> To create a new database configuration, select New Database Configuration. For information, see <a href="#">Creating a database configuration</a>.</p> <ul style="list-style-type: none"> <li>■ <b>Runtime Properties:</b> Specify runtime execution details. The following options are available: <ul style="list-style-type: none"> <li>• <b>Rule Action Threads:</b> Number of threads assigned for a rule action.</li> </ul> <p><b>Note:</b> If the number of transactions performed by the rule engine involving external actions is huge, increase the size of the rule action threads. The default size is 6.</p> </li> </ul>

Field	Description
	<ul style="list-style-type: none"> <li> <b>Max value for Dispatcher Queue</b>  <b>Size:</b> Maximum number of asynchronous rule actions that can be queued in the dispatcher queue. The default size is 50000.  <b>Note:</b> Setting it to 0 allows infinite number of rule actions in the queue and may result in an Out of Memory error. Depending on the requirements of the application, change the size of the Dispatcher Queue. A higher value of the parameter enables higher memory, and a lower value limits memory usage. This value is exposed to JMX as a cold setting.         </li> <li> <b>Trace Rule Execution:</b> Select the check box to enable monitoring of rule execution.  <b>Note:</b> Enabling this option impacts performance as this feature requires the recording of details in the database during the rule execution.         </li> </ul>
Enable Audit	Auditing of WS-AppServer data. However, auditing takes place only when the WS-AppServer audit configuration is defined.
Multi tenant	Enables WS-AppServer to run in Multitenant Mode. For details, see Running WS-AppServer in multitenant mode in the <i>AppWorks Platform Advanced Development Guide</i> .
Organization	Organization where the service is being created.
Select Database Configuration	<p>Select one of the existing database configurations that have been created. The details of the selected database configuration appear in the fields within the same window.</p> <p><b>Note:</b> To create a new database configuration, select <b>New Database Configuration</b>. For information, see <a href="#">Creating a database configuration</a>.</p>
Advanced Properties	Click  to view the advanced properties and fill them. See Advanced properties in the <i>AppWorks Platform Advanced Development Guide</i> .

## E-mail connector interface

The E-mail connector interface contains the following fields.

Field	Description
Authentication required	Option to enable authentication to send emails. You can enter the user authentication information using the SetProfile Web service operation of E-mail connector. See SetProfile (UDDI) in the <i>AppWorks Platform API Reference Guide</i> .
Content Type	The content of the email is in the text format by default. Select this check box to indicate that the email content is in the HTML format.
<b>Incoming mail server</b>	
Server Protocol	Protocol to retrieve emails using an email client. The following standard protocols are supported: <ul style="list-style-type: none"> <li>■ POP3: Select this protocol to retrieve emails only from the Inbox folder of your mailbox. This protocol allows an email client to download emails from the email server and then deletes them from the server.</li> <li>■ IMAP: Select this protocol to retrieve emails from all the folders of your mailbox. This protocol also allows an email client to download emails, but it still retains the emails on the mail server.</li> </ul>
Host Name / IP Address	Name of the server to receive emails.
Port number	Port at which the server protocol service is running. 110 (Default).
Set SSL	Option to enable security on the mail server. If you select this check box, the SSL Properties group box appears, displaying the SSL properties that are configured by default. These properties vary based on the selection in the Server Protocol list. To add more properties, click  , and provide the Name and Value in the new row.
<b>Outgoing mail server</b>	
Host Name / IP Address	Name of the SMTP server to send emails.
Port number	Port at which the SMTP service is activated. 25 (Default).
Set SSL	Option to enable security on the email server.

Field	Description
	If you select this check box, the SSL Properties group box appears, displaying the SSL properties that are configured by default. These properties vary based on the selection in the Server Protocol list. To add more properties, click  , and provide the Name and Value in the new row.

## CoBOC Services Configuration interface

### CoBOC service settings

The Service tab helps configure the required CoBOC services for a CoBOC service container.

The following table describes the fields on the Services tab.

Field	Description
Enable Events	Select if you are using Web service operations that use the Notification Service component for business objects transacting in CoBOC. <b>Note:</b> If you designed notification models for objects belonging to a template, they will not send notifications unless you select this check box.
Enable Admin	Select if CoBOC must publish its transactions to the admin service.
Enable Rules	Select if you are using Web service operations that invoke rules to run on objects transacting in CoBOC. If you created rules to run on objects belonging to a template, they will not run unless you select Enable Rules. The Rule Repository tab is visible only when you select Enable Rules on the Service tab.
Enable Timer Rules	Select to fire timer rules at appropriate schedules on objects transacting in CoBOC. This option is visible only when you select Enable Rules. The Scheduler tab is visible only when you select Enable Timer Rules.

The administrator can edit the default setting for the CoBOC Syncup configuration. For more information about the procedure to change the default setting, see Changing CoBOC service container SyncUp configuration in the *AppWorks Platform Advanced Development Guide*.

### CoBOC cache settings

The Cache tab helps configure the cache settings for a CoBOC service container.

The following table describes the fields on the Cache tab.

Field	Description
Hashbins	A memory store to hold business objects. Specify the number of such stores available for maintaining the cache. <ul style="list-style-type: none"> <li>■ Minimum value: 1</li> <li>■ Maximum value: 10 (Default)</li> </ul>
Initial Capacity	Minimum capacity of each hashbin (memory store) to store business objects. Specify the initial capacity of each hashbin in terms of the number of business objects it can hold. <ul style="list-style-type: none"> <li>■ Minimum value: 100 (Default)</li> </ul>
Object Idle Time (in Mins)	Time the cache instances can remain idle in the cache. Specify the maximum time (seconds) the instances can remain in the cache. <ul style="list-style-type: none"> <li>■ Minimum value: 1</li> <li>■ Default value: 60</li> </ul>

## Database settings

The Database tab helps configure the CoBOC database.

The following table describes the fields on the Database tab.

Field	Description
Select Database Configuration	To create a database configuration, select <b>New Database Configuration</b> from the list and enter the required information in the dialog box that opens. See <a href="#">Creating a database configuration</a> .
	To continue with an existing database configuration, select it from the list. The associated fields are automatically filled.
Advanced Options	Expand the group box and enter the necessary details. See Advanced properties in the <i>AppWorks Platform Advanced Development Guide</i> . <b>Note:</b> Setting Cursor Cache Size to a high value causes high memory consumption. Based on the application usage, an administrator must set an optimal value. This is also applicable to the Query Cache size. For example, the optimal cursor cache size can be between 50 - 100, subjective to the application usage.

## Rule repository settings

The Rule Repository tab helps configure the settings for rule execution.

The following table describes the fields on the Rule Repository tab.

Field	Description
Specify Service	Rule Repository service to which the rules are published. Click  , select the required rule repository service in the Choose Target window, and then click <b>OK</b> .
Rule Action Threads	Number of threads that are assigned for a rule action. If the number of transactions performed by the rule engine, involving external actions, is huge, increase the size of the rule action threads. The default size is 5.
Trace Rule Execution	Enables rule tracking. <b>Note:</b> Enabling this option impacts performance because this feature demands recording of details in the database during the rule execution.
Monitor Rule Execution  <b>Note:</b> This feature was deprecated in Cordys BOP 4.1.	Enables monitoring of the rules. The status of the rules can be monitored from the CoBOC Transaction Monitor.
Max value for Dispatcher Queue Size	Maximum number of asynchronous rule actions that can be queued in the dispatcher queue. The default size is 50000. <b>Note:</b> Setting it to 0 allows infinite number of rule actions in the queue and might result in an Out of Memory error. Depending on the application's requirements, change the size of the dispatcher queue. A higher value of the parameter allows for higher memory and a lower value limits memory usage. It is exposed to JMX as a cold setting.

## Event settings

The following table describes the fields on the Event tab.

Field	Description
Event Display URL	Web page that is displayed when a new event occurs in the cache. By default, this box contains a URL, which is displayed to the user when a new event is pushed into the AppWorks Platform Inbox. You can define the content of the page.
Event Display ID	Display ID of the AppWorks Platform frame in which the URL is displayed.
Event Display Frame	AppWorks Platform frame to select to display the URL.
Notification	Notification service to which the events are published. Click  , select

Field	Description
Service	the required notification service in the Choose Target window, and then click <b>OK</b> .

### Scheduler settings

Field	Description
Select Database Configuration	To create a database configuration, select <b>New Database Configuration</b> from the list and enter the required information in the dialog box that opens. See <a href="#">Creating a database configuration</a> .
	To continue with an existing database configuration, select it from the list. The associated fields are automatically filled.
Advanced Options	Expand the group box and enter the necessary details. See Advanced properties in the <i>AppWorks Platform Advanced Development Guide</i> .

### Monitor service group configuration interface

The following table describes the Monitor Service Group configuration interface:

Field name	Description
Batch Size	Maximum number of service containers that can be started simultaneously.
Computer Name	Name of the computer where the service group monitor has to be created. The computer name must not contain the DNS suffix.
Configure ISVP Connector	Configuration of the application package connector. See the <a href="#">Application package connector interface</a> for more information.
Java Virtual Machine	Name of the Java Virtual Machine (JVM) on which the monitor is running.
Machine Name / IP Address	IP address of the participating computer in the syncup.
Port Number	Port number for connecting the service group.
Refresh Rate (ms)	Refresh rate in milliseconds for querying each service container.
Startup Time (ms)	Maximum timeout allowed for a service container to start before accepting another service container in the batch.

## UDDI connector interface

The UDDI Connector interface contains the following fields.

Field	Description
Use HTTP Proxy	Option to send HTTP requests through a proxy server.
Host	Name of the proxy server.
Port	Port of the proxy server.
Authentication Type	Type of authentication required for the proxy server.
Username	Authenticated user name for the proxy server.
Password	Authenticated user password to access the proxy server.
Maximum Response Timeout	Time frame in milliseconds that UDDI uses to terminate a connection or the HTTP connection that is established with the endpoint URI of the Web service.
Ignore Certificate Validation	Option to disable the validation of certificates that are used to invoke a Web service. <b>Note:</b> The UDDI connector can be used to communicate or invoke Web services over a SSL secured connection. The certificates that are used while invoking such a service are validated by the HTTP client when the service container starts. The validation takes place at the service container level for all the Web service operations. Select this option if you do not want this validation.

**Note:** You can provide the username and password details only if you select the Authentication Type as Basic, Digest, or NTLM. If you select Anonymous authentication as the Authentication Type, the username and password fields are disabled.

## FTP Configuration interface

The FTP Configuration interface contains the following fields.

Field	Description
Name	Name of the FTP configuration being created.
Description	Description of the configuration.
Server	FTP server host machine.
Port	Port number used by the FTP server.
Username	User ID of the user who accesses the FTP server.
Password	Password of the user who accesses the FTP server.

Field	Description
Base Directory	Name of the directory from which the files are uploaded to the FTP server. <b>Note:</b> In a high availability setup, ensure to specify a base directory that is based on the SAN file system or NTFS.
Mode	Mode of the FTP server: <ul style="list-style-type: none"><li>■ Active: Establishes a connection to the FTP server in active mode.</li><li>■ Passive: Establishes a connection to the FTP server in a passive mode.</li></ul>
Protocol	Standard protocols used for file transfer: FTP and SFTP.

### Application package connector interface

The application package connector helps you manage applications. It provides methods that are internally used for creating, installing and uninstalling applications. The interface is divided into the following tabs.

#### Credentials

**User name and Password:** The credentials are required to access the Applications folder (`<<AppWorks_Platform_installdir>\isvcontent\packages`). These credentials are the same as those entered during the installation of AppWorks Platform.

During installation, if you choose anonymous authentication for the applications, leave these fields empty.

#### Database

This tab displays the details of the database that is configured during installation. You cannot change the database configuration of the Monitor. Therefore, the fields that are displayed cannot be edited. However, you can change the advanced properties. See Advanced properties in the *AppWorks Platform Advanced Development Guide*.

#### OS process configuration interface

The following table describes the fields in the OS process configuration interface.

Field	Description
Name	Name of the OS process.
JVM Arguments	If there are any arguments to pass for the Java executable. To include multiple arguments, type them on separate lines.

### Modifying a service container

Modifying a service container allows you to:

- Change the configuration details of a service container
- View log details
- Add application connectors

### Before you begin

You must have the role of systemAdmin or organizationalAdmin to modify the service container information in an organization.

### To modify a service container:

1. On the **Welcome** page > **My Applications**, click  (System Resource Manager). System Resource Manager lists the available service containers.
  2. Select the service container and double-click it. Properties are displayed in the Properties - <Service Container Name> App Palette.
  3. Modify the necessary information. For more information, see Service Container Configuration Interface in the *AppWorks Platform Advanced Development Guide*.
- Note:** To configure an application connector, see Configuring Application Connectors in the *AppWorks Platform Advanced Development Guide*.
4. Click .
- The service container information is modified.

## Configuring OS processes for a service container

Multiple service containers can be configured to run in a single JVM by creating an OS process for a JVM. See [Running multiple service containers in a single JVM](#). The service container is configured and runs within the selected OS process in a JVM, which helps optimize memory usage as several service containers run within a single JVM.

The Application Server OS Process is created during the installation of AppWorks Platform. It cannot be deleted or modified. After installation, by default, the following service containers are assigned to the Application Server OS Process:

- Platform
- Notification
- Business Process Management

**Important:** In a general scenario, a single OS process can be shared by different service containers. Consider these exceptions:

- The Business Process Management service container can share an OS Process with the following service containers as long as they point to the same CoBOC database details:

- CoBOC
  - Data Transformation
  - Rule Repository
  - Scheduler
- Do not configure more than one WS-AppServer Service to a single OS Process. Doing so results in configuration errors. When adding more WS-AppServer service containers to an existing WS-AppServer Service, do not configure WS-AppServer service containers so they point to different databases using different database configurations. If service containers point to different databases, they might cause problems with data retrieval and persistence process.

#### To assign an OS Process to a service container:

1. Select **Assign OS Process** in the Properties - <Service Container Name> App Palette. See Service Container Configuration Interface in the *AppWorks Platform Advanced Development Guide*.
2. Set the required OS Process.

**Note:** Using Assign OS Process, you can assign and unassign service containers to the Application Server OS Process and set the Application Server OS Process. This ensures that the assigned service containers run in TomEE.

For more information on creating and modifying OS processes, see:

- [Creating an OS process](#)
- [Modifying an OS process](#)
- [Displaying configured service containers](#)
- [Deleting an OS process](#)

#### ***Creating an OS process***

By creating an OS process for Java Virtual Machine (JVM), you can group and run multiple service containers in a single OS process, which saves memory space compared to running the service containers individually.

##### **Before you begin:**

You must have the role of systemAdmin or organizationalAdmin to create an OS process.

##### **To create an OS process:**

1. On the **Welcome** page > **My Applications**, click  (System Resource Manager). The System Resource Manager window opens, listing the available service containers in the service containers section.
 

**Tip:** You can also click  to view the existing service containers.
2. Click . The Manage OS Processes dialog box opens.

3. In the **Monitor Service Container** list, select a value.  
The available OS processes are listed in **OS Process - <Monitor Service Name>**.
4. Click to create an OS process.  
The **OS Process Properties** group box is displayed in the same window.
5. Specify the following details:
  - **Name:** Name of the OS process.
  - **JVM Arguments:** The arguments, if any, that need to be passed for the Java executable.  
**Note:** To include multiple arguments, type them in separate lines.
6. Click (Save).  
The OS process is created in the Monitor Service Container. After creating an OS Process, restart the Monitor.

#### To configure a service container to an OS process:

1. In **Properties - <Service Container>**, select **Assign OS Process**.
2. Select an OS Process.
3. Click (Save).

### *Modifying an OS process*

You can edit the name and Java Virtual Machine (JVM) arguments associated with an OS process.

#### Before you begin

You must have the role of systemAdmin or organizationalAdmin to modify an OS process.

#### To modify an OS process:

1. On the **Welcome** page > **My Applications**, click (System Resource Manager).  
The System Resource Manager window opens, listing the available service containers in the Service Containers App Palette.
- Tip:** You can also click to view the existing service containers.
2. Click in the App Palette.  
The Manage OS Processes window opens.
3. In the **Monitor Service Containers** list, select a value.  
The available OS processes and **Application Server** are listed in the **OS Processes - <Monitor Service Container Name>** group box.
4. Click the OS Process that you intend want to modify.  
The **OS Process Properties - <OS Process Name>** group box is displayed with the associated properties.

**Note:** The **Application Server** OS process cannot be modified. Clicking it displays the path in which TomEE is installed.

5. Modify the required OS process details and click **Save**.

The selected OS Process is modified. For more information, see [OS process configuration interface](#).

## Displaying configured service containers

Multiple service containers can be configured to a single OS process. After configuration, you can view the configured service containers in the System Resource Manager.

### Before you begin:

- You must have the role of systemAdmin or organizationalAdmin to view the service containers on an OS process.

### To display the configured service containers:

1. On the Welcome page, click  (System Resource Manager).  
The System Resource Manager window opens. The available service containers are displayed in the Service Containers App Palette.

**Tip:** Alternatively, click  to view the existing service containers.

2. Click  in the App Palette.  
The Manage OS Process window opens.
3. Select a Monitor service container.  
The available OS processes are displayed in **OS Process - <Monitor Service Name>**.
4. Select an OS Process to view the configured service containers.  
The **Service Containers Configured** group box appears. All the service containers that are grouped under the selected OS process are displayed.  
  
The service containers configured to an OS process are displayed.

## Deleting an OS process

Deleting an OS process detaches the respective service containers so they can run as individual JVMs.

### Before you begin

You must have the role of systemAdmin or organizationalAdmin to delete an OS process.

### To delete an OS process:

1. On the **Welcome** page > **My Applications**, click  (System Resource Manager).  
The System Resource Manager window opens, listing the available service containers in the Service Containers App Palette.

**Tip:** You can also click  to view the existing service containers.

2. Click  in the App Palette.  
The Manage OS Processes window opens.
3. In the **Monitor Service Containers** list, select a value.  
The available OS processes and **Application Server** are listed in the **OS Process - <Monitor Service Name>**.
4. Select the OS process and click  (Delete).  
A confirmation message appears.  
**Note:** The **Application Server** OS process cannot be deleted. Clicking it displays the path in which TomEE is installed.
5. Click **Yes** in the confirmation dialog box.  
The selected OS process is deleted.

## Monitoring a service container

You can start, stop, restart, view logs, and perform other tasks related to service containers.

### Before you begin

You must have the systemAdmin or organizationalAdmin role to monitor service containers in an organization.

### To monitor a service container:

1. On the **Welcome** page > **My Applications**, click  (System Resource Manager).  
The System Resource Manager window opens, listing the available service containers.
2. Right-click a service container and perform one of the following actions:

Click	Description
<b>Start</b>	Starts the service container.
<b>Stop</b>	Stops the service container.
<b>Restart</b>	Restarts the service container.
<b>Reset</b>	Resets the service container.
<b>Delete</b>	Deletes the service container.
<b>View Log</b>	View log details for a service container. See <a href="#">Log viewer interface for a service container</a> .
<b>Show Error Details</b>	View configuration error details of a service container.
<b>Clone Service</b>	Copies the details of the selected service container so you can create a new service container by modifying the details. See <a href="#">Clone Service Container Configuration Interface</a> in the <i>AppWorks Platform Advanced Development Guide</i> .

Click	Description
<b>Properties</b>	Displays <b>Properties &lt;Service Container Name&gt;</b> App Palette at the bottom. See Service Container Configuration Interface in the <i>AppWorks Platform Advanced Development Guide</i> .

## Pausing a service container

This topic describes how to pause and resume all the service containers in a cluster.

Pausing and resuming of service containers is a cluster-wide setting. When pausing or resuming, all the service containers in the cluster move to either the passive or started state.

Service containers have one of the following states:

- **Started:** Service containers act normally, processing incoming requests, and triggering work themselves. Only the active service containers like BPM, Scheduler, CWS and MDM can trigger work, such as handling a timeout in a BPM process instance or starting a new schedule by themselves.
- **Passive:** Active service containers still respond to incoming requests, but they do not initiate work themselves. In the passive state, the BPM service containers do not handle BPM process timeouts, and the scheduler service containers do not trigger new schedules. This means that a paused cluster does not initiate work by itself, because all the service containers are in a passive state.
- **Stopped:** A stopped service container cannot process any requests.

Service containers handle incoming web service requests, like GetCustomers and UpdateOrder. When no web service request is sent, no response is sent.

Active service containers can trigger work by themselves, without an incoming web service request. Active service containers include BPM, Scheduler, CWS and MDM service containers. These service containers trigger work like handling a timeout in a BPM process instance, or starting a new schedule. In specific situations, such as upgrading a cluster, the system must not trigger work by itself.

### Before you begin

You must have the systemAdmin role to pause and resume service containers in a cluster.

### To pause or resume service containers in a cluster:

1. On the **Welcome** page > **My Applications**, click  (System Resource Manager). The System Resource Manager lists the available service containers.  
**Note:** Pause and resume of service containers can only be performed from the system organization.
2. Click one of the following on the toolbar:

<b>Pause service containers</b>	a. Click <b>Pause all Service Containers</b> . b. Click <b>Yes</b> to pause all service containers. The status of the started service containers change to passive.
<b>Resume service containers</b>	a. Click <b>Resume all paused Service Containers</b> . b. Click <b>Yes</b> to resume all paused service containers. The status of the paused service containers changes to started.

**Note:** The passive state behaviors for BPM, Scheduler, and BAM are as follows:

- Short-lived BPMs and page flow BPMs are processed as usual even in the passive state.
- Long-lived BPMs are queued in the passive state, and resumed upon activation.
- All delays, time-outs, and escalations within a BPM are put on hold in the passive state, and resumed upon activation.
- Schedules are not invoked in the passive state. All the missed schedules are triggered upon activation.
- Fail over management is put on hold in the passive state, and resumed upon activation.
- Scheduled process archiving is not invoked in the passive state, and is resumed upon activation. However, manually triggered process archival is processed as usual.
- BAM event processing is put on hold in the passive state, and resumed on activation.

## Managing connection points

A connection point is a generic label of a middleware address in AppWorks Platform. This contains administrative information about the middleware, such as the Uniform Resource Identifier (URI) of the socket. For information about types of connection points available in AppWorks Platform, see Connection Point Configuration Interface *AppWorks Platform Advanced Development Guide*.

The connection points can be created, modified, and deleted in AppWorks Platform.

- [Creating a connection point for a WebGateway instance](#)
- [Modifying a connection point](#)
- [Deleting a client connection point group](#)

## Creating a connection point for a WebGateway instance

Create a new instance of the WebGateway by creating a new connection point. While creating the connection point, specify the ESB client listening port.

### To create a connection point for a WebGateway instance:

1. Click **Start > Programs > <AppWorks Platform Version> > <Instance Name> > Tools > Management Console.**

**Tip:** On Linux-based computer, do the following:

- Go to the Linux terminal and navigate to the `<AppWorks Platform_installdir>/bin`.
- Type `./cmc.sh` to run the shell script.  
The Management Console window opens.

2. Right-click `cordys cn (cn=cordys)` and click **New**.  
The Add window opens.
3. Type a name for the WebGateway instance in **Name**. The name must be the name as given in the `com.eibus.web.gateway wcp.property` prefixed with a number up the value given in the `com.eibus.web.gateway.maxinstances wcp.property`.

For example, with `com.eibus.web.gateway.maxinstances=2` and  
`com.eibus.web.gateway=webgateway@srv-cordys-1w` create the following entries:  
`1webgateway@srv-cordys-1w` and `2webgateway@srv-cordys-1w`.

The `cn` details are automatically generated in **Parent in Directory Service**.

4. From Object Class, select **busconnectionpoint**.
5. Click **Next**.  
The Attribute Viewer opens.
6. In **labeleduri**, specify a port number. The format is `socket://<machine name>:port number`.

**Note:** Assign a port that is within the non-ephemeral port range, which by default is 10000-32767. Do not use a port that is already in use by another connection point or application. For more information, see [Port ranges within AppWorks Platform](#).

7. Click **OK**.  
The connection point for the new instance is created.

## Modifying a connection point

This topic describes the procedure to modify the configuration of a connection point.

### Before you begin

You must have the role of an administrator to modify a connection point in an organization.

**To modify a connection point:**

1. On the **Welcome** page > **My Applications**, click  (System Resource Manager). System Resource Manager lists the available service containers.
2. Right-click a service container and select **Manage Connection Points**. The Connection Points App Palette opens.
3. Double-click the connection point to modify it. The Connection Point dialog box opens.
4. Edit the connection point details and click **Save**. The connection point details are modified. For more information, see Connection Point Configuration Interface in the *AppWorks Platform Advanced Development Guide*.

## Managing client connection point groups

Service consumers or client applications send SOAP requests to service groups using a communication interface called a connection point.

Connection points are available for TCP/IP and serve as a network address for service containers. For example, service containers receive a message on a socket connection point and send a reply on a socket connection point.

Service containers can have one or multiple connection points. Depending on the SOAP request involving multiple connection points, the connection points can be grouped as client connection point groups.

- [Creating a client connection point group](#)
- [Modifying a client connection point group](#)
- [Deleting a client connection point group](#)

### Creating a client connection point group

Similar connection points can be grouped together as a client connection point group. All the connection points created under a group must be of the JMS type.

**Before you begin:**

- You must have the role of systemAdmin or organizationalAdmin to create a client connection point group in an organization.

**To create a client connection point group:**

1. On the **Welcome** page > **My Applications**, click  **System Resource Manager**. The System Resource Manager window opens, listing available service containers in the Service Containers App Palette.

**Tip:** Or, click  to view the existing service containers.

2. Right-click a service container and select **Manage Client Connection Point Groups**.  
The Connection Point Groups dialog box appears, listing the existing groups.
3. To add a connection point group, click .  
A row is added in **Connection Point Groups**.
4. Type a name for the group and click (Save).  
The Connection Points - <Connection Point Group Name> table allows you to create connection points. See [Creating a connection point for a WebGateway instance](#)
5. Click (Save).  
A client connection point group is created.
6. For the changes to be effective, restart the service.

## Modifying a client connection point group

You can modify a connection point group in the following ways:

- Change the name
- Create connection points
- Delete connection points
- Modify connection point properties

### Before you begin

You must have the role of systemAdmin or organizationalAdmin to modify a connection point group in an organization.

#### To modify a client connection point group:

1. On the **Welcome** page > **My Applications**, click (System Resource Manager).  
The System Resource Manager window opens, listing available service containers in the Service Containers App Palette.
2. Right-click a service container and select **Manage Client Connection Point Groups**.  
The Connection Point Groups dialog box opens, listing the existing groups.
3. Select a client connection point group to modify.  
The corresponding connection points appear in the **Connection Points - <Connection Point Group name>** table.
4. Make the necessary changes and click .  
The client connection point group details are modified.

## Deleting a client connection point group

Deleting a client connection point group deletes all its corresponding connection points.

## Before you begin

You must have the role of systemAdmin or organizationalAdmin to delete a client connection point group in an organization.

**Note:** You can delete one or more connection points that belong to a group.

### To delete a client connection point group:

1. On the **Welcome** page > **My Applications**, click  (System Resource Manager).  
The System Resource Manager window opens, listing available service containers in the Service Containers App Palette.
2. Right-click a service container and select **Manage Client Connection Point Groups**.  
The Connection Point Groups dialog box opens, listing the existing groups.
3. Select a group in the Connection Point Groups table.
4. Click  (Delete).
5. Click **Yes** in the confirmation dialog box.
6. Click  (Save).  
The selected Client Connection Point Group is deleted.

## Working with Web Service Interface Explorer

A Web service operation usually contains a query or a task that determines the data that is being requested or the action that must be taken. Web service interfaces are an aggregation of Web service operations and are associated with service groups. These Web service operations contain an implementation and an interface. See Implementation of a Web service operation in the *AppWorks Platform API Reference Guide* for implementation and see [Web service operation interface](#) for interface. AppWorks Platform exposes its Web service operations as Web services.

Every Web service operation contains two schema definitions:

- Web Services Description Language (WSDL) - WSDL is XML grammar that provides the detailed description of the request or response to send or receive, including parameters and data types.
- XML Schema Definition (XSD) - XSD specifies the type definition of the business object to return.

You can work with Web service interfaces using the Web Service Interface Explorer. To open it, click  **Web service Interface Explorer** on the Welcome page or in the System Resource Manager window. You can search for Web service interfaces or operations. See [Searching for Web services in Web service Interface Explorer](#). Perform the following:

- Test a Web service operation. See [Validating Web service operations](#).
- Delete a Web service interface or operation. See [Deleting a web service interface or operation](#).

- View the implementation and interface of a Web service operation. See [Viewing the implementation and interface of a Web service operation](#). You can also set security permissions. See [Configuring ACL for Web service interfaces and operations](#). And publish Web services to an external UDDI registry. See Publishing Web services to an external UDDI registry in the *AppWorks Platform Advanced Development Guide*. Web service interfaces are classified as follows:

- Packaged: These originate from the application packages and are represented as .
- Custom: These are generated by the user as per the business needs and are represented as .

## Validating Web service operations

Use the Operations Test tool to test Web service operations. This tool validates a Web service operation by composing the SOAP request and sending it to the appropriate service group for execution.

### Before you begin:

- A service group that implements the Web service operation to be tested must be configured and the service container must be running.
- You must have the role of systemAdmin or organizationalAdmin.

### To validate Web service operations:

1. Search for the Web service operation to test. See [Searching for Web services in Web service Interface Explorer](#).  
The relevant Web service operations are displayed in Search Results.
2. Right-click a Web service operation and select **Test**.

**Note:** You can also do the following:

1. In Search Results, right-click one of the results and select **Properties** to display the Web service interfaces and operations in their corresponding App palettes.
2. In the Web Service Operations App Palette, right-click an operation and select **Test** to display the Operation Test Tool.
3. Replace the default text **PARAMETER** in the Task Request section with a value for the attribute to send to the selected Web service operation.
4. Click **Invoke**.  
The Result Messages section is updated with the request and response messages.
5. Click a **Response or Request Message** for the results of the Web service operation for the specific parameter value.  
The message received in the XML format is displayed. See [Result Messages interface](#).
6. Click **Clear Messages** to clear the request and response messages from the Result Messages section.

**Note:** Depending on the option selected, the following values are set for the message options while sending the SOAP request.

Option	Values set
None	0
No Reply	1

## Deleting a web service interface or operation

This topic describes the procedure to delete a web service interface or operation.

### Before you begin

- You must have the role of systemAdmin or organizationalAdmin.
- You cannot delete the application package based Web service interfaces or operations.

### To delete a web service interface or operation:

1. Search for the Web service interface or operation to delete. See [Searching for Web services in Web service Interface Explorer](#).  
The relevant Web service operations are displayed in the Search Results box.
2. Right-click a Web service interface or operation and select **Properties**.  
The Web Service Interface Properties - <Name of the Web Service Interface> dialog box displays the Web service interfaces and operations in their corresponding App palettes.
3. Right-click a web service interface or web service operation, and select **Delete**.
4. Click **Yes** in the confirmation dialog box.  
The Web service interface or operation is deleted.

## Viewing the implementation and interface of a Web service operation

Every operation contains an implementation, interface (WSDL), and XSD. See Implementation of a Web service operation in the *AppWorks Platform API Reference Guide* and [Web service operation interface](#). To use an operation, the interface must be clearly and unambiguously described. The standard language for describing the interface of Web services is WSDL (Web Service Description Language). A requester must know how to address the service and the kind of response to expect.

**Before you begin:**

- You must have the role of systemAdmin or organizationalAdmin to view implementation and interface details of a Web service operation.

**To view the implementation and interface of a Web service operation:**

1. Search for a Web service operation. See [Searching for Web services in Web service Interface Explorer](#).  
The relevant Web service operations are displayed in Search Results.
2. Right-click a Web service operation and select **Properties** to display the Web service interfaces and operations in their corresponding App palettes.
3. In the Web Service Operation App Palette, right-click an operation and select **Interface** or **Implementation**.  
The corresponding interface and implementation details are displayed.

## Managing database configurations

In AppWorks Platform, some application connectors can be configured to use relational databases for storage and retrieval of data. Database connection details for application connectors must be specified during service container configuration for the application connectors to use the information to connect to the database.

Database-related configurations are stored in a common location in the repository for all application connectors to reuse the configuration by a name.

A database configuration consists of the following fields, which are associated with a descriptive name and stores the configuration details (drivers and settings) of a database.

- JDBC Driver
- Driver Class
- Connection String
- JDBC Driver XA Class
- Default Database
- DB User
- Password

For the JDBC driver database configuration, see [JDBC details interface](#).

While configuring services, you can reuse a stored database configuration without entering the details again.

You can manage a database configuration through the following tasks:

- [Creating a database configuration](#)
- [Modifying a database configuration](#)

- [Deleting a database configuration](#)
- [Viewing the database configuration usage report](#)

## Creating a database configuration

You can create a database configuration to store the database details.

### Before you begin:

You must have the role of a systemAdmin or organizationalAdmin to create a database configuration.

### To create a database configuration:

1. On the **Welcome** page > **My Applications**, click  (System Resource Manager).  
The System Resource Manager window opens.
  2. On the toolbar, click  **Manage Database Configurations**.  
The Manage Database Configurations page opens.
  3. Click .

A new row is added to the table and a section to provide database configuration details is displayed in the bottom pane.
4. In the **Name** field, type the name of the database configuration.
  5. This is optional. In the **Description** field, describe the database configuration.
  6. Provide the relevant details in the corresponding fields. See [JDBC details interface](#).
    - Select **Create New Database** to create a database. Provide appropriate details in the **DBA Name** and **DBA password** fields.
    - You can create a tablespace while creating a new database for PostgreSQL or new user for Oracle, if you select Oracle Thin/OCI or PostgreSQL as your JDBC driver. See [Creating a tablespace in the AppWorks Platform Advanced Development Guide](#).
  7. To ensure AppWorks Platform can connect to the database with the provided parameters, click  (Test Connectivity).
  8. Click  (Save).

A new database configuration is created and the name, description, and organization under which the database configuration is being created, are displayed in the new row.

**Note:** For a new user in Oracle, only Create permissions are granted to the user. For other permissions, connect to the Oracle server using the client and select the permissions.

**Note:** You can create database configurations while deploying an application package or creating a service group.

- Application package deployment: On the Provide Required Inputs page, select **New Database Configuration** from the **Select Database Configuration** list. See [Deploying applications](#).

- Service Group Creation: On the Provide Details for <Connector> page of the New Service Group wizard, select **New Database Configuration** from the **Select Database Configuration** list. See [Creating a service group](#)

## Modifying a database configuration

You can modify the description, and the JDBC driver details of a database configuration. The name cannot be changed.

**Note:** After modifying a database configuration, you must restart all service containers that are configured to that database.

### Before you begin

You must have the role of a systemAdmin or organizationalAdmin to modify a database configuration.

### To modify a database configuration:

1. On the **Welcome** page > **My Applications**, click  ( System Resource Manager).  
The System Resource Manager window opens.
2. In the toolbar of System Resource Manager window, click  (Manage Database Configurations).  
The Manage Database Configurations page opens.
3. Select **Database Configuration** to modify and double-click it.  
The bottom section is enabled for editing.
4. Modify the necessary details. For more information, see [JDBC details interface](#)
5. To ensure AppWorks Platform can connect to the database with the provided parameters, click  (Test Connectivity).  
A confirmation message indicates the status.
6. Click  (Save).  
The database configuration is modified.
7. Restart all service containers that are configured to the modified database configuration.

**Note:** If TomEE is used as a Web application server, and a database configuration is used by a component running in TomEE, such as Entity Runtime, restart TomEE.

## Deleting a database configuration

You can delete database configurations that are not being used by any service group.

### Before you begin

You must have the role of a systemAdmin or organizationalAdmin to delete a database configuration.

**To delete a database configuration:**

1. On the **Welcome** page > **My Applications**, click  (System Resource Manager).  
The System Resource Manager window opens.
2. On the toolbar of System Resource Manager window, click  **Manage Database Configurations**.  
The Manage Database Configurations page opens.
3. Select the check box next to the database configuration to delete and click  **Delete**.
4. Click **Yes** in the confirmation dialog box.  
The database configuration is deleted.

## Viewing the database configuration usage report

The usage report displays the service containers that are configured to a database.

**Before you begin:**

- You must have the role of a systemAdmin or organizationalAdmin to view the usage report of a database configuration.

**To view a database configuration usage report:**

1. On the Welcome page, click  (System Resource Manager).  
The System Resource Manager window opens.
2. On the toolbar of the System Resource Manager window, click  (Manage Database Configurations).  
The Manage Database Configurations window opens. The existing database configurations are displayed.
3. Select the check box for a database configuration to view its usage report.

**Tip:** You can select multiple database configurations and view the consolidated report.

4. Click .
- The Database Configurations Usage Summary page is displayed. The usage report provides the following information:
- Database Configuration: Name of the database configuration.
  - Organization: Organization under which the database configuration was created.
  - Service: Service containers using the database configuration.
  - Service Organization: Name of the organization under which the service container was created.

# Configuring service containers for reliable messaging

Reliable messaging is the ability to queue service request messages and ensure guaranteed delivery of these messages to their corresponding destinations. It includes the ability to provide message delivery notification to the message sender or service requester.

AppWorks Platform supports distributed transactions for its components. They can deliver and receive messages reliably, even when there are component, system, or network failures. The reliable messaging feature of AppWorks Platform uses distributed transactions through the two-phased [eXtended Architecture \(XA\)](#) protocol and supports Java Message Service (JMS) queues.

In AppWorks Platform, multiple components communicate through SOAP messages. To ensure the delivery and acknowledgment of these messages, you must configure the components involved in a transaction to a reliable messaging infrastructure in one of the following ways:

- Run the service containers in the same TomEE application server or the same OS process.
- Use JMS with a JMS Connector.

When service containers are configured in the same OS process and communicate with each other, the ESB layer initiates a global transaction for the components to participate and follow a two-phase commit mechanism. The transaction manager coordinates this transaction and invokes a Prepare command to all the participants. After the acknowledgment is received from all the participants, it invokes the Commit command. In this approach, if there is a failure at any instance, infrastructure or application related, the entire transaction is reverted.

To communicate reliably with AppWorks Platform, you can use an external JMS queue that can be handled by the JMS Connector. See [Configuring the JMS connector](#). To use messages from an external queue, for example in a BPM process, you must configure the BPM and JMS service containers in the same TomEE application server or the same OS process.

## Example:

When there is an external JMS queue with a message to be used in a business process, the JMS connector is configured to handle the queue messages. The business process sends a request to the JMS connector to retrieve another message from the queue and then uses the data from the message to store it in a database through WS-AppServer or add it to an entity. In such scenarios, the JMS connector, the BPM connector, and the WS-AppServer connector must be configured to run in the same TomEE application server or in the same OS process. Even if Entity Runtime is used, the scenario is reliable as Entity Runtime runs in the TomEE application server.

To configure a database for distributed transactions, see [Configuring AppWorks Platform service containers for distributed transaction](#).

## Configuring AppWorks Platform service containers for distributed transaction

An XA compliant driver can participate in an XA compliant transaction as defined in the X/Open specification. The key advantage is that resources from different vendors can participate in a single distributed transaction because many of the JDBC drivers from the major vendors are XA compliant. You can also search for a driver in the manufacturer's website.

### ***Configuring or enabling XA transactions***

#### **To configure or enable XA transactions:**

1. Enable XA in AppWorks Platform.
2. Configure the database server to support XA transactions.

#### **To enable XA in AppWorks Platform:**

In service container configuration, set

`com.opentext.process.platform.enable.distributed.transaction=true` as a JRE configuration argument. For detailed steps, see [Modifying a service container](#).

#### **To configure a database server to support XA transactions:**

<b>SQL Server</b>	<p>To configure the Microsoft SQL server JDBC driver to support XA transaction, see <a href="https://docs.microsoft.com/en-us/sql/connect/jdbc/understanding-xa-transactions">https://docs.microsoft.com/en-us/sql/connect/jdbc/understanding-xa-transactions</a> and do the following:</p> <ol style="list-style-type: none"> <li>1. Download and install the latest Microsoft SQL server JDBC driver from <a href="http://msdn.microsoft.com/en-us/data/aa937724.aspx">http://msdn.microsoft.com/en-us/data/aa937724.aspx</a>.</li> <li>2. Copy <code>sqljdbc_xa.dll</code> from the <code>&lt;JDBCDRIVERFOLDER&gt;</code> to the bin folder of SQL Server installation. This is applicable to all the SQL servers.</li> <li>3. Register the DLL using the <code>xa_install.sql</code> script. To register a DLL, you must have administrative privileges on SQL Server.</li> <li>4. Enable the XA transaction for MSDTC on the SQL server system and restart the SQL server.</li> <li>5. Set the <code>sqljdbc.jar</code> driver in the service container class path.</li> </ol>
<b>Oracle</b>	<p>To enable XA transactions in the Oracle database, do the following:</p>

	<ol style="list-style-type: none"> <li>1. Start the Oracle SQL command prompt.</li> <li>2. Connect with the credentials: sys as sysdba.</li> <li>3. Grant the following rights:           <ul style="list-style-type: none"> <li>■ Grant select on sys.dba_pending_transactions to &lt;user name&gt;;</li> <li>■ Grant select on sys.pending_trans\$ to &lt;user name&gt;;</li> <li>■ Grant select on sys.dba_2pc_pending to &lt;user name&gt;;</li> <li>■ Grant execute on sys.dbms_system to &lt;user name&gt;;</li> </ul> </li> </ol> <p><b>Note:</b> &lt;user name&gt; refers to the credentials of the AppWorks Platform user who has connected to the database (as specified in the AppWorks Platform Database configuration).</p>
<b>PostgreSQL</b>	Set the <code>max_prepared_transactions</code> parameter in the <code>postgresql.conf</code> file. The value must be at least as large as the <code>max_connections</code> value.

## Transaction configuration

Service containers run both inside and outside TomEE. You can set several options, for example, txRecovery and log file location.

### To change the transaction configuration:

- For service containers running inside TomEE, use the `<tomee>/conf/tomee.xml` file. For more information on properties and how to set them, see <http://tomee.apache.org/transactionmanager-config.html>.
- For service containers running outside TomEE, set the configuration with the corresponding Java system properties that have the prefix `com.cordys.xatransaction.geronimo`. For example:

```
com.cordys.xatransaction.geronimo.logFileDir=/opt/transactionlog
```

**Note:** The default values of the properties inside and outside TomEE are the same. The only exception is `logFileDir`. For service containers running outside TomEE, set `logFileDir` to `$CORDYS_HOME/transactionlogs`.

# Chapter 6

# Managing Web gateway

AppWorks Platform is a Web-based platform and uses the HTTP protocol for communication. Every AppWorks Platform instance must have the Web server, TomEE, installed. The Web application server must be available before AppWorks Platform is installed. See the [AppWorks Platform Installation Guide](#) for best practices on configuring TomEE.

The Web server is used for communication from the front-end. It serves the user interface and provides access to the Platform and application services.

The Web gateway is the point of contact for the Platform and external applications. It receives HTTP requests, which contain SOAP messages and forwards these to a service container. When a SOAP message is received, the Web gateway routes the message to a specific Service Group. After the SOAP message is handled, the response is sent back to the caller. For information on the Web gateway and its applications, see [AppWorks Platform Web gateway](#).

The Web gateway receives a number of requests. There is statistical information that is available to understand the performance of the Web gateway. You can monitor the Web gateway using the [Monitoring Web gateway](#) tool.

When a request is sent to the Web gateway, it forwards the request to the appropriate service group. The client who sent the request, waits for the response from the Web gateway. If the Web gateway does not receive a response, it must inform the client. A time must be set to ensure that the Web gateway does not wait indefinitely to receive the response. For more information on configuring the Web gateway time-out, see [Configuring Web gateway timeout](#).

SOAP requests are XML code. When these requests are not properly formatted, the application cannot work with the request and the request processing fails. To avoid the time spent in processing an ill-formatted request, AppWorks Platform provides a mechanism to validate the SOAP requests and responses at the Web gateway itself before they are processed by service containers. SOAP requests coming into the Web gateway can be validated to ensure that they follow the SOAP 1.1 protocol. Similarly, the payload, which is part of the SOAP request excluding the SOAP protocol, in the SOAP request too can be validated to ensure that it follows WSDL Web service operation standards.

**Managing the Web Gateway includes the following tasks:**

- [Enabling payload validation at gateway \(deprecated\)](#)
- [Enabling SOAP protocol validation at gateway](#)

- [Enabling request validation at gateway](#)
- [Enabling response validation at gateway](#)

## Testing Web Gateway

You can also test the performance of the Web gateway by sending it multiple requests. To do this, AppWorks Platform provides the Web Gateway testing tool. See [Using Test Web Gateway](#) in the *AppWorks Platform Advanced Development Guide*.

Web gateway can also be used for resolving SAML artifacts with user credentials within AppWorks Platform. For information on SAML artifacts, see [Using SAML artifacts](#). To configure the Web gateway with security options, see [Managing gateway with security options](#).

## Monitoring Web gateway

Monitoring Web gateway is an important task in managing a Web gateway. This is essential for measuring the performance of the Web gateway. The Web Gateway Monitor window contains various statistics associated with the Web gateway.

The gateway caches LDAP information for better performance. You can view LDAP cache statistics, such as total number of requests, number of misses, number of hits, number of flushes, and number of cached results.

1. On the **Welcome** page > **My Applications**, click  (Web Gateway Monitor).  
The Web Gateway Monitor details page opens and displays the following statistics that aid in monitoring:

- Gateway Statistics

Field name	Description
Total requests	Total number of requests passed through the gateway.
Total processing time	Total time taken for processing the requests.
Average processing time per request	Average time taken to process a single request.
Total back end wait time	Time delay while waiting for a response.
Invalid messages	Total number of invalid messages passed through the gateway.
Unknown users	Total number of unknown users accessing the gateway.
Invalid organization requests	Total number of invalid organization requests to the gateway.
Messages sent to Server	Total number of messages sent to the gateway.
Response from Server	Total number of replies received from the gateway.
Timed out requests	Total number of timed-out requests. that is replies not received due to time out.
LDAP errors	Total number of LDAP errors encountered.
Errors	Total number of errors that occurred.
Refresh List	Refresh the fields in the Gateway Statistics table.
Clear Statistics	Click <b>Clear Statistics</b> to clear the fields in the Gateway Statistics table.

- For LDAP Cache Statistics:

Field name	Description
Total requests	Number of requests to the LDAP.
Number of hits	Number of successful queries to the LDAP.
Number of misses	Number of queries that did not yield a successful response from the LDAP.
Number of flushes	Number of times the LDAP cache was flushed.
Number of cached results	Number of results from the LDAP that are cached at the Web gateway.
Clear LDAP cache	Clear the contents of the LDAP cache stored at the Web gateway.

## Configuring Web gateway timeout

Timeout indicates the time the Web server must wait for a response from the application. The timeout value is specified in milliseconds. The default is 30000 milliseconds.

1. Click Start > Programs > AppWorks Platform > <Instance Name> > Tools > Management Console.  
The Management Console window is displayed.
2. Click Platform Properties.  
The Platform Properties window is displayed.
3. Add the following property:  
`com.eibus.web.gateway.timeout = <value>`.
4. Restart the Web server.

The Web Gateway timeout is configured.

## Enabling payload validation at gateway (deprecated)

Payload is validated at the gateway to ensure that the payload of the SOAP request coming into the gateway is according to the Web Services Description Language (WSDL) Web service operation.

**Note:** This topic describes the procedure for enabling only the validation of the payload. This validation excludes the validation of the SOAP protocol. The property that must be enabled for validating payload is deprecated, and the following new properties are added:

- To validate only the SOAP protocol in a message, enable SOAP validation property. For information, see [Enabling SOAP protocol validation at gateway](#).
- To validate the SOAP protocol and the request, enable the request validation property. For information, see [Enabling request validation at gateway](#).
- To validate the SOAP protocol, the request, and the response, enable the response validation property. For information, see [Enabling response validation at gateway](#).

When a SOAP request is sent to the Web gateway, the request is forwarded to the appropriate service group using the namespace in the request. To process the request, the Enterprise Service Bus (ESB) layer checks for WSDL with the operation name from LDAP. If the WSDL is found, it is returned for processing. If the element name and the operation name are not the same in WSDL, ESB cannot validate it. All WSDLs in AppWorks Platform contain an element name in the WSDL schema, which is the same as the operation name. ESB can validate them. But a valid WSDL does not always need an element name in the WSDL schema, which is same as its operation name. If a WSDL request is formed using the element name and not the operation name, the ESB cannot find the WSDL implementation even though the SOAP request is valid. When payload validation is enabled and the element name is not the same as the operation name, the validation fails.

**To enable payload validation at the gateway:**

1. In the Management Console, set the `com.eibus.web.payload.validation` property as 'true' in the `wcp.properties` file.

**Note:** When the `com.eibus.web.payload.validation` property in the `wcp.properties` file is set to false, payload validation is disabled.

2. Restart the Web server.  
The payload validation at the gateway is enabled.

## Enabling SOAP protocol validation at gateway

SOAP requests coming in to the gateway are validated to ensure that they follow the SOAP 1.1 protocol.

**To enable SOAP protocol validation at gateway:**

1. Modify the following property in the **wcp.properties** file using the Management Console.

`cordys.gateway.protocol.validation=true`

For information on modifying a property, see [Managing AppWorks Platform properties](#).

**Note:** If the property is set to **false** in the **wcp.property** file, the SOAP protocol validation is disabled.

2. Restart the Web server.  
The SOAP validation at gateway is enabled.

**Note:** This validates the SOAP protocol part of the requests alone with respect to the SOAP 1.1 protocol. To validate SOAP protocol and the request, enable the request validation property. For more information, see [Enabling request validation at gateway](#). To validate SOAP protocol, the request, and the response, enable the response validation property. For more information, see [Enabling response validation at gateway](#).

## Enabling request validation at gateway

This property validates the request at the Web gateway, and ensures that the SOAP request transferred to the gateway is according to the WSDL Web service operation. The SOAP protocol is also validated using this property.

**To enable request validation:**

1. On the Management Console, set the `cordys.gateway.request.validation` property to True in the `wcp.properties` file. For more information on modifying a property, see [Managing AppWorks Platform properties](#).

**Note:** When the `cordys.gateway.request.validation` property in the `wcp.properties` file is set to False, the request and SOAP validation are disabled.

2. Restart the Web server.  
This enables the request validation at the gateway.

**Note:** To validate the SOAP protocol, request, and response, enable response validation at the gateway. For more information, see [Enabling response validation at gateway](#). Enable SOAP validation to validate only the SOAP protocol. See [Enabling SOAP protocol validation at gateway](#) for more information.

## Enabling response validation at gateway

Response is validated at the Web gateway to ensure that the response of the SOAP request coming to the gateway is according to the WSDL Web service operation. The SOAP protocol and the SOAP request are also validated. Therefore, when the SOAP response is validated, the request is validated as well.

1. On the Management Console, set the `cordys.gateway.response.validation` property to **Ttrue** in the `wcp.properties` file. For more information on modifying a property, see [Managing AppWorks Platform properties](#).

**Note:** When the `cordys.gateway.response.validation` property in the `wcp.properties` file is set to False, response validation is disabled.

2. Restart the Web server.

The response validation at the gateway is enabled.

**Note:** This validates the SOAP protocol, request, and the response. To validate SOAP protocol and the request, enable request validation at the gateway. For more information, see . To only validate the SOAP protocol, enable SOAP validation at the gateway. See [Enabling SOAP protocol validation at gateway](#) for more information.

# Chapter 7

# Managing applications

AppWorks Platform provides an application development environment to build, package, and deploy applications. These software packages are termed as application packages and are packaged in the `.cap` format. An application package comprises many artifacts such as business process models, forms, Web services, and metadata, which define its business logic essential for building an enterprise application.

During design time, these artifacts are created and grouped as application content. To use the same content in a deployment environment, it is packaged and distributed as an application package (`.cap`). Deploying an application package refers to installing the package on the target computer and using the content for various business needs.

**Note:** A system administrator is responsible for end-to-end management of these application packages.

**Note:** Creating packages in the `.isvp` format has been deprecated. All content is packaged only in the application package (`.cap`) format. Applications packed in `.isvp` format cannot be deployed from version 16.1 onward. However, the applications that are already deployed in `.isvp` format continue to work as usual.

The applications provided by AppWorks Platform are available with certain roles and Web service interfaces apart from other content types, for your use. You can use these applications to create the artifacts required to build your own Web applications. AppWorks Platform also provides support for translating the Web applications you build, and customizing them for a global audience. In AppWorks Platform, you can define various languages to translate your Web applications and specify the contexts in which translations must be provided.

For applications that are packed in the `.cap` format, you can access the Application Deployer from the Welcome page of AppWorks Platform, and perform the following tasks:

- View application package status and details
- Manage application packages in a primary distributed setup
- Deploy applications
- Undeploy applications
- Redeploy incomplete applications
- Upgrade applications
- Roll back application packages

You can also perform these tasks in the silent mode using Ant tasks. AppWorks Platform provides several Ant task utilities to deploy and undeploy application packages, and create database configurations and service containers. See [Application Package Deployment utility](#).

**Note:** Ant tasks support only application packages of type .cap to deploy in the silent mode.

An administrator can continuously monitor the runtime behavior of the system and manage it through Java Management extensions (JMX). These extensions monitor the performance of the runtime objects, such as service containers of an application.

- Translating text to a target language (See the *AppWorks Platform Advanced Development Guide*)
- [Managing operations through Java Management Extensions](#)

## Working with application deployer

In the Application Deployer, you can view the status of application packages and other details such as name of the package, its version, the nodes on which it is deployed and so on.

### Using application deployer

#### To use the application deployer:

1. On the Welcome page > My Applications, click  (Application Deployer). A Deployment Overview page appears, displaying the status and applications in the Application List pane on the right-side.
2. By default, the applications that have the status 'New' will be shown.
3. Select the desired status from the overview page and click to view the list of applications with that status.

#### Types of status

- New : An application package will be in 'New' state, if the package is either not deployed on any node available in the cluster or if a higher version of the package is available for upgrade.
- Deployed: An application package will be in 'Deployed' state, if the same version of the package is deployed on all the nodes available in the cluster.
- Incomplete: An application package will be in 'Incomplete' state if:
  - The deployment of a package fails on one of the available node
  - The versions deployed on the nodes is inconsistent.
- Partial: An application package will be in "Partial" state when the package of same version is deployed successfully only in some of the nodes.

## Viewing application package details

From the Application List pane, click on the required application package. Alternatively, you can right-click on the application package and select  (Details). The details of the application are displayed in the adjacent section comprising Application Summary and Dependencies tabs:

The Application Summary tab of the application provides the following information.

Field	Description
Name	Refers to the name of the Application.
Vendor	Vendor of the Application.
Description	Describes the functionality of the application.
Version	Version of the application.
Build Number	Build number of application.
Packaged At	Indicates the timestamp at which the application is packaged.

The Dependencies tab contains information related applications that are dependent on the selected applications such as Name and Version of the application.

The information related to the deployment on the nodes will be displayed below the tab pages.

Field	Description
Node	Refers to the name of the system which the application is loaded.
Version	Indicates the version of the application.
Build	Build number of application.
Installed At	Indicates the timestamp at which the application is installed.
Deployed By	Refers to the DN of the administrator who deployed the application.

## Deploying applications

**Important:** Application deployment has strict prerequisites to ensure a smooth deployment.

### Before you begin:

To deploy an application, ensure that you meet the following requirements:

- You must have the role of systemAdmin to deploy or upgrade an application.
- The System Organization must be your default organization to deploy or upgrade an application.
- Before upgrading an application, it is recommended to manually take a backup of the LDAP content, database, and the filesystem related to the application.
- To deploy an application with the system environment variables, DLL registration, or virtual folder creation as content on a Windows computer, you must change the access rights. For more information, see [Loading an application with system environment variables content](#)
- If the application to be deployed has a dependency on other applications, ensure that these applications are deployed or available for the deployment.
- Ensure that the version of the application you are deploying is the same as the version of the runtime reference. A version mismatch can potentially result in the deployment failure.
- To deploy an application that is signed by a valid certificate.
- The required Service Containers are started and running. Deployment of artifacts requires the related Service Containers to be available.
- Disable payload validation if it is enabled. See [Disabling payload validation](#).

**Tip:** Instead of using the Application Deployer Wizard as shown in the following procedure, you can copy the application packages to install to the <AppWorks Platform\_installdir>/capcontent/packages folder.

### To deploy an application using Application Deployer:

1. On the Welcome page, click  (Application Deployer).  
The Deployment Overview page appears, displaying all the applications in Applications List.
2. Click **Browse**. If your application does not appear in the list; select the required application, click **OK** and do one of the following:
  - Click **Upload** to upload the application and deploy it later. The selected application is displayed in the Applications List. When ready, right-click the required application and select  **Deploy**. The Application Deployer wizard displays Selected Applications.
  - Click **Upload and Deploy** to upload and display the Application Deployer wizard.

**Tip:** If you are sure of the dependencies and the signature of the packages, you can select  **Express Deploy** to go directly to Application Summary.

3. Click , select the version of the package to deploy, and click **OK**.  
By default, the latest version is selected.
4. Click **Next**. Package Impact displays the list of actions to perform on the selected packages and their dependencies.

5. Click **Next** to display either the Application Verification page or the Impact Analysis page.

**Note:** The Application Signature Verification Status page displays when one or more applications fail signature verification. If the status of an application appears to be tampered with or unsigned, and you still want to proceed with the deployment, select the option for that application. Ensure that the security permissions to deploy are set to **Prompt** or **Allow**. To allow all the packages for deployment, you can choose **Select All**. Additionally, if you are deploying in a distributed setup and signature verification fails, then you must ensure one of the following:

- The application package is uploaded in the primary computer.
- The security permissions to deploy are set to **Prompt** or **Allow**.

**Note:** If you are not allowed to deploy the application, contact the system administrator and retrieve a relevant signed application. The signature verification of the applications is done.

7. Click **Next**, if necessary, to display Impact Analysis, which shows the artifact information of the first package. To view the artifact impact of other packages, choose the package from the list.

**Note:** In case of a cluster setup that has primary and secondary nodes, Impact Analysis gives an overview of which artifact is deployed on the primary node and which one is deployed on the secondary nodes.

8. If you have selected to deploy an MDM model, you must do the following in User Inputs:
  - Select the proxy user and the MDM proxy user from the drop-down list that are required to communicate with other applications on AppWorks Platform You may select an existing MDM Publisher service group or configure a new one:
    - Click  to select an existing MDM Publisher service group.
    - Click  to configure a service group. The Hub Publisher field is selected by default.

**Note:** The MDM Publisher and the MDM Service service groups that are associated with the MDM model must not be renamed after the model is deployed successfully.

- Click **Next**.

**Note:** If you are deploying an MDM model in a primary-distributed setup, it is adequate to deploy it in the primary computer alone. Select only the primary node and click **Next**.

9. Click **Next**. The Applications Summary page appears, displaying the actions that will be performed on the available nodes of the package. You can specify the following.
  - **Revert on Failure:** Select to revert the deployment action in case of failure. Note that if the application fails to install, the status of the application is displayed as incomplete on the Deployment Overview page. To complete the installation, see [Redeploying incomplete applications](#).

- **Timeout:** Specifies timeout value for the deployment (in minutes). This specifies the maximum amount of time to take when deploying an application. The client gets timed out after this period.
  - **Save as Template:** Enter a name in **Template Name** to reuse the deployment scenario later. The list of created templates appears on the Deployment Overview page. You can right-click a template to deploy, view, or delete the template. The list is empty if there are no templates created. For more information, see [Using custom templates](#).
10. Click **Deploy** to complete the process. The application is deployed using the Application Deployer task.
- Note:** If you are deploying a AppWorks Platform BAM MDM Model application package and AppWorks Platform is migrated from earlier versions to Cordys BOP 4.1 CU7.1 (and eventually to the later versions), run this database script:
- ```
<AppWorks_Platform_<br/>installdir>\components\bam\database\application\dropscripts\BAM_SYNC_<br/>TABLES_<<Database_Vendor>>.sql
```
- This script must be executed against the database configured for the BPM service container.
- Note:** You do not need to run this script if you directly installed Cordys BOP 4.1 CU7.1 and then migrated to Cordys BOP 4.2.

#### After you complete:

If you disabled payload validation for deploying an application using Application Deployer, you must enable payload validation. See [Enabling payload validation](#).

## Undeploying applications

You can undeploy an application using Application Deployer.

#### To undeploy an application using Application Deployment:

1. On the Welcome page, click  (Application Deployer).  
The Deployment Overview page appears, displaying the applications in the Application List pane.
  2. Click  (Deployed) to display the list of deployed applications.
  3. Right-click an application and select  **Undeploy**. The selected application and the corresponding dependencies are displayed.
- Note:** By default, **Undeploy the Dependent Applications** is selected. Clear the check box if you do not want to undeploy the dependent applications.
4. Click **Next**. The artifact impact for the first application is displayed. Select any package from the list to view its impact.

5. Click **Next**. The summary page appears, displaying the packages with a list of nodes on which the application will be undeployed.
6. Click **Undeploy** to undeploy the selected application.

The selected application is undeployed.

## Redeploying incomplete applications

While deploying an application, if Revert on Failure is not selected and the application fails to load completely, it is shown in Deployment Overview as Incomplete. Such applications can either be redeployed or undeployed.

### To redeploy an incomplete application:

1. On the Welcome page, click  (Application Deployer) to display Deployment Overview.
2. Click  (Incomplete).  
Applications that are not completely deployed are displayed in Applications List. If there are no incompletely deployed applications, the number is 0.
3. You can either:
  - Redeploy the application: Right-click the required application and select **Deploy** to display the Application Deployer wizard. By default, the current version of the package is displayed. Click **Next**. The application starts deploying again from where it stopped earlier.
  - Undeploy the application: Right-click the required application and select **Undeploy**. An Application Undeploy wizard is displayed. Click **Next** to undeploy the application.

**Note:** You can also rollback to a previous version of an incomplete package. In the Application Deployer wizard, click , select the version of the application package to roll back to, and click **Next**. The application is rolled back to the selected version.

Based on your selection, the applications are redeployed or undeployed or rolled back to a previous version of an incomplete package.

## Upgrading applications

**Note:** An application is displayed for upgrade only when its build or version number is greater than its existing version. You can view the status of the applications before deploying or upgrading them. Application packages of .isvp format cannot be upgraded with packages of .cap format. You have to undeploy the packages in the .isvp package and redeploy the packages in the .cap format.

You can view the status of the applications before deploying or upgrading them.

### To upgrade an application:

1. On the Welcome page, click  (Application Deployer).  
Deployment Overview displays all the applications in Application List.

2. Click **Browse** and select the required application to upgrade.
3. Click **OK** and do one of the following:
  - Click **Upload** to select and upload the application. After the application package is uploaded, click **New** to view the details of the application.
  - Click **Upload and Deploy** to upload and start the deployment process. The Application Deployer wizard is displayed. This option will directly start deploying application after uploading is done.
  - Note: Alternatively, you can copy the Application Packages to be installed to the <AppWorks Platform\_installdir>/capcontent/packages folder.
4. Right-click the required application and select  (**Deploy**).  
The Application Deployer wizard displays Package Impact.

**Important:** The remaining steps in this procedure are the same as when deploying applications. See [Deploying applications](#) starting with the Package Impact page.

When the procedure is complete, the application is upgraded in the Application Deployer.

If you encounter issues while upgrading or stop upgrading and go back to the earlier version, you can use the Rollback option. For more information, see [Rolling back application packages](#).

## AppWorks Platform application package

Applications can be built on AppWorks Platform and comprise various artifacts such as business process models, forms, schemas, Web services, and metadata, which define its business logic essential to build enterprise application. These applications can be packaged and can be deployed in other systems for reuse. The packaging formats for applications deployed on AppWorks Platform are Independent Software Vendor Package (ISVP) and AppWorks Platform application package. While the ISVP format, due to its drawbacks, has been deprecated, the application package with enhanced packaging capabilities is introduced.

Packaging applications using the application package format has the following benefits:

- Support for cluster-wide deployment: A AppWorks Platform installation typically consists of multiple systems, together forming a cluster. You can now deploy your packages on the cluster at one go. That is, you need not deploy on each individual node in the cluster separately.
- Support for recommencement of deployment on failure: The application package keeps track of the status of each artifact and in case of any failure reports it. After the correction is made, the deployment continues right from where it stopped, thus reducing time and resource overheads.
- Refined upgrade or deployment process: The application package keeps record of each artifact deployment. It knows, for each artifact, whether an application has to be

deployed on each node or not. It compares the artifacts in the application package file with the deployed artifacts, and loads only those artifacts that are new or changed.

- Impact analysis of the deployment: The application package provides a detailed analysis of the impact an application deployment has on the existing system. It provides a detailed report for each artifact and operation to be performed.
- Improved log file content: The application package log files clearly indicate the phase of the deployment or upgrade and list the relevant information in a way that is useful for administrators.

## Managing application packages in the Organization space

In AppWorks Platform, you can deploy application packages into runtime in the following spaces:

- Shared space (formerly known as ISV space): This space includes all the platform packages and custom packages that are intended to be accessible across all the organizations.
- Organization space: This space is unique for an organization and the packages deployed in this space are accessible only to the users of that organization.

Deploying application packages into an organization space enables the administrators to deploy customized versions of an application in specific organizations. These packages will be accessible only to the users of that organization. This is useful especially in the scenarios where a AppWorks Platform instance is used for deploying packages of different customers and also restrict the access of the application package only to few intended users. It also gives an additional advantage of managing the dependency on the Shared space; users can choose to continue with an older version of a package even when the package of similar version deployed into the Shared space is upgraded.

AppWorks Platform also allows you to have multiple organizations working on a single instance. It supports runtime multitenancy wherein all the artifacts are multitenant-aware at runtime. This ensures that each organization is independent of other organizations; changes made to artifacts in one organization will not impact other organizations that share the same artifacts. See Multitenancy in the *AppWorks Platform Overview Guide* for a detailed information.

### Note:

- You can deploy application packages of `.cap` format only in the Organization space. The application packages of the `.isvp` format cannot be deployed into Organization space.
- As a System Administrator, you can deploy application packages to an Organization space from the System organization. You need to have Administrator role in the target organization.

- As a Organization Administrator, you can deploy application packages to your organization space based on the authorization configured for that organization space by System Administrators. See [Authorize organization administrator to manage packages](#).
- The packages which are marked as deployable at 'Organization' level are deployable to the Organization space.
- Deployment of packages to System organization is not supported

See Setting properties of an application package in the *AppWorks Platform Advanced Development Guide* for information on configuring a package to be deployable at organization level.

## Managing application packages across spaces

You can select a version of a package to deploy or upgrade and rollback during deployment. The following scenarios explain the various options of managing different versions of packages across the Shared and Organization spaces.

### ***Viewing application packages deployed in the Organization space***

While managing application packages across organizations, you might need to have an overview of the packages and their versions deployed in a specific organization. You might also need to view the list of organizations where a specific application package version is deployed. As an administrator, AppWorks Platform enables you to have a complete picture of the application package deployment landscape and have a better control on the deployment or upgrade of dependent packages across the Shared and Organization spaces.

For instance, consider an application package A of version 1.0 is deployed in the Shared space and another application package B of version 1.0, that is dependent on the package A, is deployed in the Organization space. In this case, if application package A is ready to upgrade to version 2.0, it may have a significant impact on the dependent package B based on the usage of the application content and its change set. Before performing such upgrades, the following views enable the system administrators to see the impact of upgrade across organizations and take appropriate measures for the dependent packages.

**To view organization summary:**

1. Open Application Deployer.  
The deployment overview is displayed.
2. Click the **Summary** tab and click **Organizations** on the **Deployment Overview** section.
3. Select **All** or a set of organizations as required.
4. Select one of the options below to filter the application packages:
  - a. To view cluster status of all the packages, click the **Show All** option.
  - b. To filter a set of packages, click  adjacent to the **Show Selected** option.  
A modal dialog with a list of packages is displayed.
5. Click **View**, to view the details of the selected packages.  
A tabular view of the packages with the status details in the organization is displayed.
6. For the node level deployment status of the package in a particular organization, click on the version or the status link.  
Each column in the grid, represents an available node in the cluster. Each cell element displays the version of the package deployed on that node.

***Deploying application packages in the Organization space*****To deploy application packages:**

1. On the Welcome page, click  (Application Deployer ).  
The Deployment Overview page opens displaying all the applications in the Applications List pane on the right.
2. Click **Organization** in the **Space** section on the **Deployment Overview** panel.
3. Click .  
The **Organization List** dialog box opens.
4. Select the required organizations and click **OK**.  
The name of the selected organization is displayed in the output field.  
If multiple organizations are selected, the value in the output field is displayed as **Multiple Organizations**.

**Note:**

- The **Package Status** details are displayed only if you had selected single organization for deployment.  
The **Applications List** pane displays the packages related to that organization only.
  - The **Package Status** details are not displayed if you had selected multiple organizations for deployment.  
The **Application List** panel is populated with the application packages which are deployable into an Organization space.
5. Right-click the required applications and select **Deploy**.  
The Application Deployer wizard opens displaying the Selected **Applications** screen.

6. Click **Next**.

The Package Impact wizard opens displaying the list of actions that will be performed on the selected packages and their dependencies.

Based on the organization and the package selection, the deployment operation may change. The following table explains the possible deployment operation based on the package being selected and its current deployment status in an Organization space. It is applicable for all the packages across all the organizations selected.

| Package being selected | Current deployment status | Operation                                |
|------------------------|---------------------------|------------------------------------------|
| Package A 2.0          | Package A 1.0 is deployed | Upgrade                                  |
| Package A 1.0          | Package A 2.0 is deployed | Rollback                                 |
| Package A 2.0          | Package A 2.0 is deployed | No Action. (The package will be skipped) |

7. Click **Next**.

The **Application Signature Verification Status** page is displayed.

If the status of any of the selected application package appears to be tampered with or unsigned, see [Security settings and options](#) to continue with deployment.

8. Click **Next**.

The **Impact Analysis** page is displayed.

By default, the artifact information of the first package and the organization is displayed. To view the artifact impact of other packages in other organizations, select the required package and organization from the list.

**Note:** In case of a cluster setup, which has primary and secondary nodes, the impact gives an overview of the artifacts deployed on the primary node and the artifacts deployed on the secondary nodes.

9. If you have selected to deploy an MDM model, you must do the following in the **User Inputs** page:

- In **Select Organization**, select the organization.
  - In **Select Package To Configure Details**, select the package.
  - Select the proxy user and the MDM proxy user required to communicate with other applications on AppWorks Platform.
  - You may select an existing MDM Publisher service group or configure a new one:
    - Click  to select an existing MDM Publisher service group.
    - Click  to configure a service group.
- The Hub Publisher field is selected by default. See Fields for configuring MDM service groups in the *AppWorks Platform Advanced Development Guide* for information on the fields.

**Note:** The MDM Publisher and the MDM Service service groups that are associated with the MDM model must not be renamed after the model is deployed successfully.

- e. Click **Next**.
- f. Configure service groups in all the organizations for a package. Ensure that the MDM Repository database is unique per organization for a package.

**Note:** If you are deploying an MDM model in a primary-distributed setup, it is adequate to deploy it in the primary computer alone.

Select the primary node alone in the displayed screen and click **Next**.

10. Click **Next**.

The **Applications Summary page** opens displaying the actions that will be performed on the available nodes of the package for each selected organization.

11. You can do the following in the **Applications Summary** page.

- a. Select the **Revert on failure** option to revert the deployment action in case of failure.

**Note:** If the application fails to install, the status of the application is displayed as **Incomplete** on the **Deployment Overview** page. See [Redeploying incomplete applications](#) for information on completing the deployment.

- b. Specify the timeout value for the deployment in **Timeout (in minutes)**.  
This specifies the time to be taken for the application deployment.  
The client will be timed out after this period.

12. Click **Deploy** to complete the process.

The deployment progress is displayed for each package and in each organization space.

The applications are deployed in the organizations using Application Deployer.

### ***Undeploying application packages in the Organization space***

#### **To undeploy application packages:**

1. On the Welcome page, click  (Application Deployer).  
The Deployment Overview page opens displaying all the applications in the Applications List pane on the right.
2. Click **Organization** in the **Space** section on the **Deployment Overview** panel.
3. Click .  
The **Organization List** dialog box opens.
4. Select the required organizations and click **OK**.  
The name of the selected organization is displayed in the output field.  
If multiple organizations are selected, the value in the output field is displayed as **Multiple Organizations**.

**Note:**

- a. The **Package Status** details is displayed only if you had selected single organization for deployment.  
The **Applications List** pane displays the packages related to that organization only.
  - b. The **Package Status** details is not displayed if you select multiple organizations for deployment.  
The Application List panel is populated with the application packages which are deployable into an Organization Space.
5. Right-click the required applications and select **Undeploy**.  
The Application Deployer wizard opens displaying the **Selected Applications** page and their dependent packages.
6. Click **Next**.  
The Impact Analysis page is displayed.  
By default, the artifact information of the first package and the organization is displayed. To view the artifact impact of other packages in other organizations, select the required package and organization from the list.
7. Click **Next** to continue.  
The Applications Summary page opens displaying the undeploy actions that will be performed on the available nodes of the package for each organization selected.
8. Click **Undeploy** to complete the process.  
The undeployment progress is displayed for each package and in each Organization space.

The applications are undeployed from the organizations using Application Deployer.

## Dependencies between packages across spaces

In a typical application development lifecycle, multiple people are working on multiple projects simultaneously. This mode of development enforces dependencies across the projects and the application packages being developed. In AppWorks Platform, an application package can depend on multiple application packages. The package dependencies can also be deployed across different spaces (Shared and Organization). A package deployed in the Organization space can only depend on the set of packages deployed in both the current Organization space and the Shared space.

**Note:**

- Packages deployed in the Shared space cannot depend on packages deployed in an Organization space.
- Packages deployed in Organization A cannot depend on packages deployed in Organization B.
- Packages deployed in the Shared space cannot be undeployed if there are any dependent packages in the Organization space.

Also, note that at any point of time, deployment is allowed in one space. You can either deploy to Organization space or to Shared space but not both at a time. This is applicable for dependent packages as well.

Therefore, consider a scenario where a package A with version 1.0 is deployed in the Shared space. Now, the package A with version 2.0 is also uploaded. When a package B with version 1.0 which depends on package A and if the package B is selected to be deployed to the Organization X, the package A will be its dependent package and since version 2.0 is available, will be considered as upgrade. However, since the selected space is different (Organization, in this case), the package A will not be upgraded to version 2.0

## Dealing with application package versions

You can deploy a version of an application package in multiple organizations. For example, an application package A of version 1.0 can be deployed in multiple organizations.

Application packages of different versions can also be deployed in different organizations. However, at any point of time, only one version of a package is allowed to be deployed in an Organization or Shared space.

**Note:** All artifacts are not organization aware. If an application package is being deployed in organization space which contains an artifact that does not support the organization level deployment, the deployment action fails with an appropriate error message. See [Runtime behavioral exceptions in the organization space](#) for more information.

## Authorize organization administrator to manage packages

By default, system administrators are allowed to deploy application packages into an organization space. However, by configuring authorization for a given organization space, the system administrator can delegate this task to manage packages to the organization administrator for that space. Based on the level of authorization, the organization administrators are able to manage application packages within their organization space.

There are two levels of authorization configuration options:

- Full access (packages in organization and shared space)
- Conditional access (selected packages in organization and shared space)

### ***Full access (packages in organization and shared space)***

This option will enable organization administrators to have full control on the application packages that are deployable to organization space. The packages within their organization space can be deployed, upgraded, or undeployed. When this option is configured, the organization administrators are able to perform the following tasks:

- Access to deploy packages that are available in the shared and organization space.
- Upload and deploy packages into their organization space.
- Deploy, upgrade, or undeploy the packages in their organization space which are already deployed by the system administrator.

## Conditional access (selected packages in organization and shared space)

System administrators can configure this level of authorization option, to ensure access to specific set of packages available in the shared and organization space, to organization administrators in their organization space. Additionally, system administrators can give full access to packages in organization space and also authorize the organization administrators to manage a specific set of packages that are available in the Shared space.

**Note:** When Conditional Access level authorization is configured, the organization administrators are able to upload new packages into their organization space only when the additional configuration option Full Access to Organization space is selected. Otherwise, the option to upload is disabled.

### Configuring authorization for an organization

#### Before you begin:

- You must have a system administrator role in the System organization to configure authorization on packages.
- The packages which are marked as deployable at Organization level are permitted for configuring authorization.

#### To configure authorization for an organization:

1. Go to System organization and on the Welcome page, click  (Application Deployer).
2. Click **Configure Authorization**.  
Authorization configuration page appears, displaying all the authorized organizations on the right pane.
3. Click **Add authorization** and click  adjacent to **Organization** in the **Set Authorization** section.  
A dialog box with a list of organizations is displayed.
4. Select the required organization from the list and click **OK**.
5. Select the type of authorization:
  - a. **Full Access** - Provides complete access to the packages in the organization space and organization-aware packages available in the shared space.
  - b. **Conditional Access** - Provides access to the selected set of packages within the organization and shared space.
    - Select **Full Access to packages in Organization space** to provide complete access to all the packages within the organization space.
    - Click **Select Packages** to authorize specific set of packages.  
A dialog box with a list of packages is displayed.
    - Select the set of packages to be authorized and click **OK**.  
The selected set of packages are listed in the **Authorized Packages** table.
6. Click **Authorize** to save the authorization configuration.

## Runtime behavioral exceptions in the organization space

All artifacts are not organization aware. If an application package is being deployed in the Organization space, which contains an artifact that does not support organization level deployment, the deployment action will fail with an appropriate error message.

### ***Documents that do not support deployment to the Organization space***

Ensure that you structure your CWS project content and package the content accordingly.

The following artifacts do not support deployment to the Organization space:

- Object Template
- File
- Jar files

### ***Documents and exceptions***

The following table lists the runtime exceptions that may arise per artifact in the Organization space:

| Document                      | Exception scenario                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|-------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| User Interface Delivery Model | <ul style="list-style-type: none"><li>■ If tasks were already delivered to the Inbox, then the delivery model used for task delivery will not be deleted when the related application package is uninstalled. In cases where an application package is uninstalled from the Organization space and deployed newly into the Shared space, you may notice that new tasks being delivered to the organization still use the delivery model remaining in the organization.</li></ul>                                                                       |
| Rule                          | <ul style="list-style-type: none"><li>■ If a rule is deployed in the Shared space as well as in the Organization space and the rule deployed in the Organization space expires, when the rule is triggered from this organization, the rule present in the organization is ignored as it is expired and the one present in the Shared space is triggered.</li><li>■ In the Deployed Rules user interface, even if a rule published to the Shared space is explicitly selected for execution, the organization level rule is always executed.</li></ul> |
| File                          | <ul style="list-style-type: none"><li>■ Files can only be deployed to the Shared space and not to the Organization space.</li></ul>                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Java Archive                  | <ul style="list-style-type: none"><li>■ The Java archive files (.jar), except Ws-AppServer application jars, can only be deployed to the Shared space and not to the Organization space. Only Ws-AppServer based application jars</li></ul>                                                                                                                                                                                                                                                                                                            |

| <b>Document</b>                 | <b>Exception scenario</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|---------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                 | <p>can be packaged and deployed to the Shared space and Organization space. If application contains jars other than Ws-AppServer based jars, for example external or third-party jars, they cannot be packaged and deployed to the Organization space.</p>                                                                                                                                                                                                                                                                                                                                                   |
| MDM Model                       | <ul style="list-style-type: none"> <li>■ You cannot deploy the same MDM application model to the Shared space as other tenants are pointing to the same application databases. In scenarios where there is a need to deploy the same MDM application model for multiple tenants, ensure that the application databases are different for each deployment.</li> <li>■ You cannot perform concurrent upload or download operations in multiple organizations on an MDM Model deployed in the Shared space.</li> </ul>                                                                                          |
| Schedule                        | <ul style="list-style-type: none"> <li>■ Schedules, which are deployed into the Shared space through ISVP or CAP cannot be instantiated and consumed in more than one organization. Schedules available in the organizations can be instantiated by the users of that organization only. If the same schedule is required in multiple organizations, then it must be packaged and deployed in those organizations.</li> <li>Schedules deployed to specific organizations can be instantiated by the users of that organization.</li> </ul>                                                                   |
| KPI                             | <ul style="list-style-type: none"> <li>■ The trend for a KPI that is deployed in Shared space is possible to be worked upon only in one organization. As the deployment of the schedules at the shared level can only be triggered in one organization, users must manually instantiate the schedule in the organization in which the trend generated by the KPI is required to be viewed.</li> <li>■ After deploying the KPIs containing the e-mail models to the organization space, the e-mail model Web services must be attached to the notification service container in that organization.</li> </ul> |
| Business Event Response (BER)   | <ul style="list-style-type: none"> <li>■ After deploying the BERs containing the e-mail models to the Organization space, the e-mail model Web service must be attached to the notification service container in that organization.</li> </ul>                                                                                                                                                                                                                                                                                                                                                               |
| Process Monitoring Object (PMO) | <ul style="list-style-type: none"> <li>■ When ever a BPM model is modified and deployed into an organization, the related PMOs must be also modified and deployed into the organization.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                          |

| <b>Document</b>       | <b>Exception scenario</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|-----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Web Service Interface | <ul style="list-style-type: none"> <li>■ Two different CAPs having two different Web service interfaces with the same name cannot be deployed in the same organization. The loading of the second CAP fails to insert the Web service interface as another Web service interface with the same name is already available in LDAP (deployed through the first CAP).</li> <li>■ If a Web service interface with the same name exists in the organization space and shared space, the one in the organization space is considered first. In this case, the runtime referenced Web service interface name matches that of the existing Web service interface in the organization space. The Web service interface can be deployed to the organization space through a CAP or published from CWS design-time.</li> </ul> |
| Business Calendar     | <ul style="list-style-type: none"> <li>■ When a customized business calendar is deployed into an organization, the reload of the existing runtime reference over the artifact does not indicate that the binding is now updated to what is deployed to the organization. This behavior does not impact the runtime execution behavior. When executed, if the organization version is available with the name being referred, then only the organization version is considered for execution.</li> </ul>                                                                                                                                                                                                                                                                                                             |

### Other exceptions

Following are the other issues that are not specifically related to a document but can be associated to an execution behavior. These are mentioned component-wise to help you relate them easily.

| <b>Component</b>      | <b>Exception scenario</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|-----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| BPM                   | <ul style="list-style-type: none"> <li>■ Process documentation configuration is not organization specific. See Documenting business process models in the <i>AppWorks Platform Advanced Development Guide</i>.</li> <li>■ Calls to the custom Java methods using XPaths in the BPM models do not use organization-specific jar files.</li> <li>■ Direct instantiation of process models from Deployed Process Models, which are deployed directly into an Organization space and which consume Web services that are deployed into the Shared space will not work as expected. Instantiation of such models by any other means will work.</li> </ul> |
| Case Instance Manager | <ul style="list-style-type: none"> <li>■ There is only one case model shown even though the model is deployed to both the Shared and the Organization spaces.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |

| Component             | Exception scenario                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|-----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                       | <ul style="list-style-type: none"> <li>▪ Instances in the detail view are only shown for a single case model (though instances exists for the system model and the organization model).</li> </ul>                                                                                                                                                                                                                                                   |
| Schedule Manager      | <ul style="list-style-type: none"> <li>▪ Schedules deployed to both Shared and organization of a single schedule are shown in the <b>Inactive Schedules</b> tab of the Schedule Manager.</li> <li>▪ For the scenario mentioned above, if the schedule of the Organization space is deployed, then the schedule deployed to the Shared space is also hidden.</li> </ul>                                                                               |
| BAM                   | <ul style="list-style-type: none"> <li>▪ The BAM service container is required in the organization to work with BAM.</li> <li>▪ All the BAM service containers in the AppWorks Platform instance must always point to the same repository.</li> <li>▪ BAM content must not be spread between the Shared space and the Organization space. To work with BAM in an organization, content must be installed completely in that organization.</li> </ul> |
| Web Service Interface | <ul style="list-style-type: none"> <li>▪ It is possible to deploy the Web service interface or method set on the organization level, but it cannot be linked to a service group, which is defined at the system level.</li> </ul>                                                                                                                                                                                                                    |

## Using custom templates

While deploying applications, you can choose to save a set of applications to be deployed as a template. These templates can be used to deploy applications in the same computer or other computer too.

**Note:** You cannot create custom templates for packages being deployed in the Organization space.

### To deploy applications in a custom template on the same instance:

1. On the Welcome page, click  (Application Deployer).  
The Deployment Overview page opens displaying the templates under **Custom Templates**.
2. Select the relevant template, right-click on the template, and then click **Express Deploy**.  
The applications which are part of the template are deployed.

**To deploy applications in a custom template on another instance:**

1. Open the instance where you saved the templates.
2. On the Welcome page, click  (Application Deployer).  
The Deployment Overview page opens displaying the templates under **Custom Templates**.
3. Right-click the relevant template, click **View**.  
The Template View window opens displaying the XML of the template.
4. Copy the XML to a file on the computer to be deployed. For example: test.xml.  
The template is created and you can use this as an input file for your command-line deployment.
5. Optional. If required, you can create a user input file and save it to a location on the computer to be deployed.  
This step is needed if you have to provide additional inputs for your applications in the template while deploying. For example, if your application contains MDM content, then you may need to provide a different database details. In that case, you must create the user input file with all the relevant database details.  
You can see AppWorks Platform zip folder > captemplates folder > CAPUserInput file as a reference.
6. Ensure that the relevant applications in the template are present in the `<AppWorks Platform_installdir>/capcontent/packages` folder.
7. From the extracted zip folder, open the `cap-silent-install.properties` file in a text editor. Modify the following properties according to the configuration of the computer.
  - `cordys.cap.silent.template.inputfile` refers to the file path which contains template of the required applications to be deployed.
  - `cordys.cap.silent.userinputfile` refers to the file path which contains user inputs for the applications.
  - `cordys.cap.silent.deployment.server` refers to the name of the computer on which AppWorks Platform baseline installation is done.
  - `cordys.cap.silent.deployment.port` refers to the port number of the website on which AppWorks Platform baseline installation is done.
  - `cordys.cap.silent.deployment.failOnError` decides whether to stop or continue the execution of applications if one of them has failed to deploy or upgrade. This is an optional property and the default value is `false`.
8. Set the environment variable `CORDYS_HOME` pointing to the `<AppWorks Platform_installdir>`.
9. Set `PATH=%PATH%;%CORDYS_HOME%/lib`.
10. Set `CLASSPATH=%CLASSPATH%;%CORDYS_HOME%/cordyscp.jar`.
11. Run the following java command:  
`java -Duser.name=<ldapuser> -cp <AppWorks Platform_installdir>\cordyscp.jar`

```
com.cordys.cap.tool.silentdeployment.SilentCAPDeployer <Location of cap-silent-install.properties file>  
where ldapuser is the user created during the AppWorks Platform baseline deployment.
```

The applications are deployed using the custom template. The status of each application is displayed in the command prompt as it is being deployed.

## Managing application packages in a primary distributed setup

In a Primary-Distributed setup, two or more installations of AppWorks Platform share the same directory service (OpenText CARS) and distribute the load of service requests among themselves. This setup is preferred in high availability required environments. See *Appworks Platform High Availability Deployment Guide*. It installs only the runtime environment required for the service containers to be configured and executed on a system. This kind of setup is useful for load balancing.

**Note:**

You must ensure to keep the Primary-Distributed cluster setup consistent while managing the application packages. This means that all the packages of a specific version and their dependent packages must be deployed in all the nodes of the cluster. The runtime behavior of the distributed setup is improper if the consistency is not maintained.

## Deploying application packages in a Primary-Distributed setup

**To deploy application packages in a Primary-Distributed setup:**

1. Open Application Deployer.
2. Click **New** to view the list of the application packages to be deployed.
3. Right-click the selected application packages and select **Deploy**.  
The Application Deployer wizard opens displaying the **Application Packages and their Dependencies** dialog box.
4. Click **Next**.  
The Application Signature Verification Status page is displayed.

**Note:** This page is displayed only if one or more applications fail the signature verification. If the status of an application appears tampered or unsigned, to proceed with the deployment, select the **Allow to Proceed** option. Ensure that the security permissions to deploy are set to **Prompt** or **Allow**. See [Security settings and options](#). If the signature verification fails with the error message 'File not found', then one of the following must be ensured:

- The application package is uploaded in the primary computer.
- The security permissions to deploy are set to **Prompt** or **Allow**.

**Note:** If you are not allowed to deploy the application, contact the system administrator and obtain a relevant signed application.

5. Click **Next**.

If the **Show Impact Analysis** option is selected in the **Applications and their Dependencies** dialog box, the **Impact Analysis of Selected Applications** dialog box is displayed.

It provides information about the artifact and the impact (Create, Update, Move, Rename, and Delete) the deploy operation has on the artifact.

The impact provides an overview of which artifacts will be deployed on the primary node and secondary node respectively.

6. Click **Next**.

A list of the nodes configured in the Primary-Distributed setup is displayed.

7. Select the nodes to deploy the application packages and click **Next**.

The Applications Summary page opens displaying the list of the nodes on which the application will be deployed. See [Deploying applications](#) for more detailed actions to complete the deployment.

## Cluster summary view

Sometimes, an application package may fail to deploy on a node of the Primary-Distributed setup (cluster) and succeed on the other node. It is also possible that while upgrading an application package to an higher version, one of the nodes in a cluster might not be available. This forces the package to get into an inconsistent situation. While it is important to have a consistent deployment of application packages in a cluster, having a clear overview of the consistent deployment becomes even more apparent.

To have a clear overview of the cluster setup, AppWorks Platform, provides **Cluster summary** view. This view helps the system administrators to have a clear view of the packages, it's versions and the status of the deployment across the cluster. The cluster wide status of the packages is calculated by summarizing the status of a package on all the available nodes of the cluster.

Looking at this view, the administrators can decide if any further actions needs to be performed to bring the cluster setup to a consistent state. One can define filters to short list set of packages with specific status.

### Types of status

The cluster wide status of the packages is explained below with a 3-node cluster setup.

**Deployed:** An application package is in the **Deployed** state in a cluster only when the package with same version is successfully deployed in all the available 3 nodes.

**Partial:** An application package is in the **Partial** state when the package of same version is deployed successfully in 2 of the 3 available nodes and is yet to be deployed in the 3rd available node.

- A package successfully deployed in all the 3 available nodes can still get into partial state when the package is undeployed from one of the available node. This may happen when some of the nodes are not available during undeployment.
- After successfully deploying a package of same version in a 2-node cluster, a package can get the partial status when a new node is added because package might not be deployed yet in the newly added node.

**Incomplete:** An application package is in the **Incomplete** state in a cluster when:

- The deployment or upgrade of a package fails on one of the available node.
- An higher or lower version of an application package is deployed in one of the available node.

### **Viewing cluster summary view**

**To view cluster summary:**

1. Open Application Deployer.
2. Switch to the **Summary** tab, which is located at left bottom of the page.
3. By default, **Shared** space type is selected.
4. Select **Show All** or **Show selected** options:
  - To filter a set of packages, click **zoom** button adjacent to the **Show Selected** option. A modal dialog box opens with a list of packages. You can also filter Platform packages from the selected list.
  - To view cluster status of all the packages, click the **Show All** option. This option also include all the platform packages.

**Note:** You can also filter the set of packages based on their cluster status as explained above. Select one or more filters (Deployed, Partial and Incomplete) check boxes to filter packages with the matching cluster status. By default, Partial and Incomplete check boxes are selected.

5. Click **View Summary**, to view the details of the selected packages. A tabular view of the packages with version and status details is displayed.
6. The **Status** column in the **Summary view** displays cluster status of the selected packages from all the available nodes.
7. Each column in the grid, represents an available node in the cluster. Each cell element displays the **version** of the package deployed on that node.
  - a. If a package is in incomplete state in a node, then the cell value of that package is labeled with **Incomplete** status along with it's version.
  - b. If a package is yet to be deployed in a available node, the cell value of that node for that package is marked as **x**.

- c. If a package in a node is having different version of a package compared to other nodes in the cluster, then the cell value displays the version of the package highlighted with **red** color.

**Tip:** You can use the **column chooser** option to show/hide a set of nodes from the tabular view to quickly compare packages across set of selected nodes.

## Exceptions and recommendations

The following table lists the exceptions and recommendations for managing application packages in a Primary-Distributed setup:

### *Deploying an incomplete package on a new node*

| Problem        | <b>A package is in an incomplete state in a node of a cluster. If the same package is being deployed on an another node and goes to an incomplete state, revert of such a package does not work.</b>                                                                                                                                                                                                                                                                                                                                                             |
|----------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Use case       | <ol style="list-style-type: none"> <li>1. Deploy a package A in node N1 of a cluster. If this package deployment fails for a valid reason such as if the required service container is not started or the database connection timed out, and so on, based on the <b>Revert on failure</b> option selected, the package is in the <b>Incomplete</b> state.</li> <li>2. Deploy the same package A on a different node N2 of the cluster. If this package also goes to the incomplete state, reverting the package from both the nodes does not succeed.</li> </ol> |
| Recommendation | If there is a package in an incomplete state on a node, do not deploy the same package of the same or higher version on another node until the issue is addressed on the former node.                                                                                                                                                                                                                                                                                                                                                                            |

### *Undeploying a base package from a node when the dependent package is deployed in an Organization space*

| Problem  | <b>A base package 'A' is deployed in the Shared space on both the nodes N1 and N2 of a two-node cluster. The dependent package 'B' is deployed in an Organization space on node N1. In this scenario, if the base package 'A' is being undeployed from the other node N2, the undeployment does not succeed.</b> |
|----------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Use case | <ol style="list-style-type: none"> <li>1. Deploy a package A in the Shared space on both the nodes N1 and N2 of a two-node cluster.</li> <li>2. Deploy a dependent package B in the Organization space only on node N1.</li> </ol>                                                                               |

|                       |                                                                                                                                                                                                                                                                                                                  |
|-----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Problem</b>        | <b>A base package 'A' is deployed in the Shared space on both the nodes N1 and N2 of a two-node cluster. The dependent package 'B' is deployed in an Organization space on node N1. In this scenario, if the base package 'A' is being undeployed from the other node N2, the undeployment does not succeed.</b> |
|                       | 3. Undeploy the base package A from node N2.<br>The undeployment does not succeed.                                                                                                                                                                                                                               |
| <b>Recommendation</b> | This issue occurs since the dependent package B is not deployed on both the nodes. Hence, it is recommended to undeploy package B from node N1 and then undeploy the base package A from node N2.                                                                                                                |

### ***Displaying an incorrect version in a cluster in the All Organization view***

|                            |                                                                                                                                                                                                                                                                                                                                                                                                              |
|----------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Problem Description</b> | <b>The 'All Organizations' view shows the incorrect package version as deployed for a specific organization.</b>                                                                                                                                                                                                                                                                                             |
| Use case                   | 1. Deploy package A with version 1.0 in the Organization space X on both the nodes N1 and N2 of a two-node cluster.<br>2. Upgrade with version 2.0 in the same Organization space X on node N1. The deployment fails and goes to an incomplete state.<br>3. Click the All Organizations view and select organization X. It displays version 2.0 as the deployed version though it is in an incomplete state. |
| Recommendation             | Fixing the reason for the upgrade failure on node N1 shows the correct version.                                                                                                                                                                                                                                                                                                                              |

### ***'File not found' error occurring at the signature verification step during deployment in a cluster***

|                            |                                                                                                                                                                                                                                                                                                                                                                       |
|----------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Problem Description</b> | <b>An application package fails at the signature verification step while deploying it in a cluster. This occurs when the package to be deployed is uploaded on to one of the distributed computers. During validation, since the signature verification is done on the primary computer, the system searches for the application package on the primary computer.</b> |
| Use case                   | 1. Upload the unsigned package A with version 1.0 on one of the distributed nodes N1.<br>The package is listed in the New list.<br>2. Select the package and deploy it.<br>3. The Signature Verification step fails, displaying the 'File not found' error.                                                                                                           |

|                            |                                                                                                                                                                                                                                                                                                                                                                       |
|----------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Problem Description</b> | <b>An application package fails at the signature verification step while deploying it in a cluster. This occurs when the package to be deployed is uploaded on to one of the distributed computers. During validation, since the signature verification is done on the primary computer, the system searches for the application package on the primary computer.</b> |
| Recommendation             | If Code Signing settings are set to <b>Prompt</b> or <b>Disallow</b> in Security Administration and signature verification is required, it is recommended to create the Security Administration service container on the distributed node.                                                                                                                            |

## Customizing application installer wizard

Some applications need some custom pages to be displayed during their installation. These pages are required to gather inputs necessary for installation. You can add such custom pages in the application installer. These pages are then displayed during the installation of the applications.

### To customize an application installer wizard:

1. Specify a custom page. See the [Specifying custom page](#) for more information.
2. Specify the operating system compatibility. See [Specifying operating system compatibility](#) for more information.

## Specifying custom page

An attribute URL needs to be added in the content tag to specify a custom page in the Application Installer wizard. The value of the URL attribute must be the Web address that has to be displayed.

For example,

```
<customcontent description="Custom Content" loader="CustomLoader"
url="/isv/newcontent.htm"/>
```

In this example, prior to the installation of the application, the /isv/newcontent.htm page must exist on the Web server of the machine where the application will be installed. There might be instances where the page that has to be displayed is not present on the Web server, instead it is bundled with the application (during creation). In such cases, the URL value must be preceded by isv\_temp\_dir.

For example,

```
<customcontent description="Custom Content" loader="CustomLoader" url="isv_temp_
dir/isv/newcontent.htm"/>
```

In this example, it is assumed that the file newcontent.htm is available in the application file at /isv/newcontent.htm. The developer of the custom page may also want to pass some information to the custom loader. This can be achieved by accessing the promptset node as follows:

```
var promptSetNode = window.system.data['promptset'];
```

The data that has to be passed to the custom loader has to be added to promptSetNode in the XML format. This data can be accessed by the custom loader using the getPromptsFromRequest() Web service operation. This Web service operation returns the promptset node along with the data appended.

The custom page can obtain the reference of the Application Installer wizard through

```
window.system.data['isvpLoaderWizard']
```

This can be used to customize the page in the wizard's context. For example, to set a heading for the custom page, the following code is used:

```
WizardId = window.system.data['isvpLoaderWizard'];
WizardId.heading = 'Custom Page Heading';
```

In the new JScript library framework, the heading of the custom page must be set using the setHeading() Web service operation.

## Specifying operating system compatibility

An attribute compatible needs to be added to the content tag to specify the operating system compatibility in the Application Installer wizard. The value of this attribute can be Windows or Linux.

For example,

```
<customcontent loader='CustomLoader' description='Custom Content'
compatible='windows'>
```

If the compatible attribute is not specified, it is assumed that the content type is compatible with both Linux and Windows.

While installing an application, that has incompatible content with the underlying operating system, an error message is displayed to indicate that the application cannot be installed as it contains incompatible content.

## Rolling back application packages

While upgrading an application package, if you face any issues or want to retain the earlier version, you can rollback the application package to the earlier version.

1. On the **Welcome** page, click  (Application Deployer).  
A Deployment Overview window opens displaying all the application packages in the Application List pane on the right-side.
2. Click **Browse** in the Deployment Overview page and select the required application package to be rolled back.
3. Right-click the application package and select  (Rollback).  
The Application Deployer wizard opens displaying the Package Impact screen.
4. Click , select the version of the application package to be rolled back from the list, and click **OK**.  
By default, the earlier version of the current deployed version is displayed in the list.
5. For the remaining process, see the steps provided for [Deploying applications](#).

The application is rolled back to the selected version.

**Note:** Rollback is not available for mandatory application packages.

## Installing an application in silent mode (deprecated)

**Note:** Installing an application in the silent mode is deprecated.

You can load applications in the silent mode on AppWorks Platform from the command prompt. This avoids navigating through the user interface screens.

### Before you begin:

Create a template with installation inputs. You can create this template during the application installation.

### To install an application in the silent mode:

1. Run the AppWorks Platform baseline installation. See the *AppWorks Platform Installation Guide*.

**Note:** Ensure that an OpenText AppWorks Platform (<instance name>) is running on your computer.

2. Access and retrieve the template with installation inputs, which is created with the key details.
3. To modify the template:
  - a. Modify the timeout in the root node <isvpcommand/> to set the timeout while installing applications.

- b. Set the `skip` attribute to `true` on the `<isvp/>` node for the applications to ignore during the installation.
  - c. Add or remove application entries if necessary. To install an application, copy the complete `<isvp>` node and modify its information.
  - d. Edit the `<url>` tag to indicate the computer name from where to install the required application.
4. In the command line, change the working directory to `<AppWorks Platform_installdir>\Lib` and run the following Java command:

```
java -cp <installdir>\cordyscp.jar com.cordys.isvp.tool.ISVPCommand -r  
<Path of the application loading template file>
```

The application package is installed in the silent mode.

## Verifying applications

You can verify applications using the *VerifyDeployedApplications* tool, which is a command-line tool to verify a signed application after the application is deployed at the client location. It verifies the artifacts of the file system against the artifacts that are shipped with the application.

### Before you begin:

Ensure that the Security Administration service container is started.

### To use the VerifyDeployedApplications tool:

1. Add the following libraries or JAR files to the classpath:
  - `facllib.jar`
  - `basicutil.jar`
  - `ccutil.jar`
  - `cordyscp.jar`
  - `eibxml.jar`
  - `esbclient.jar`
  - `esbserver.jar`
  - `ldap.jar`
  - `log4j.jar`
  - `managementlib.jar`
  - `webgateway.jar`
  - `xds.jar`

2. Run the following command:

```
java com.eibus.secadmin.tools.VerifyDeployedApplications -input "D:\inputXML.xml" -output "D:\output.xml".
```

Where:

- The **-input** parameter specifies the location of the applications to verify. If not specified, all the applications that are deployed in the AppWorks Platform environment are verified. See the sample code snippet of the input file.

```
<Applications>
    <Application type="isvp">Cordys Task Modeler Server.isvp</Application>
    <Application type="isvp">Cordys XMLStore.isvp</Application>
    <Application type="isvp">Cordys Common Content.isvp</Application>
    <Application type="isvp">User interface Modeler Server.isvp</Application>
    <Application type="isvp">Cordys Data Transformation.isvp</Application>
    <Application type="isvp">AppWorks Platform Documentation.isvp</Application>
</Applications>
```

- The **-output** parameter specifies the location where the verification result is published. If not specified, the output is written to the console. See the sample code snippet of the output file.

```
<Applications>
    <Application id="Cordys Task Modeler Server.isvp">
        <IsSigned
            xmlns="http://schemas.cordys.com/secadmin/verification/1.0"
            xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/"> true </IsSigned>
        <SignatureVerificationResult
            xmlns="http://schemas.cordys.com/secadmin/verification/1.0"
            xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/">
            <Status>Valid</Status>
        </SignatureVerificationResult>
        <IsSigningCertificateTrusted
            xmlns="http://schemas.cordys.com/secadmin/verification/1.0"
            xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/">
            <Status>true</Status>
        </IsSigningCertificateTrusted>
        <SigningCertificateValidation
            xmlns="http://schemas.cordys.com/secadmin/verification/1.0"
            xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/">
            <Status>true</Status>
        </SigningCertificateValidation>
    </Application>
</Applications>
```

In this example, Cordys Task Modeler Server.isvp is verified for the following criteria:

- The application is signed.
- Whether the file system artifacts in the computer and the artifacts in the application are same. If not, all the missing and modified files are listed.
- The signing certificate of the application is trusted.
- The signing certificate of the application has code signing key usage.
- The results are specified within the `Status` tags.

After the applications are verified, the results are published at the location specified for the output parameter.

# Chapter 8

# Managing licenses

Customers pay for software products per their usage in a specific period, which could be a month. Software products could be any of the following:

- AppWorks Platform product: AppWorks Platform owns this and customers pay AppWorks Platform for using the product.
- Product: an application package-based product built using the components of AppWorks Platform. Customers pay AppWorks Platform for using the AppWorks Platform components and the application package.

AppWorks Platform licensing enables invoicing by providing the necessary information through a usage report to bill customers for both AppWorks Platform and the application packages.

Every customer generates a usage report for a specific period (usually one month). Based on this report, customers are issued licenses to continue using AppWorks Platform. To continue using the application, it is mandatory for customers to send usage reports periodically and renew the necessary licenses.

A usage report is generated for every AppWorks Platform site, which has one or more AppWorks Platform products or application packages installed. This report contains the number of uniquely named users for that site.

On receipt of the usage report, a license key is sent to the customer. The AppWorks Platform License Service generates this license key, which enables the customer to continue using the application package.

License keys that are issued to customers of AppWorks Platform depend on the site registration details. License keys are of two types:

- Demo license key: This key is valid for 60 days from the date of issue. Customers can use the product for the prescribed period after which the license key expires.
- Operational license key - This key is valid for 60 days from the date of issue. To renew license keys, customers are required to send their usage report every month to the AppWorks Platform License Service. On receiving a valid usage report, the License Service issues new operational license keys to customers.

The license keys are encrypted and stored in LDAP.

You can manage AppWorks Platform licenses through the following tasks:

- Viewing the usage report
- Configuring the delivery mode of the usage report
- Viewing the master usage report
- Sending the master usage report to the licensing server
- Updating the license key automatically
- Modifying the license key
- Sending the usage report through proxy
- Modifying AppWorks Platform license information

For information about details required during installation, see [Installation role in licensing](#).

For information about the various contingencies related to licenses, see [Contingency management](#).

## Viewing the usage report

A usage report contains the number of unique named users for a site, which is calculated per application across all organizations and application installations. The report is sent to the AppWorks Platform License Service for invoicing. You can view this report using the License Manager.

### Before you begin:

- You must have the systemAdmin role.
- You must append the LicenseManagement Web service interface (available in the Cordys ESBServer Application) to the Monitor service group. By default, this Web service interface is attached when creating the Monitor service group.

### To view the usage report:

1. On the Welcome page, click  (License Manager).  
The License Manager window opens. The Installation Details and Contributor Details panes are displayed.
2. Go to **Contributor Details > Report Information**, and then select the **With actual user names** check box.  
Selecting this check box displays the actual names of the users in the usage report instead of the names of the computers using AppWorks Platform.
3. In the Report Information section, click **View**.  
The Usage Report page opens with a list of applications.
4. Click an application to view its application users.

## Configuring the delivery mode of the usage report

A usage report is necessary to evaluate the license renewal. Customers must send their usage report on a regular basis to the licensing server or to the master computer depending on the installation type. See [Installation types](#).

### Before you begin:

- You must have the systemAdmin role.

### To send the usage report to the licensing server:

1. On the Welcome page, click  (License Manager).  
The License Manager window opens. The Installation Details and Contributor Details panes are displayed.
2. Go to **Contributor Details > Report Information**, and then click **Auto Send**.  
The Configuration for Automatic Sending of Usage Report dialog box opens.
3. To send the usage report:

Automatically every 30 days	a. Select the <b>Yes</b> option, and then click <b>Yes</b> . The Master usage report is sent automatically every 30 days. In the Report Information section, Last Sent is updated with the date the report is sent.
Manually on a need basis	a. Select the <b>No</b> option, and then click <b>Yes</b> . b. In the Key Information section, click <b>Update</b> . A confirmation dialog box opens. c. Click <b>OK</b> .

## Configuring AppWorks Platform to send usage reports

The automation of usage reports occurs only for operational licenses of AppWorks Platform.

### Before you begin:

- You must have the system administrator's role to configure AppWorks Platform for sending the usage reports automatically to the licensing server.

### To send usage reports automatically:

1. On the Welcome, page, click  (License Manager) .  
The License Manager window opens.
2. Click **Auto Send** in Report Info of the Contributor Details page.  
The Configuration for Automatic Sending of Usage Report dialog box opens.

3. Select **Yes**, and then click **Save**.

AppWorks Platform is configured to generate and send the usage report automatically to the licensing server at regular intervals of time.

**Note:**

- The usage report or the contributor report is sent automatically every 30 days.
- The master usage report is sent automatically every 32 days.

## Viewing the master usage report

You can view a master usage report using the License Manager only for the multiple AppWorks Platform installation type. In this installation type, a site can have more than one AppWorks Platform installation. One of the installations is termed as the master, while the others are termed as contributors.

**Before you begin:**

- You must have the systemAdmin role.
- In the contributor computer, you must append the LicenseManagement Web service interface (available in the Cordys ESBServer application package) to the Monitor service group. By default, this Web service interface is attached when creating the Monitor service group.
- In the master computer, you must append the LicenseMaster Web service interface (available in the Cordys ESBServer application package) to the Monitor service group. By default, this Web service interface is attached when creating the Monitor service group.

**To view the master usage report:**

1. On the Welcome page, click  (License Manager).  
The License Manager window opens. The Installation Details and Contributor Details panes are displayed.
2. Go to **Master Details > Report Information**, and then select the **With actual user names** check box.  
Selecting this check box displays the actual names of the users in the usage report instead of the names of the computers using AppWorks Platform.
3. In the Report Information section, click **View**.  
The Usage Report - Master page opens with a list of applications.
4. Click an application to view its application users.

## Sending the master usage report to the licensing server

You can view the master usage report only for the multiple AppWorks Platform installation type. See [Installation types](#).

### Before you begin:

- You must have the systemAdmin role.
- You must configure the delivery mode of the usage report to ensure the master computer sends its usage report to itself. See [Configuring the delivery mode of the usage report](#).

### To send the master usage report to the licensing server:

1. On the Welcome page, click  (License Manager).  
The License Manager window opens. The Installation Details and Contributor Details panes are displayed.
2. Click the **Master Details** tab.  
The Master Details page appears.
3. In the Key Information section, click **Update**.  
In the Contributor Details section, Last Sent is updated with the date the report is sent.

## Updating the license key automatically

It is possible to update the license key automatically.

### Before you begin:

- You must have the systemAdmin role.

### To update the license key:

1. On the Welcome page, click  (License Manager).  
The License Manager window opens. The Installation Details and Contributor Details panes are displayed.
2. On the Contributor Details pane, in the Key Information section, click **Get**.  
The new key details appear in the Key Information section and are updated in LDAP.

## Modifying the license key

If the license key is not updated automatically, you can update it manually.

**Before you begin:**

- You must have the systemAdmin role.

**To modify the license key:**

1. On the Welcome page, click  (License Manager).  
The License Manager window opens. The Installation Details and Contributor Details panes are displayed.
2. On the Contributor Details pane, in the Key Information section, click **Change**.  
The License Key Information dialog box opens with the provision to change the key.
3. In Enter New Key, type the new key, and then click **OK**.  
The new key appears in the Key Information section.

## Sending the usage report through proxy

If the computer is connected to the Internet through proxy, it cannot send the usage report to the AppWorks Platform License Service. You can modify the AppWorks Platform properties file to allow the usage report to be sent through the proxy.

**To send the usage report when the computer is connected to Internet through proxy:**

1. Click **Start > Programs > AppWorks Platform <version> > <Instance Name> > Tools > Management Console**.  
The Management Console opens.
2. Click **Platform Properties**.
3. Set the following properties in the AppWorks Platform properties file:

```
bus.http.ProxySet = true
bus.http.proxyHost = <proxyhostname> (here proxyhostname must be an IP address)
bus.http.proxyPort = <proxyhostport>
bus.http.proxyUser = <proxyusername>
bus.http.proxyPassword = <proxyPassword> (in plain text)
bus.http.proxyAuthorization = <authorizationType>
```

For example, the following values can be set:

```
bus.http.ProxySet = true
bus.http.proxyHost = 10.192.60.47
bus.http.proxyPort = 8080
bus.http.proxyUser = proxyAdmin
bus.http.proxyPassword = adminPassword
bus.http.proxyAuthorization = Basic
```

**Note:** If HTTPS must be enabled, set the following property in the AppWorks Platform properties file:

```
bus.gateway.protocol = https
```

4. Restart the OpenText AppWorks Platform (<instance name>) and the Web server for the changes to take effect.

# Chapter 9

# Using composite application logging

AppWorks Platform allows you to execute, monitor, change, and continuously optimize the processes and operations that are critical for your business continuity. During this process, an administrator must constantly monitor the health of the processes and debug the issues that hinder the process flow, which is achieved through logging.

A composite application might span multiple applications. Logging enables you to track the process flow across these applications and becomes essential for the feedback mechanism. Other than tracking the log messages generated through logging, it is important to correlate the context of these messages to debug an issue.

Composite Application Logging (CAL) is a comprehensive logging framework that enables automatic insertion and propagation of contexts across the application processing lifecycle. The context can be related to user, application, environment, and call stack. Few sample contexts are user ID, URL of the application, Web service name, and computer name. This context helps to build the causality relation and enables the drill-down analysis of log messages.

CAL enables a logging service at the inception, processing, completion or at any stage of a business process lifecycle, and traces the end-to-end flow of the request.

Most of the AppWorks Platform-based applications use multiple service containers to operate and generate component-specific logs. These logs are stored at different places, which make it difficult to debug an issue. Additionally, a composite Web service cannot trace the error or a SOAP fault using a specific log entry in a file.

In the case of composite applications, there is a need to publish, collect, store and view log messages pertaining to multiple applications. CAL addresses this by providing a centralized logging mechanism. All log messages can be pushed to a database. In addition, CAL offers the following features:

- Single configuration point for the entire AppWorks Platform system. The configuration is dynamic and the service containers need not be restarted.
- Single view of log messages generated across multiple services used by the composite application.

CAL is based on logger context that serves as a single container, where in each application adds its own component-specific information to this container as the control passes from one component to another. It further allows you to configure and induce application-specific information and messages that are bound to a single ID. CAL assigns a unique ID to every

instance of execution in the request flow. Thereby, it offers a way to view the log messages that are categorized based on the instance of execution. This unique identification helps you identify the application associated with each ID and easily detect and isolate issues at the platform or application level.

The contexts that are propagated from one component to another are used to search and debug logs generated by all the services from a single user interface. Using this interface, you can query and filter log messages. This helps you view and extract the log messages based on search criteria.

CAL facilitates end-to-end message logging and message tracking for composite applications. It supports logging for most commonly used databases such as Oracle, MS SQLServer, and PostgreSQL.

## Viewing log messages using the Log Viewer

The Composite Application Logging (CAL) implementation facilitates the comprehensive logging mechanism where multiple applications can log their details at a single place.

The Logging service group is necessary for database logging and is enabled by default for all the service containers. You must configure a database that serves as the single location to store log messages generated during the execution of a composite application. You can specify this location during the AppWorks Platform installation in the Database Information page of the wizard.

The AppWorks Platform Log Viewer helps you find log messages through search criteria.

### Before you begin:

- You must have the Log Viewer Admin role.

### To view the log messages:

1. On the Welcome page, click  (Log Viewer).  
The Log Viewer window opens and displays the log messages that are generated in the last 24 hours that are of Error and Fatal severity types.
2. Expand the Search Filters group box and do the following:
  - a. In Diagnostic Context, enter the combination of a name-value pair to match the log content. Example: `host=hostname&processid=1234`
  - b. To view the logs generated in a specific period, set the dates in From and To:
    - Setting only the From date displays the logs generated from that date to the latest date.
    - Setting only the To date displays the logs generated in the last 24 hours.
    - Not setting a period displays all the available logs.
  - c. In the Severity list, select a severity type. See [Categories and severity levels](#). Selecting a severity level displays all its messages and the messages of the levels under it. For example, setting INFO as the severity level displays the messages

belonging to the INFO, WARN, ERROR, and FATAL levels.

d. Click **Search**.

**Note:** To refine your search criteria, you can select Advanced Search. See [Advanced search options in Log Viewer](#). Log messages that match the search criteria display in a table. By default, the table displays only Severity, Timestamp, Message, and Category values of the log messages.

**Note:** To view details of more options, right-click the table header and select Column Chooser. This provides the list of options from which you can select the required parameters.

3. Select a log message from the search results.

Related details display in the Details group box. See [Log message search parameters](#).

**Note:** Context-specific information is entered in Diagnostic Context by the respective component. You can use these details as name value pairs for further grouping of messages by applying the search criteria. Ensure to delimit your values by '&'. For example, `host=CIN0321L&processid=5948&messageid={BEABD11B-BB14}` is an MDC value.

Log messages and the corresponding details are displayed for viewing.

The Log Viewer application fails to work when you change the location of a log file. However, for standalone service containers (with a custom log policy), you can change the file name. If there is a genuine need for creating logs in a separate location other than the `<AppWorks Platform_installdir>/Logs` folder, you can use the symbolic link of the operating system. For information about creating symbolic links in Windows, see <http://support.microsoft.com/kb/205524>. In Linux, you can use the `'ln'` command for symbolic links.

Currently, there are no archiving or deleting features for the AppWorks Platform log table. However, a table by the name `CORDYS_LOG` is created in the database that is specified during installation. You can delete the entries from the table using external SQL editors. See the database scripts to create the AppWorks Platform log table at:

```
<AppWorks Platform_
installdir>/components/managementlib/dbscripts/createscripts/CAL_<Database
type>.sql.
```

#### To disable logging to the database:

1. Go to **System Resource Manager > Service Groups App Palette > Logging > Service Group Properties - Logging**.
2. Clear the **Enable Composite Application Logging** check box.

## Log message

This Web service operation accepts the log request from the presentation layer. It does not return any response.

### SOAP request

```
<SOAP:Envelope xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP:Body>
    <LogMessage xmlns="http://schemas.cordys.com/log/1.0/">
      <DC name="user">The user DN</DC>
      <DC name="url">The URL of the application</DC>
      <DC name="applicationID">The application ID used by BDI</DC>
      <DC name="fileName">The htm or js file name where the log message is
      originating</DC>
      <DC name="BPM Name">MyProcess</DC>
      <Level>ERROR</Level>
      <LocalizableMessage>
        <MessageCode>Fully Qualified Message ID</MessageCode>
        <Insertion>insertions to the message if required</Insertion>
        <Insertion>insertions to the message if required</Insertion>
      </LocalizableMessage>
      <Message>Custom messsage</Message>
    </LogMessage>
  </SOAP:Body>
</SOAP:Envelope>
```

### Request parameters

Following are the request parameters:

Parameter	Description	Data type
DC	Diagnostic context.	String
Insertion	Simple string or a localizable message.	String
Level	Logger level that the receiving SOAP service or web application must use. The value of this field overrides values set earlier. Valid input values are DEBUG, INFO, WARN, ERROR, and FATAL.	String
LocalizableMessage	Standard messages from the message bundle, which can take one or more insertions.	Complex

Parameter	Description	Data type
Message	<p>Optional custom message. Use only if the localizable message is unavailable.</p> <p><b>Note:</b> OpenText recommends that you do not use this parameter.</p>	String
MessageCode	<p>Fully qualified message ID from the message bundle. Valid input values are strings in &lt;message bundle name&gt;. &lt;message-id in the bundle&gt; form.</p>	String

# Chapter 10

## Auditing

Auditing is a process of tracking changes that occur in a specific period. These changes can be made by different users such as a user updating data through an application or an administrator deploying a package. In some scenarios, auditing is required to comply with government regulations. It involves maintaining a record of what modifications were made, who made them, when they were made, and on what data they were made. This is known as audit information.

Audit information enables you to track changes made to an artifact and rectify defects, if any, found in one of these changes. AppWorks Platform provides application auditing features for administrators on different artifacts. Auditing can keep track of changes made to an AppWorks Platform computer, which includes the following administrator actions:

- Deploying a package
- Maintaining users
- Maintaining, starting, and stopping a service container
- Updating XML Store and LDAP
- Changing security settings

You can also audit Web service requests on both the following levels:

- Web Gateway
- Service container

You can audit authentication on a user level. Tracking of the following events is possible:

- User sign-in
- User sign-off
- Sign-in failure

The administrator can configure auditing to audit all the actions. Applying audit filters can limit the audit data generation because auditing can result in a large amount of data. The administrator can view and search for audit data. This helps to find changes made by a specific user or changes made in a certain period on a specific artifact, for example, changing a specific property on the LDAP service container in the System organization last week. The administrator can also search for all user sign-ins that failed on a specific date.

**Note:** Maintaining audit records has a performance impact on the AppWorks Platform runtime. Enabling auditing might also result in a large amount of audit data, especially for Web gateway and Web service artifacts. To prevent this, use filtering and archive the audit data on a regular basis.

## Database auditing

You must track changes made to the database for future reference. Database auditing helps store changes made to the database in the form of an audit table by generating methods on the table as defined by the administrator. You can generate specific default methods for the audit table using the Method Set Generator. These include methods that indicate the insertions, deletions, and updates performed on the database. If you require to run specific queries on the audit table, you can build and use custom methods to view the database changes.

The WS-AppServer connector is equipped with a facility to audit databases. The administrator can define the audit configuration details, which are stored in LDAP with the service details. After defining the audit configuration, the administrator can either enable or disable the auditing feature.

Database auditing provides a log of all changes made to the database over a period. Since the auditing feature is enabled by the WS-AppServer connector, all requests being processed by the service are audited. The name of the AppWorks Platform user who made the changes is available as part of the auditing information.

You can only audit changes made through the service (through the AppWorks Platform framework). Any changes made directly to the database tables do not reflect.

## Configuring audit settings for a database

Audit settings help in recording all the database-related activities. You can identify specific tables in the database for which auditing is required. This feature enables you to monitor specific tables and not the entire database.

### Before you begin:

- You must have the WS-AppServer Administrator role.
- You must attach the method set Audit 1.0 Web service interface to the WS-AppServer service.
- You must select the Enable Audit check box for the WS-AppServer connector parameters.

The audit settings feature creates one audit table for each selected table. Apart from all the columns in the base table, the audit table contains the following four additional columns:

Column name	Description
CordysUser	Credentials of the user who modified the

	<p>data.</p> <p><b>Note:</b> The length of the CordysUser field is restricted to 20 characters. If the length of the user name is greater than 20 characters, it is truncated to 20 characters while updating the audit table.</p>
CordysOperationType	<p>Type of data operation:</p> <ul style="list-style-type: none"> <li>■ I: Insert</li> <li>■ U: Update</li> <li>■ D: Delete</li> </ul>
CordysTimestamp	Date and time when the data was stored in the database. This is the time stamp of the system where the Audit service is active.
CordysGuid	Unique GUID (identifier) for each insert, update, and delete request.

**Note:** For an update, only the new record is stored in the audit table. The fields of the table that are not provided in the insert, update, or delete request are set to null in the audit table.

#### To configure the audit settings for a database:

1. On the Welcome page, click  (Database Metadata Manager).  
The Database Metadata Manager window opens. The Metadata explorer and the contents of the configured WS-AppServer service are displayed.
2. Expand the contents of the WS-AppServer service until you view  (Tables) and its contents.  
All the tables in the database are listed.
3. Right-click a table and select **Audit Configuration**.  
The Audit Configuration application palette appears to the right of the Metadata explorer with the Table Name. A new field, Audit Table Name is created and displayed on the palette window containing the default name <original tablename>\_Audit. The columns of the database table are presented in a tabular format with a check box against each column.

**Note:** Uninstalling AppWorks Platform does not delete the audit fields. You must manually remove them from the database.

4. In the Audit column of the table, under Fields, select the check box against the column to audit and store in the newly created audit table.

**Note:** The column that is a primary key for the table is selected by default and cannot be cleared. It indicates that such a column is audited by default.

5. Click **Save**.  
A confirmation message is displayed stating that the configuration is saved.

The audit settings are configured for the database table. The audit table <original tablename>\_Audit is displayed with the  icon and added to the list of tables. Click the audit table to view its details on the Audit Configuration tab.

#### **After you complete:**

- Restart the related service for the audit settings to be effective.
- To delete the audit details on the Audit Configuration tab, click **Purge Audit Table**. The audit details are removed from the audit table.

## **Viewing audit data for artifacts**

An artifact is an object or a system resource used during software development and maintenance.

Auditing involves maintaining records of administrative actions, such as the type of action performed, the artifact on which the action was performed, who performed the action, and when the action was performed. Audit information enables tracking of the actions performed on the artifact and rectifying the defects caused by these actions. An audit action, for example, can be restarting a service container, installing an application package, publishing a business process, invoking a Web service operation, adding a user in LDAP, and changing an XML document in the XML Store.

#### **Before you begin:**

- You must have the Audit Viewer role, which is an internal role.

#### **To view the audit data:**

1. On the Welcome page, click  (Audit Viewer).  
The Audit Viewer window displays the artifacts with the audit data in a tabular format.
2. In the Search group box, specify the criteria for the artifact search using the following options:

<b>Option</b>	<b>Description</b>
Selecting an artifact type	<p>In the Artifact Type list, select an artifact type. Only those artifact types that can be audited are displayed in the list.</p>
Performing a wildcard search	<p>In Artifact, enter text for a wildcard search. A wildcard character represents one or more characters. Use the following two wildcard characters in your search term:</p>

<b>Option</b>	<b>Description</b>
	<ul style="list-style-type: none"> <li>■ % (percentage): Matches any number of characters, even zero characters.</li> <li>■ _ (underscore): Matches exactly one character.</li> </ul> <p><b>For example:</b></p> <ul style="list-style-type: none"> <li>■ gar% matches 'garry', 'garrett', and 'gargoyle' or anything that begins with the characters 'gar'.</li> <li>■ %te% matches 'textual', 'context', and 'text123'.</li> <li>■ tex% matches 'text' but not 'monotext'.</li> <li>■ %tem finds 'system' but not 'temporary'.</li> <li>■ _ean matches all the four letter words ending in 'ean', that is, Dean, Sean, and Mean.</li> <li>■ to_ matches 'top' and 'tow' but not 'today'.</li> </ul> <p><b>Note:</b> Using wildcard characters might have a negative impact on the search performance.</p>
Specifying an organizational user	<p>Perform the following steps:</p> <ol style="list-style-type: none"> <li>a. In Performed by, enter a user name or click  (Browse) to select an organizational user. The Organizational Users dialog box displays the organizational users.</li> <li>b. Select the user who executed the action on the artifact. Alternatively, you can click  (Find) to search for and select the user. The name of the user displays in Performed by.</li> </ol>
Specifying the start and end dates	<p>Perform the following steps:</p> <ol style="list-style-type: none"> <li>a. Click  (Date) for From Date and To Date respectively to select the start and end dates during which the action on the artifact was performed.</li> </ol>

<b>Option</b>	<b>Description</b>
	<p>Calendars appear for the From Date and To Date respectively.</p> <p><b>Note:</b> From Date is filled by default with the date from one week ago at midnight.</p> <p>b. Select the start and end dates. The dates display in From Date and To Date respectively.</p>

3. In the Space group box, you can specify the criteria for the search using the following options:
  - To view audit data logged at the shared level, select **Shared**.
  - To view audit data logged at the organization level, select **Organization**.
    - To view audit data from all the organizations, select **All**.
    - To view audit data from a specific organization, select **Specific**. You can browse and select the organization name or enter it in the box.

**Note:** The Shared and Organization options are only visible in the System organization.

4. In the Actions group box, select the actions for the search.
- Note:** The actions are specific to the artifact type. In the Search group box, if you do not select an artifact type from the Artifact Type list, you cannot filter the actions. When you select an artifact type, the corresponding actions display in the Actions group box.
5. In the Status group box, select the status for the search using one of the following options:
    - Incomplete: The action started but did not complete, for example, there was a system failure.
    - Complete: The action was complete.
    - Failed: The action failed.
    - Not Authorized: The action did not complete because authorization was inadequate.
  6. Set the maximum number of artifacts and actions to be read.  
When the Max Number of Artifacts and Max Number of Actions are set to 100 respectively, a maximum of 100 artifacts and 100 actions by artifact are read and displayed in the Audit Viewer window. The lower these numbers, the better the performance of reading and displaying the audit data.
  7. Specify the way the audit data is presented. To group the actions by the artifacts, select **Group Actions by Artifacts**. See [Audit Viewer interface](#). For example, you can view all the service containers on which actions such as start, stop, and reset have been

performed. You can click each service container to view who performed the action, which action was performed, and when it was performed.

**Note:** If the Group Actions by Artifacts check box is not selected, only the table actions display sorted in the descending order on date and time. You can view who performed the action, which action was performed, and when it was performed, sorted on date and time, with or without a filter.

8. Click **Search**. For information about the artifacts on which matching audit actions have been found, see [Audit Viewer interface](#).
9. In the Artifacts Audit Information section, select an artifact row in the Artifacts table. This is required only if you selected Group Actions by Artifacts. If not, the Actions table is displayed directly.  
All the actions based on the search criteria are displayed in the Actions table.
10. To view the input data for an audit action, select the audit action row and click  (Show Action Input).
11. To view the artifact in a pictorial format, click  (Show Artifact). This option is available only if you selected the BPM or Decision Table artifact type from the Artifact Type list in the Search group box.

**Note:** The graphical view of the published artifacts is displayed only if the Debug Process Models option is enabled. See Enabling Debugging on previous Business Process Models in the *AppWorks Platform Advanced Development Guide*.

**Note:** To view the graphical view of artifacts (BPM and Decision Table) of non-system organizations from the System organization, the System Administrator must also have the Administrator role in those organizations.

The audit data for the AppWorks Platform Artifacts is viewed.

**Note:** If Web gateway auditing is enabled, the Web service calls that fill the rows are audited too and are in the incomplete state. Therefore, the Web gateway artifact is displayed when you search for incomplete records. After the Web service calls are complete, no records are displayed.

## Configuring audit for artifact types

You can enable or disable auditing per artifact type.

### Before you begin:

- You must have the Audit Administrator role, which is an internal role.
- You must start the Audit service container for a hot change of the configuration. For a fresh install, you must restart AppWorks Platform and the Web server.

### To enable or disable the auditing per artifact type:

1. On the Welcome page, click  (Audit Configuration).

The Audit Configuration window opens. The current audit configuration sorted per artifact type is displayed.

2. To configure audit for the artifact type, select **Enable**.

**Note:** The audit configuration provided for an artifact type in the Audit Configuration window reflects in all the AppWorks Platform organizations.

3. Click .

The audit is enabled for the selected artifact types.

**Note:** Auditing of the Web service and Web gateway artifacts might result in large amounts of data. To prevent this, use a filter or archive the audit data on a regular basis.

## Audit artifacts and actions

You can audit several artifacts. Each artifact has a set of actions. The following audit artifacts are available.

Artifact type	Action types	Artifact key
Authentication	<ul style="list-style-type: none"> <li>■ Logon</li> <li>■ Logoff</li> <li>■ Validation</li> </ul>	User DN
BPM	<ul style="list-style-type: none"> <li>■ Add</li> <li>■ Update</li> <li>■ Delete</li> </ul>	BPM Qualified Name Example: test\calculateBPM
Business Calendar	<ul style="list-style-type: none"> <li>■ Add</li> <li>■ Update</li> <li>■ Delete</li> </ul>	Business Calendar Qualified Name Example: test\bcl
CAP (enabled by default)	<ul style="list-style-type: none"> <li>■ Upload</li> <li>■ Deploy</li> <li>■ Undeploy</li> <li>■ Upgrade</li> <li>■ Rollback</li> <li>■ Revert</li> </ul>	CAP Name Example: Cordys BAM Modeler
CAP Security Settings (enabled by default)	<ul style="list-style-type: none"> <li>■ Update</li> </ul>	
Certificate Store	<ul style="list-style-type: none"> <li>■ Add</li> <li>■ Update</li> </ul>	Subject DN

<b>Artifact type</b>	<b>Action types</b>	<b>Artifact key</b>
	<ul style="list-style-type: none"> <li>■ Delete</li> </ul>	
ISVP (enabled by default)	<ul style="list-style-type: none"> <li>■ Load</li> <li>■ Unload</li> <li>■ Rollback</li> <li>■ Upgrade</li> </ul>	<p>ISVP Name Example: Cordys BAM Modeler</p>
LDAP	<ul style="list-style-type: none"> <li>■ Add</li> <li>■ Update</li> <li>■ Delete</li> </ul>	Full DN
Organizational Unit	<ul style="list-style-type: none"> <li>■ Add</li> <li>■ Update</li> <li>■ Delete</li> </ul>	<p>Organizational Unit Qualified Name Example: test\ou1</p>
Rule	<ul style="list-style-type: none"> <li>■ Add</li> <li>■ Update</li> <li>■ Delete</li> </ul>	<p>Rule Qualified Name Example: test\rule1</p>
Service Container	<ul style="list-style-type: none"> <li>■ Start</li> <li>■ Restart</li> <li>■ Reset</li> <li>■ Stop</li> </ul>	<p>Service Container DN Example: cn=Entity Server,cn=Entity Server, cn=soap nodes,o=system,cn=cordys, cn=CAbuild,o=vanenburg.com</p>
Trusted Application Publishers	<ul style="list-style-type: none"> <li>■ Add</li> <li>■ Update</li> <li>■ Delete</li> </ul>	Subject DN
Web Gateway	<ul style="list-style-type: none"> <li>■ Invoke</li> </ul>	<p>Method Set and Method Name DN Example: cn=UpdateXMLObject,cn=Method Set XMLStore,cn=Cordys XMLStore,cn=cordys, cn=defaultInst,o=vanenburg.com</p>
Web Service	<ul style="list-style-type: none"> <li>■ Invoke</li> </ul>	<p>Method Set and Method Name DN Example: cn=UpdateXMLObject,cn=Method Set XMLStore,cn=Cordys XMLStore,cn=cordys, cn=defaultInst,o=vanenburg.com</p>
XML Store	<ul style="list-style-type: none"> <li>■ Add</li> </ul>	<p>Full Path Example:</p>

<b>Artifact type</b>	<b>Action types</b>	<b>Artifact key</b>
	<ul style="list-style-type: none"> <li>■ Update</li> <li>■ Delete</li> </ul>	/Cordys/notification/inboxconfig

Audit artifacts are useful when creating audit filters. See [Configuring audit filters for artifact types](#).

## Archiving audit data

The auditing component writes information to tables in the database specified on the Audit service container. The auditing configuration determines the amount of information stored in the audit tables. Database scripts enable audit data archival. This topic describes how to install and use the database scripts.

Database administrators archive audit data. The reader must have knowledge of basic database administration tasks such as creating tables and stored procedures and executing stored procedures.

You can use archiving scripts to archive the data in the following tables:

- CORDYS\_ARTIFACTS\_AUDIT: Contains the identifying information of an artifact for which an audit trail is stored
- CORDYS\_ARTIFACTS\_REVISIONS: Contains the audit trail itself

Archiving starts from the CORDYS\_ARTIFACTS\_REVISIONS table. All the records that match the archiving criteria are moved from the source table to the archive table ARCHIVE\_ARTIFACTS\_REVISIONS. As part of this, the corresponding parent records from CORDYS\_ARTIFACTS\_AUDIT are moved to the archive table ARCHIVE\_ARTIFACTS\_AUDIT only if the same information does not exist in the archive table. See [Archiving criteria](#).

## Installing the archiving scripts

The archiving scripts are not automatically installed when installing AppWorks Platform. The scripts are located in the directory `<AppWorks Platform_installdir>/components/audit/dbschema/archivescripts`. There are three scripts in this directory, one for each supported database type:

- `audit_archive_mssql.sql` for Microsoft SQL Server databases
- `audit_archive_oracle.sql` for Oracle databases

You must run the scripts in the context of the database (schema) that contains the audit tables CORDYS\_ARTIFACTS\_AUDIT and CORDYS\_ARTIFACTS\_REVISIONS.

Running the scripts creates a number of stored procedures that together provide the archiving functionality.

## Creating archive tables

When archiving scripts are run for the first time, the target tables for the archiving are automatically created in the same schema where the audit tables are located. It is possible to create the tables manually, for example, if specific storage options are required.

The structure of the target tables are copies of the source tables with additional columns:

- CORDYS\_ARTIFACTS\_AUDIT is archived to ARCHIVE\_ARTIFACTS\_AUDIT. In the target table, one additional column ARCHIVE\_DATE is added. This column contains the date (without time) on which the record was created or archived.
- CORDYS\_ARTIFACTS\_REVISIONS is archived to ARCHIVE\_ARTIFACTS\_REVISIONS. In the target table, two additional columns are added:
  - ARCHIVE\_DATE contains the date (without time) on which the record was archived.
  - AUDIT\_DATE contains the date (with time), version of the date-time the original record was created, calculated from the value of the PERFORMED\_AT column.

The scripts for archive tables differ based on the database type.

### Microsoft SQL Server

ARCHIVE\_ARTIFACTS\_AUDIT

```
SELECT TOP 0 * INTO [ARCHIVE_ARTIFACTS_AUDIT] FROM [CORDYS_ARTIFACTS_AUDIT];
ALTER TABLE [ARCHIVE_ARTIFACTS_AUDIT] ADD [ARCHIVE_DATE] DATETIME;
CREATE UNIQUE INDEX [UX_AUDIT_ID] ON [ARCHIVE_ARTIFACTS_AUDIT] ([AUDIT_ID]);
```

ARCHIVE\_ARTIFACTS\_REVISIONS

```
SELECT TOP 0 * INTO [ARCHIVE_ARTIFACTS_REVISIONS] FROM [CORDYS_ARTIFACTS_REVISIONS];
ALTER TABLE [ARCHIVE_ARTIFACTS_REVISIONS] ADD [ARCHIVE_DATE] DATETIME,
[AUDIT_DATE] DATETIME;
```

### Oracle

ARCHIVE\_ARTIFACTS\_AUDIT

```
CREATE TABLE ARCHIVE_ARTIFACTS_AUDIT AS SELECT * FROM CORDYS_ARTIFACTS_AUDIT WHERE ROWNUM < 1;
ALTER TABLE ARCHIVE_ARTIFACTS_AUDIT ADD ARCHIVE_DATE DATE;
CREATE UNIQUE INDEX UX_AUDIT_ID ON ARCHIVE_ARTIFACTS_AUDIT (AUDIT_ID);
```

## ARCHIVE\_ARTIFACTS\_REVISIONS

```
CREATE TABLE ARCHIVE_ARTIFACTS_REVISIONS AS SELECT * FROM CORDYS_
ARTIFACTS_REVISIONS WHERE ROWNUM < 1;
ALTER TABLE ARCHIVE_ARTIFACTS_REVISIONS ADD ARCHIVE_DATE DATE;
ALTER TABLE ARCHIVE_ARTIFACTS_REVISIONS ADD RECORD_DATE DATE;
```

**PostgreSQL**

## ARCHIVE\_ARTIFACTS\_AUDIT

```
CREATE TABLE ARCHIVE_ARTIFACTS_AUDIT (LIKE CORDYS_ARTIFACTS_AUDIT);
ALTER TABLE ARCHIVE_ARTIFACTS_AUDIT ADD COLUMN ARCHIVE_DATE DATE;
CREATE UNIQUE INDEX UX_AUDIT_ID ON ARCHIVE_ARTIFACTS_AUDIT (AUDIT_ID);
```

## ARCHIVE\_ARTIFACTS\_REVISIONS

```
CREATE TABLE ARCHIVE_ARTIFACTS_REVISIONS (LIKE CORDYS_ARTIFACTS_
REVISIONS);
ALTER TABLE ARCHIVE_ARTIFACTS_REVISIONS ADD COLUMN ARCHIVE_DATE DATE, ADD
COLUMN RECORD_DATE DATE;
```

**Using the archive tables**

After the archiving scripts are installed, a stored procedure is created to archive the auditing data. This stored procedure accepts four parameters:

- Archive date: Mandatory
- Batch size: Mandatory
- Organization: Optional, default is NULL
- Artifact type: Optional, default is NULL

**Note:** For Oracle and Microsoft SQL Server, the stored procedures support the actual optional parameters. You need not specify the parameter in the call to the stored procedure. It uses the default value.

The Batch size parameter specifies the number of records from CORDYS\_ARTIFACTS\_REVISIONS that are archived in a single database transaction. A typical value for this parameter is between 10,000 and 100,000. Choosing too low a value impacts the performance of the archiving significantly while choosing too high a value might cause issues with transactions running too long and running out of space in the transaction logs depending on the database type.

## Archiving criteria

The archiving scripts always require an archival date as the input. All audit records older than this date are archived. You can also use two optional criteria:

- **Organization:** Defines the organization for which the audit records are archived. The parameter value must be the full LDAP DN of the organization. If this parameter is not specified, all the audit records that match the other criteria are archived. If the string `SHARED` is provided, all the audit records for the shared level artifacts are archived.
- **Artifact type:** Defines the artifact type for which the audit records are archived. If this artifact type is not specified, all the audit records that match the other criteria are archived.

Examples of artifact types include:

- **LDAP:** Any changes made to the LDAP content, such as adding a role to a user, changing user information, or deleting a user.
- **Authentication:** Authentication related events, such as sign-ins when authentication is handled by AppWorks Platform and not by an external Identity Provider.

The archiving criteria differ based on the database type.

### **Microsoft SQL Server**

To archive all the data created before January 15 with a batch size of 10,000 records:

```
exec sp_archive_audit @in_archive_date = '2013-01-15', @in_batch_size = 10000;
```

To archive all the data for a specific organization that was created before January 15 with a batch size of 10,000 records:

```
exec sp_archive_audit @in_archive_date = '2013-01-15', @in_batch_size = 10000, @in_organization = '<LDAP DN of the organization>';
```

To archive all the data for the Authentication artifact type that was created before January 15 with a batch size of 10,000 records:

```
exec sp_archive_audit @in_archive_date = '2013-01-15', @in_batch_size = 10000, @in_artifact_type = 'Authentication';
```

To archive all the data for the Authentication artifact type and a specific organization that was created before January 15 with a batch size of 10,000 records:

```
exec sp_archive_audit @in_archive_date = '2013-01-15', @in_batch_size =
10000, @in_organization = '<LDAP DN of the organization>', @in_artifact_
type = 'Authentication';
```

To archive all the data older than seven days with a batch size of 10,000 records:

```
DECLARE @archive_date DATETIME
SET @archive_date = DATEADD(dd, -7, GETDATE());
exec sp_archive_audit @in_archive_date = @archive_date, @in_batch_size =
10000;
```

## Oracle

**Note:** The Oracle version of the archiving functionality is bundled in to a single PL/SQL package called ARCHIVE\_AUDIT.

To archive all the data created before January 15 with a batch size of 10,000 records:

```
exec archive_audit.sp_archive_audit(TO_DATE('20130115', 'YYYYMMDD'),
10000);
```

To archive all the data for a specific organization that was created before January 15 with a batch size of 10,000 records:

```
exec archive_audit.sp_archive_audit(TO_DATE('20130115', 'YYYYMMDD'),
10000, '<LDAP DN of the organization>');
```

To archive all the data for the Authentication artifact type that was created before January 15 with a batch size of 10,000 records:

```
exec archive_audit.sp_archive_audit(TO_DATE('20130115', 'YYYYMMDD'),
10000, NULL, 'Authentication');
```

To archive all the data for the Authentication artifact type and a specific organization that was created before January 15 with a batch size of 10,000 records:

```
exec archive_audit.sp_archive_audit(TO_DATE('20130115', 'YYYYMMDD'),
10000, '<LDAP DN of the organization>', 'Authentication');
```

To archive all the data older than 7 days with a batch size of 10,000 records:

```
exec archive_audit.sp_archive_audit(SYSDATE-7, 10000);
```

## PostgreSQL

To archive all the data created before February 15 with a batch size of 10,000 records:

```
SELECT * FROM sp_archive_audit('2016-02-15', 10000, NULL, NULL);
```

To archive all the data for a specific organization that was created before February 15 with a batch size of 10,000 records:

```
SELECT * FROM sp_archive_audit('2016-02-15', 10000, '<LDAP DN of the organization>', NULL);
```

To archive all the data for the Authentication artifact type that was created before February 15 with a batch size of 10,000 records:

```
SELECT * FROM sp_archive_audit('2016-02-15', 10000, NULL, 'Authentication');
```

To archive all the data for the Authentication artifact type and a specific organization that was created before February 15 with a batch size of 10,000 records:

```
SELECT * FROM sp_archive_audit('2016-02-15', 10000, '<LDAP DN of the organization>', 'Authentication');
```

To archive all the data older than 7 days with a batch size of 10,000 records

```
SELECT * FROM sp_archive_audit(current_date - interval '7' day, 10000);
```

## Known limitations

The following are the known limitations.

### Possible conflict between archiving and auditing

The archiving scripts remove records from the CORDYS\_ARTIFACTS\_REVISIONS table. Therefore, it is possible to audit a transaction in AppWorks Platform for the same artifact instance that has just been deleted by the archiving script. In this case, the transaction that is audited can fail. It is advisable to run the archiving script only for a date, which is a few days prior to the current date and either during a maintenance window or at a time when the number of transactions is the lowest.

## **More records processed than the specified batch size**

The actual number of records that are processed in a single transaction are usually higher than the specified batch size. This occurs because the batch size determines the number of records from the `CORDYS_ARTIFACTS_REVISIONS` table that are processed. As part of the archiving, related records from the `CORDYS_ARTIFACTS_AUDIT` table are also copied or moved. These do not count against the batch size.

## **Configuring audit filters for artifact types**

Audit filters audit only specific actions or artifacts. They define audit actions on an audit artifact type and use regular expressions to filter specific audit artifact keys.

Audit filters are configured in the System organization and stored in the XML Store. Any audit filter at the organization or shared space level is automatically applied when the Audit service container starts or resets.

An audit filter file can specify one or more filters for only one artifact type. You can have multiple filter files in the XML Store for the same artifact type. If any filter matches, an audit record is written. When auditing of an artifact type is enabled and there is no filter, all the actions on all the artifacts of the artifact type are audited.

### **Before you begin:**

- You must have the Audit Administrator role.
- You must be logged in to the System organization.
- You must configure audit for the artifact types. See [Configuring audit for artifact types](#).

### **To create or update a filter at the System organization level:**

1. On the Welcome page, click  (XMLStore Explorer).
2. Navigate to the folder `Collection/Cordys/audit/filters/`. If the folder does not exist, you must create it.
3. To add filters, add items to the XML store in the System organization at the organization level. For the filter name, OpenText recommends that you enter a name with the artifact type. See [Adding Items to XMLStore](#).
4. After adding filters or changing the configurations in the XML Store, reset the Audit service container to activate the new settings using the System Resource Manager. Verify that the filters have the following structure.

```
<ArtifactType name="LDAP" xmlns="http://schemas.cordys.com/1.0/auditing">
  <Filters>
    <Filter>
      <ActionTypes>
        <ActionType>Add</ActionType>
        <ActionType>Update</ActionType>
```

```

<ActionType>Delete</ActionType>
</ActionTypes>
<Expression>
    <! [CDATA[ ^cn=Single Sign-On, cn=Single Sign-On, cn=soap nodes, .* ]]>
</Expression>
</Filter>
</Filters>
</ArtifactType>

```

where:

ArtifactType name	Name of the artifact type to which the filters belong. It must be the same as the name in the audit configuration. If there is no configuration to match any filters, an error is displayed on starting or resetting the Audit service container.
Filter	Content that must be audited for an artifact type. You can define multiple filter elements in one file.
ActionTypes	List of action type names to audit. If this element is not specified, filtering is not done based on the action type.
Expression	<p>Regular expression filtering on the artifact key. Only the actions of artifacts with a key that matches the expression are audited. Content must be placed within <code>CDATA</code> because it contains certain characters that must be escaped in XML. If this element is not specified, filtering is not done based on the key. The regular expression must be a normal Java regular expression.</p> <p>Another example for the regular expression to audit the process of creating organizational users is:</p> <pre>&lt;! [CDATA[ ^cn=.* ,cn=organizational users,o=.* ,cn=cordys,cn=defaultInst,o=cordyslab.com] ]&gt;.</pre>

The filter definition specifies that any create, update, and delete action on the Single Sign-on service container configuration is audited.

For information about the available artifact types and actions, see [Audit artifacts and actions](#).

### Packaging audit filters

Audit filters are stored in the XML Store and can therefore be packaged through Collaborative Workspace (CWS) and deployed on other computers.

OpenText recommends creating a separate project for the audit configuration for ease of delivering the new audit configurations.

### To package audit filters:

1. Create a CWS project.
2. Create the folder structure: /auditconfig/Collection/Cordys/audit/filters/.
3. Add an XML object such as `LDAP Filter.xml` to the filters folder.

4. Add the audit filter content to the XML object created in Step 3.
5. Right-click the `auditconfig` folder and select **Start Point of Qualified Name**.
6. To publish the folders and content to the XML Store, add an XML Store Content Definition.
7. Package the project.

# Chapter 11

## Application server

AppWorks Platform leverages Apache TomEE, an application server that was introduced to host the Entity Runtime. It also hosts the Web Gateway component. From AppWorks Platform 16 onward, Apache TomEE is the only supported application server. Currently, AppWorks Platform supports only Apache TomEE as an application server. For information about the supported version, see *AppWorks Platform Supported Environments*.

### Managing Apache TomEE

It is mandatory to use Apache TomEE as an application server for AppWorks Platform. The Apache TomEE application server is an additional component that is installed, configured, and managed separately from AppWorks Platform.

### Installing Apache TomEE

You must install the Apache TomEE application server and AppWorks Platform on the same node. Some components of AppWorks Platform such as Entity Runtime are deployed on the application server. These components use the authentication configuration specified in the Security Administration task. For example, when you configure OTDS with AppWorks Platform, the AppWorks Platform-specific components that are deployed in Apache TomEE also use the same OTDS configuration for authentication. The user and role information is shared between these components.

The AppWorks Platform components that require the Apache TomEE application server are deployed through application packages. These contain the application server WAR files that are deployed in Apache TomEE when deploying the application packages. You can undeploy these WAR files by undeploying the corresponding application packages. Deploying the WAR files in Apache TomEE makes the SOAP-based Web services available. These web services can be accessed within AppWorks Platform through a Platform connector. A Web service interface that is bound to the Platform connector is used as a proxy to the hosted Web application. The Platform connector integrates Web services, which are deployed in Apache TomEE, into AppWorks Platform.

## Adding servlet filters to Apache TomEE

When adding servlet filters in TomEE, enable `async-supported` for the filter and add `REQUEST`, `FORWARD` and `ASYNC` dispatchers to filter-mapping. The configuration must be packaged in a JAR as `META-INF/web-fragment.xml` and placed in the `webroot/WEB-INF/lib` folder.

An example of a web-fragment configuration:

```
<?xml version="1.0" encoding="UTF-8"?>
<web-fragment version="3.0" xmlns="http://java.sun.com/xml/ns/javaee"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
    http://java.sun.com/xml/ns/javaee/web-app_3_0.xsd">
<name>myFragment</name>
<ordering>
    <before>
        <others/>
    </before>
</ordering>
<filter>
    <filter-name>MyFilter</filter-name>
    <filter-class>com.mycompany.MyFilter</filter-class>
    <async-supported>true</async-supported>
</filter>
<filter-mapping>
    <filter-name>MyFilter</filter-name>
    <url-pattern>/*</url-pattern>
    <dispatcher>REQUEST</dispatcher>
    <dispatcher>FORWARD</dispatcher>
    <dispatcher>ASYNC</dispatcher>
</filter-mapping>
</web-fragment>
```

## Configuring Apache TomEE

An administrator must configure Apache TomEE. See [Configuring Apache TomEE](#).

## Securing Apache TomEE

You must harden or secure the Apache TomEE application server before use in production. A default installation of Apache TomEE is configured to be reasonably secure for most use cases by default, but for production usage, it might require hardening. For information about hardening Apache TomEE, see [Securing Apache TomEE](#).

## Monitoring Apache TomEE

You can manage and monitor Apache TomEE in a standard manner with JMX. However, Apache TomEE also maintains its own log files. See [Managing Apache TomEE](#).

## Running service containers in Apache TomEE

From AppWorks Platform 16 onward, you can run service containers in Apache TomEE. Classic service containers run in a separate JVM, started by the Monitor service. You can also configure these to run in the application server. See [Configuring OS processes for a service container](#). These service containers run inside Apache TomEE and not as separate JVM processes. By default, the Business Process, Notification, Scheduling, Rule Management, Data Transformation, and Platform service containers are configured to run in Apache TomEE, but you can also run other service containers in Apache TomEE, except the LDAP, Logging, and Single Sign-on service containers. The advantage is reduced memory consumption and correct transactional semantics when triggering actions from Entity Runtime to Business Process, Case Management and Notification service containers.

Running a service container in the application server is impacted by the following:

### Routing

The routing logic is optimized to route requests with the application server. See [Routing Web service requests](#).

### JRE configuration

The manner in which JVM properties are handled is different when a service container runs in an application server. It only is possible to set the service container classpath. The JVM properties are used as configured on the application server. See the JRE Configuration Tab section in Service container properties interface in the *AppWorks Platform Advanced Development Guide*.

### JMX connection

The JMX connection string for service containers running in an application server is different. The Mbeans for these service containers are registered in the application server RMI registry. See [Connecting to JMX through JConsole](#).

### Logging

All log messages of the service containers that run in the same application server are logged to the log file that belongs to the application server. See the Log Settings Tab section in Service container configuration interface in the *AppWorks Platform Advanced Development Guide*.

# Configuring Apache TomEE

## Before you begin:

Ensure that the AppWorks Platform Web pages are accessible to TomEE and the Servlet filters are loaded.

## Running the configureapplicationserver script

Run the `configureapplicationserver` script to:

- Provide folder permissions for AppWorks Platform to deploy WAR files and allow TomEE to read and write into specific folders of the <AppWorks Platform\_installdir>.
- Add AppWorks Platform JARs to the classpath of TomEE.
- Add configuration to the TomEE service (Windows) or TomEE daemon (Linux).
- Add context files to expose web pages and services in TomEE.

Based on your operating system, the `configureapplicationserver` script is available at:

### To run the configureapplicationserver script automatically:

- When installing AppWorks Platform, select the **Automatic Deployment** option on the TomEE Details page.  
The installer runs the `configureapplicationserver` script.

### To run the configureapplicationserver script manually:

1. When installing AppWorks Platform, do not select the Automatic Deployment option on the TomEE Details page.
2. Review the changes made to the application server and then run the `configureapplicationserver` script manually:

Operating System	Action
Windows	<p>In a PowerShell prompt, run the following as an administrator:</p> <pre>cd &lt;&lt;AppWorks Platform_installdir&gt;&gt;\bin Set-ExecutionPolicy Bypass -Scope Process .\configureapplicationserver.ps1 install</pre>
Linux	<p>In the Linux terminal, run the following as root:</p> <pre>cd &lt;&lt;AppWorks Platform_installdir&gt;&gt;/bin ./configureapplicationserver.sh install</pre>

### To uninstall configureapplicationserver:

- Run the following scripts to uninstall configureapplicationserver:

Operating System	Action
Windows	<p>In a PowerShell prompt, run the following as an administrator:</p> <pre>cd &lt;&lt;AppWorks Platform_installdir&gt;&gt;\bin Set-ExecutionPolicy Bypass -Scope Process .\configureapplicationserver.ps1 uninstall</pre>
Linux	<p>In the Linux terminal, run the following as root:</p> <pre>cd &lt;&lt;AppWorks Platform_installdir&gt;&gt;/bin ./configureapplicationserver.sh uninstall</pre>

**Note:** When uninstalling AppWorks Platform, the actions that were performed by running the script are reverted and the deployed Web application Entity Runtime is removed.

## Content expiration

When installing AppWorks Platform, the Advanced Web server configuration panel enables configuring content expiration. However, these settings do not apply when using TomEE as a Web server. Therefore, to configure content expiration for TomEE, you must set it in one of the following ways:

- ISO 8601 format
- Integer value

### To configure content expiration for TomEE in ISO 8601 format:

- Open the `wcp.properties` file and set the values of the property `com.cordys.web.caching.maxage` in the [ISO 8601 duration format](#) that is also supported in `java.time.Duration`.  
For example:

Property	Description
<code>com.cordys.web.caching.maxage=PT10S</code>	10 seconds for content expiration
<code>com.cordys.web.caching.maxage=PT15M</code>	15 minutes
<code>com.cordys.web.caching.maxage=PT1H</code>	1 hour
<code>com.cordys.web.caching.maxage=P30D</code>	30 days
<code>com.cordys.web.caching.maxage=P30DT12H5S</code>	30 days, 12 hours and 5 seconds

2. Restart TomEE.
3. Restart AppWorks Platform.

**To configure content expiration for TomEE as an integer value:**

1. Open the `wcp.properties` file and set the values of the property `com.cordys.web.caching.maxage` as an integer value, followed by **s**, **m**, **h**, or **d**, indicating a value in seconds, minutes, hours, or days.

For example:

Property value	Caching header
10s	Cache-Control:max-age=10
15m	Cache-Control:max-age=900
1h	Cache-Control:max-age=3600
30d	Cache-Control:max-age=2592000
<property not set>	Cache-Control:no-cache

2. Restart **TomEE**.
3. Restart AppWorks Platform.

## Managing Apache TomEE

Many management tasks are common to TomEE and Apache Tomcat installations.

For documentation, see:

- TomEE: <http://tomee.apache.org/documentation.html>.
- Apache Tomcat : <http://tomcat.apache.org/tomcat-8.0-doc/index.html>.

## Log messages

AppWorks Platform issues log messages from the Web gateway that are logged in the file `<AppWorks_Platform_installdir>/Logs/Application_Server.xml`. TomEE also maintains log files. You can configure the location of these log files in the file `$CATALINA_BASE/conf/logging.properties`. The default location is `$CATALINA_BASE/logs`. See <http://tomcat.apache.org/tomcat-8.0-doc/logging.html>.

## JMX

AppWorks Platform and TomEE support Java Management Extensions (JMX). This technology provides tools to manage and monitor devices (printers), applications and other service driven networks. See [Managing operations through Java Management Extensions](#). A JMX-

capable management product can connect to TomEE to browse a tree of managed beans that TomEE and AppWorks Platform define.

As described in [Configuring Apache TomEE](#), TomEE must run as a Linux daemon, Windows service, or under a dedicated user account, which makes it impossible to connect to JMX locally. Therefore, TomEE must be configured for remote JMX access. See [http://tomcat.apache.org/tomcat-7.0-doc/monitoring.html#Enabling\\_JMX\\_Remote](http://tomcat.apache.org/tomcat-7.0-doc/monitoring.html#Enabling_JMX_Remote).

## Platform connector

HTTP APIs are SOAP-based Web services of hosted Web applications on the application server. See [Application server](#). These APIs are not accessible directly within AppWorks Platform because TomEE is not directly part of the Enterprise Service Bus of AppWorks Platform. The internal routing algorithms require registration of a Web service interface in a service container. The Platform connector acts as an internal proxy for the Web service interfaces deployed in TomEE.

The Platform service container forwards the requests it receives based on configuration information in the method set to an endpoint on the application server. It uses `com.cordys.internal.cluster.url` to determine the base URL for the endpoint. See [Configuring access URLs](#). The request is enriched with proper authentication and localization headers before sending it to the application server endpoint. The response is transferred in the reverse direction.

Any applicable Web service interface is automatically configured to be handled by the Platform service container. Therefore, no manual Web service interface registration is required. There are no specific configuration options for the Platform service container.

## Securing Apache TomEE

Many security-related tasks are common for TomEE and Tomcat installations.

### Hardening TomEE

For information about hardening TomEE, see the Apache Security Considerations at <http://tomcat.apache.org/tomcat-8.0-doc/security-howto.html>, the Tomcat HTTP Header Security Filter at [https://tomcat.apache.org/tomcat-8.0-doc/config/filter.html#HTTP\\_Header\\_Security\\_Filter](https://tomcat.apache.org/tomcat-8.0-doc/config/filter.html#HTTP_Header_Security_Filter), and the Add Default Character Set Filter at [https://tomcat.apache.org/tomcat-8.0-doc/config/filter.html#Add\\_Default\\_Character\\_Set\\_Filter](https://tomcat.apache.org/tomcat-8.0-doc/config/filter.html#Add_Default_Character_Set_Filter). AppWorks Platform has built-in CSRF Protection that you can use instead of the Tomcat specific solutions.

The [AppWorks Developer community](#) provides additional resources that you may find helpful.

## Secure Sockets Layer

To set up TomEE with Secure Sockets Layer (SSL), see Apache SSL Configuration at <http://tomcat.apache.org/tomcat-8.0-doc/ssl-howto.html>. Remember to set the hostname to the CN of the certificate.

In the file CATALINA\_HOME/conf/server.xml, set:

```
<Host name="server.acme.com" ...>
```

After you completed this task, follow the instructions to set the node URL to the HTTPS protocol. See [Configuring access URLs](#).

## Windows authentication

You can configure TomEE to perform Windows Authentication in different ways. See Windows Authentication How-To at <http://tomcat.apache.org/tomcat-8.0-doc/windows-auth-howto.html>. You can also use other authentication mechanisms with TomEE (Tomcat). See the specific product configuration.

The [AppWorks Developer community](#) provides additional resources that you may find helpful.

## Hardening the security of AppWorks Platform

To make the AppWorks Platform deployment more resilient to attacks, consider following the guidelines in this topic. These guidelines are meant to help administrators decide the configuration options to be chosen for business users. This topic is for configuring AppWorks Platform and its applications only, and not for the operational environment. Analyze your organization's security requirements and risk tolerance so that you can define the requirements for AppWorks Platform implementation.

Before you begin, consider reading the AppWorks Platform Security Whitepaper.

**Important:** Computers that are upgraded from previous versions of AppWorks Platform will retain the configuration values from the previous version. If you are an AppWorks Platform administrator, read this topic to make note of any security features to implement the AppWorks Platform deployment.

### *Operational environment*

AppWorks Platform works with other infrastructure service software, for example, databases, server operating systems, and Web application servers. Review the configuration of supporting software, services, and servers to ensure that AppWorks Platform is supported by secured systems. Ensure that the following considerations are met:

## **Up-to-date software**

After installing AppWorks Platform, ensure that you apply all the latest upgrades and patches. Also, apply all upgrades and patches for all other software on the system. This prevents attackers from exploiting known vulnerabilities.

## **Minimal footprint**

Use only verified and needed components. Remove all unused software components from the environment and unused AppWorks Platform application packages from the computer.

## **Least privileged**

AppWorks Platform runs under a service account. Harden this account as a service account according to operating system requirements.

## **Layered security**

- Enable used protocols only at the network level.
- Restrict database access by following the hardening recommendations of the database vendor.
- Ensure that virus scanners are up-to-date and running.

## **Consistency**

Ensure that all nodes use the same and correct time. This is because time settings effect the expiry of credentials, auditing, and logging.

## **Monitoring**

Monitor system resources, such as CPU, memory, and network, for DOS attacks and system overload. Review the audit information and log files periodically.

## **Web server**

Ensure to properly secure the TomEE Web application server, which is a critical part in the whole setup. For instructions, see Securing Apache TomEE.

The headers of SOAP responses include information about senders and recipients, which can be misused to send malicious requests or DoS attacks. To prevent this scenario, enable the Strip Outgoing Traffic feature. For instructions, see Removing header details from SOAP responses. Also consider Removing Stacktrace Details from SOAP Responses.

## **File system**

Make sure that only minimal file system permissions are applied, as follows:

- Provide only execution rights on command files.
- Provide access rights only to the service account group on the installation.
- Ensure that the service account and its group have no permissions on other parts of the computer.
- The Apache Tomcat process runs with a umask of 007 to maintain the permissions.

## Network

Minimize the risk of potential attacks on the components of a AppWorks Platform system by isolating the network. You can do so by controlling end-user access to different network areas and tiers.

Configure AppWorks Platform with a firewall (see Reverse proxy) and remove X-Forwarded-Host, X-Forwarded-Proto and Host headers from outside calls. Do not deploy AppWorks Platform in a DMZ environment.

Use only secured connections for all protocols, for example, HTTP, FTP, email, JMS, and JMX.

Set up transport layer security (TLS) on the enterprise service bus. Ensure that communications between AppWorks Platform service containers are encrypted. For information on configuring TLS, see SSL / TLS on ESB.

## Authentication

For authentication, AppWorks Platform uses OpenText Directory Services (OTDS). For information about the policies to secure OTDS authentication configuration and password, see OpenText Directory Services Installation and Administration Guide.

By default, authentication tokens on AppWorks Platform expire in 8 hours. To change this value, see Modifying expiration time of SAML Assertion.

## Authorization

AppWorks Platform uses role-based access control (RBAC). Ensure that authorization is least privileged and the role assignments in OTDS are reviewed periodically. For regular use, do not use the account that AppWorks Platform is installed with.

## Certificates

Certificates in AppWorks Platform Security Administration and Java keystores (for example, in the AppWorks Platform and CARS certificates installation directory) are used for installation validation and authentication. Monitor and review these certificates periodically.

Additionally, consider validating digital certificates, as explained in Digital Certificates Validation.

## Auditing

Ensure that auditing is enabled for the following artifacts:

- Authentication
- CAP
- CAP Security Settings
- Certificate Store
- ISVP
- OTDS Resource
- Trusted Application Publishers

For more information, see Auditing.



# Chapter 12

# Upgrading Java

If you want to upgrade Java to a later update with the latest security fixes, you must do the following:

**Note:** This procedure is required only if you have changed the installation directory of Java.

## To upgrade Java on Windows:

1. Stop the Apache TomEE and OpenText AppWorks Platform (<instance name>) services.
2. Uninstall the old version of Java (recommended).
3. Download and install the latest update of Java.
4. Update environment variables such as `PATH` and `JAVA_HOME` as per the new installation.
5. Update the properties of the shortcuts in the Start Menu (for example Management Console, Simple Client) with the path to the new installation.
6. Start Management Console and change the `java.home` property under Platform Properties to the path of the new installation.
7. Run `<TomEE_installdir>\bin\TomEE.exe`, click the Java tab, and then update the Java Virtual Machine to use the new installation.
8. Run `regedit.exe`, browse to `HKEY_LOCAL_MACHINE\SOFTWARE\WOW6432Node\Apache Software Foundation\Procrun 2.0\TomEE\Parameters` and update `LibraryPath` as per the new installation.
9. Start the OpenText AppWorks Platform (<instance name>) and Apache TomEE services.

## To upgrade Java on Linux:

1. Stop the Apache TomEE and OpenText AppWorks Platform (<instance name>) daemons.
2. Uninstall the old version of Java (recommended).
3. Install the latest update of Java.
4. Update the environment variables, such as `PATH` and `JAVA_HOME` as per the new installation.
5. Update `JAVA_HOME` in `<AppWorks_Platform_installdir>/bin/wcpenv.sh` with the path to the new installation.

6. Start Management Console and change the `java.home` property under Platform Properties to the path of the new installation.
7. Update `JAVA_HOME` in `<Apache TomEE_installdir>/bin/setenv.sh` with the path to the new installation.
8. Start the OpenText AppWorks Platform (`<instance name>`) and Apache TomEE daemons.

# Chapter 13

## Setting themes at the organization level

This topic provides an overview of the organization level theme.

This feature is available only if you can:

- Deploy applications to the organization space
- Publish projects to the organization space

The organization level theme enables incorporating certain custom styling features at the organization level. It is one of the key business requirements for certain deployment scenarios. Consider a cloud deployment model in which a single application deployed on a cloud is being used by multiple tenants. The organization level theme enables you to customize, test, and deploy organization-specific theme packs for each tenant in the cloud according to specific application requirements.

The theme-pack files for an organization are stored in the Web server. These files can be of type Javascript(js), Web pages (HTM, HTML), images, icons (gif, png, jpg, bmp), and cascading style sheets(css). In AppWorks Platform, the most commonly used theme-packs comprise of cascading style sheets, images, and, icons.

When you access a custom application developed using AppWorks Platform, the corresponding themes are applied to the application from the respective organization.

**Note:**

1. To add a web file to the user interface, do not specify the virtual root name or the organization name in the relative path. If the file is located at <AppWorks Platform\_installdir>/webroot/<Organization Name>/folder1/folder2/sample.jpg, the path in the user interface must be folder1/folder2/sample.jpg.
2. When using the web artifacts in the html pages, use them as given below:

```
webroot | - shared | - images | - cordys.gif | - organization | - dummyORG  
(your organization) | - images |- icon.gif | - example | - x.htm
```

### **without application.js**

```
<html>
  <body>

    
    <!-- relative to page folder -->
    

  </body>
</html>
```

### **with application.js (for example, any XForm)**

```
<html>
  <head>

    <import src="../wcp/application.js"/>
    <!-- only real relative reference -->

  </head>
  <body>

    
    <!-- relative to application root -->
    

  </body>
</html>
```

# Chapter 14

## Deleting the runtime content

You can use a command-line tool that archives or deletes content from the runtime repository of the AppWorks Platform components. Content is selected for archival or deletion based on the filter criteria configured in an ArchivingDefinition. The tool archives or deletes runtime content for the following components:

- Business Process Management
- Case Management
- Notification
- Task

After the AppWorks Platform baseline installation, the Cordys Archiving Engine package is deployed with other mandatory packages. This package extracts the files to the <AppWorks Platform\_installdir>/components/archiveframework folder. The tool and the definition required to execute it are provided as command-line tools in the bin folder:

- ArchivingDefinitionManagerTool
- DataDeletionTool

Content archival or deletion involves the following steps:

1. Creating a signed-in machine user in AppWorks Platform
2. Creating archive tables
3. Creating the ArchivingDefinition
4. Registering the ArchivingDefinition
5. Executing the tool

## Creating a signed-in machine user in AppWorks Platform

The signed-in user on the computer where the tool is run must exist in AppWorks Platform and also has the administrator role to run the tool. If the user does not exist in AppWorks Platform, you must create the user and assign administrator role to the user.

To find the signed-in user name, at the command prompt, type the `whoami` command.

## Creating archive tables

This section is applicable only if you want to archive the data and set ArchiveData to `true` in ArchivingDefinition.

To archive the data, you must create the archive tables.

### To create the archive tables

1. Navigate to `<CORDYS_HOME>\components\archiveframework\database\createscripts`.
2. Based on your database, run the scripts `BPM_CASE_ARCHIVAL_<your database>.sql` and `NOTIFICATION_ARCHIVAL_<your database>.sql`.  
For example, if your database is SQL Server, run the scripts `BPM_CASE_ARCHIVAL_MSSQL.sql` and `NOTIFICATION_ARCHIVAL_MSSQL.sql`.

#### Note:

Use tablespaces or file groups for better maintenance. Tablespaces are supported by Oracle and PostgreSQL and file groups are supported by MS SQL Server.

## Creating the ArchivingDefinition

Create an XML file with the ArchivingDefinition to capture the filter criteria for content archival or deletion. A sample structure of the ArchivingDefinition is shown below.

```

<ArchivingDefinition xmlns="com.cordys.archiving.framework">
    <FQN>com.archive.order.InactiveOrdersArchivingDefinition</FQN>
    <Description locale="en-US">This ArchivingDefinition is used to delete
inactive order instances (completed and older than 120 days).</Description>
    <TypeReference>
        <ArchivableDataTypeFQN>com.cordys.archiving.bpm</ArchivableDataTypeFQN>
        <Criteria>
            <BatchSize>10</BatchSize>
            <ArchiveData>false</ArchiveData>
            <Filter>
                <!-- Archivable data type-specific filter criteria -->
            </Filter>
        </Criteria>
    </TypeReference>
</ArchivingDefinition>

```

The following are the elements of the ArchivingDefinition:

Element	Description
ArchivingDefinition	Root element that is fixed with the namespace value.

Element	Description
FQN	Fully qualified name to uniquely identify the ArchivingDefinition.
Description	Short description of the ArchivingDefinition.
TypeReference	Repeatable element that helps to isolate the filter criteria of an archivable data type when multiple archivable type filters are configured.
ArchivableDataTypeFQN	<p>FQN of the archivable data type that is registered for the component.</p> <ul style="list-style-type: none"> <li>■ BPM: com.cordys.archiving.bpm</li> <li>■ Case: com.cordys.archiving.casemanagement</li> <li>■ Notification: com.cordys.archiving.notification</li> <li>■ Task: com.cordys.archiving.task</li> </ul>
Criteria	Non-repeatable element grouping of the filter criteria for an archivable data type.
BatchSize	Batch size to consider when archiving or deleting content for the archivable data type. OpenText recommends to use 10000.
ArchiveData	<p>Archive or delete data.</p> <ul style="list-style-type: none"> <li>■ true - archives the data. When set to true, you must create the archive tables. See <a href="#">Creating archive tables</a>.</li> <li>■ false - deletes the data</li> </ul>
Filter	Parent element in which the archivable data type-specific filter criteria is provided. See <a href="#">Filter criteria</a> for more details on archiving data type filter criteria.

## Registering the ArchivingDefinition

The ArchivingDefinitionManagerTool registers the ArchivingDefinition in the archiving registry.

### To execute the tool:

1. Ensure that the user executing the tool from the command prompt or a terminal client has the following:
  - The user must be present either in LDAP as a AppWorks Platform user or the OS identity of one of the existing users must be mapped to this user.
  - The user must be assigned the System Administrator role and must also be the administrator of the organization provided as input.
2. Set the CORDYS\_HOME environment variable in the command shell or environment to the folder where AppWorks Platform is installed.

- On Windows, set `CORDYS_HOME=<AppWorks Platform_installdir>`.
  - On Linux or Unix, set `CORDYS_HOME=<AppWorks Platform_installdir>`.
3. Change the current working folder as follows:
- On Windows, open the command prompt or terminal client and change the current working folder to `<AppWorks Platform_installdir>/components/archiveframework/bin`.
  - On Linux, grant the user read and execute permissions for the `ArchivingDefinitionManagerTool.sh` script file.

```
chmod 511 ArchivingDefinitionManagerTool.sh
```

4. Execute the batch or script file with the name of the organization to be archived or deleted as the parameter.

For Windows:

```
ArchivingDefinitionManagerTool.bat <path of the file containing  
ArchivingDefinition XML> <Organization DN> <create/delete>
```

For Linux:

```
./ArchivingDefinitionManagerTool.sh <path of the file containing  
ArchivingDefinition XML> <Organization DN> <create/delete>
```

The ArchivingDefinition is created in the archiving registry in AppWorks Platform.

## Executing the tool

The DataDeletionTool is used to archive or delete content based on the ArchivingDefinition.

### Before you begin:

1. Ensure that the user executing the tool from the command prompt or a terminal client has the following:
  - The user must be present either in LDAP of AppWorks Platform or the OS identity of one of the existing users must be mapped to this user.
  - The user must be assigned the System Administrator role and must also be the administrator of the organization provided as input.

### To execute the tool:

1. Set the `CORDYS_HOME` environment variable in the command shell or environment to the folder where AppWorks Platform is installed.

- On Windows, set CORDYS\_HOME=<AppWorks Platform\_installdir>.
  - On Linux or Unix, set CORDYS\_HOME=<AppWorks Platform\_installdir>.
2. Ensure that the proper connector JAR is set to the CLASSPATH.
  3. Change the current working folder as follows:
    - On Windows, open the command prompt or terminal client and change the current working folder to <AppWorks Platform\_installdir>/components/archiveframework/bin.
    - On Linux, grant the user read and execute permissions for the ArchivingDefinitionManagerTool.sh script file.

```
chmod 511 ArchivingDefinitionManagerTool.sh
```

4. Execute the batch or script file with the name of the organization to be archived or deleted as the parameter.

For Windows:

```
DataDeletionTool.bat <FQN of the ArchivingDefinition in the archiving registry>
<Organization DN> <number of threads>(optional, default is 1)
```

For Linux:

```
./DataDeletionTool.sh <FQN of the ArchivingDefinition in the archiving registry>
<Organization DN> <number of threads>(optional, default is 1)
```

The summary of the content deletion tool execution is captured in the ArchivingSummary.xml (suffixed with ArchivingDefinition and Timestamp) file located in the <AppWorks Platform\_installdir>/Logs.

## Filter criteria

The filter criteria configured in an ArchivingDefinition to archive or delete runtime data from the Business Process Management, Case Management, Notification, and Task components are explained below.

### Filter criteria for Business Process Management:

```
<FilterCriteria>
```

```

<ProcessModels>
    <Value>ProcessName</Value>
</ProcessModels>
<InstanceStatus>
    <Value>COMPLETE</Value>
    <Value>ABORTED</Value>
    <Value>TERMINATED</Value>
    <Value>REPLACED</Value>
    <Value>SKIPPED</Value>
    <Value>OBSOLETE</Value>
    <Value>SUSPENDED</Value>
</InstanceStatus>
<TimeInterval>
    <Age>(in days)</Age>
    <Interval>
        <StartTime>2010-01-01T00:00:00.0</StartTime>
        <EndTime>2010-01-01T00:00:00.0</EndTime>
    </Interval>
</TimeInterval>
<InstantiationUser>
    <Value>cn=testUser,cn=organizational
users,o=system,cn=cordys,cn=defaultInst,o=opentext.net</Value>
</InstantiationUser>
<ArchiveAssociatedInstances>true/false</ArchiveAssociatedInstances>
</FilterCriteria>

```

### Request parameters:

<b>Filter</b>	<b>Mandatory</b>	<b>Description</b>	<b>Accepted Input Values</b>
ProcessModels	No	Multiple process model names can be provided for archival or deletion.	FQN of the BPMs.
InstanceState	No	Status of the process instances.  If ArchiveAssociatedInstances is false, then multiple statuses can be provided for archival or deletion. If ArchiveAssociatedInstances is true, then only the COMPLETE status can be provided.	When ArchiveAssociatedInstances is false, you can provide the following statuses: <ul style="list-style-type: none"> <li>■ COMPLETE</li> <li>■ ABORTED</li> <li>■ TERMINATED</li> <li>■ REPLACED</li> <li>■ SKIPPED</li> <li>■ OBSOLETE</li> <li>■ SUSPENDED</li> </ul>

<b>Filter</b>	<b>Mandatory</b>	<b>Description</b>	<b>Accepted Input Values</b>
			When ArchiveAssociatedInstances is true, you can provide only the COMPLETE status.
TimeInterval	Yes	Age - The instances before the provided age are archived or deleted. TimeInterval - The instances between this time interval are archived or deleted.	You must provide either Age or TimeInterval. It must be a non-negative number.
InstantiationUser	No	Multiple process instantiation user DNs can be provided.	User DNs.
ArchiveAssociatedInstances	No	Identifies whether to consider subprocesses for archival or deletion.	true/false; the default value is false. If 'ArchiveAssociatedInstances' is true, you can provide only the COMPLETE status in 'InstanceStatus' filter.

### Filter criteria for Case Management:

```

<FilterCriteria>
  <CaseModel>
    <Value>CaseName</Value>
  </CaseModel>
  <InstanceState>
    <Value>COMPLETED</Value>
    <Value>ABORTED</Value>
    <Value>TERMINATED</Value>
    <Value>SKIPPED</Value>
    <Value>OBSOLETE</Value>
    <Value>SUSPENDED</Value>
  </InstanceState>
  <TimeInterval>
    <Age>(in days)</Age>
    <Interval>
      <StartTime>2010-01-01T00:00:00.0</StartTime>
      <EndTime>2013-01-01T00:00:00.0</EndTime>
    </Interval>
  </TimeInterval>
</FilterCriteria>

```

```

</Interval>
</TimeInterval>
<InstantiationUser>
  <Value>cn=testUser,cn=organizational
users,o=system,cn=cordys,cn=defaultInst,o=openText.net</Value>
</InstantiationUser>
<ArchiveAssociatedInstances>true/false</ArchiveAssociatedInstances>
</FilterCriteria>

```

**Request parameters:**

<b>Filter</b>	<b>Mandatory</b>	<b>Description</b>	<b>Accepted Input Values</b>
CaseModels	No	Multiple case model names can be provided for archival or deletion.	FQN of the cases.
InstanceStatus	No	Status of the case instances.  If ArchiveAssociatedInstances is false, then multiple statuses can be provided for archival or deletion. If ArchiveAssociatedInstances is true, then only the COMPLETE status can be provided.	When ArchiveAssociatedInstances is false, you can provide the following statuses: <ul style="list-style-type: none"> <li>■ COMPLETE</li> <li>■ ABORTED</li> <li>■ TERMINATED</li> <li>■ REPLACED</li> <li>■ SKIPPED</li> <li>■ OBSOLETE</li> <li>■ SUSPENDED</li> </ul> When ArchiveAssociatedInstances is true, you can provide only the COMPLETED status.
TimeInterval	Yes	Age - The instances before the provided age are archived or deleted. TimeInterval - The instances between this time interval are archived or deleted.	You must provide either Age or TimeInterval. It must be a non-negative number.
InstantiationUser	No	Multiple case instantiation	User DNs.

<b>Filter</b>	<b>Mandatory</b>	<b>Description</b>	<b>Accepted Input Values</b>
		user DNs can be provided.	
ArchiveAssociatedInstances	No	Identifies whether to consider the associated processes for archival or deletion.	true/false; the default value is false. If 'ArchiveAssociatedInstances' is true, you can provide only the COMPLETED status in 'InstanceStatus' filter.

### Filter criteria for Notification/Tasks:

```
<FilterCriteria>
    <FilterType>STANDALONE</FilterType>
    <TimeInterval>
        <Age>0</Age>
    </TimeInterval>
</FilterCriteria>
```

### Request parameters:

<b>Filter</b>	<b>Mandatory</b>	<b>Description</b>	<b>Accepted Input Values</b>
FilterType	Yes	Type for filtering the notifications or tasks.	STANDALONE
Age	Yes	The instances before the provided age are archived or deleted.	It must be a non-negative number.



## Part II

# **Business Activity Monitoring**

# Chapter 15

## Business monitoring

The purpose of monitoring your business is to track trends, compare the results as against historical evidence and arrive at conclusions so that decision making becomes easier.

Monitoring your business helps you to cut down unnecessary costs, invest properly and make better profits. Business monitoring helps you to understand market scenarios, and strengthen your business processes so that your business not only thrives in the market but also makes an impact on it.

Using AppWorks Platform, you can monitor your business processes at all levels - process, activity, and instances. Process monitoring is used to track instances of business processes through their life span. However, monitoring is a performance overhead, but could be critical in some cases. You can continuously monitor your processes, check for failures if any, and address them in time so that all your business processes run unhindered.

Business process monitoring in AppWorks Platform comprises:

- [Adding a new process filter](#)
- [Performing operations on deployed process models](#)
- [Archiving process instances](#)

At design time, you can turn on monitoring for a particular process or activity using the **Enable Monitoring** feature.

### Levels of monitoring

Depicted below are the various scenarios in which monitoring can be set at the process, instance and activity levels.

Scenarios of process monitoring if the monitoring attribute is set at the instance level:

If the monitoring attribute is...	Then...
Enabled at the instance level	Process level property is ignored, but any activity level property is considered for monitoring.
Disabled at the instance level	No monitoring is possible at any level.

Scenarios of process monitoring if the monitoring attribute is not set at the instance level:

If the monitoring attribute is...	Then...
Enabled at the process level (the <b>Enable Process Monitoring</b> option is selected)	Activity level property is considered for monitoring.
Disabled at the process level (the <b>Enable Process Monitoring</b> option is cleared)	No monitoring is possible at any level.

If an attribute is defined neither at the instance level nor at the process level then the **Enable Process Monitoring** property in the Business Process Management Service configuration, which is the default behavior interface is considered. For any monitoring and crash recovery option to work, it is mandatory that you enable it first in the Process Engine tab of the **Business Process Management Service** properties. See [Modifying monitoring and crash recovery settings](#) and Business process management service properties interface in the *AppWorks Platform Advanced Development Guide*.

If the process instance is running in debug mode, the **Enable Process Monitoring** property at the activity level is retained and activity data is always published.

You can select or clear the **Enable Process Monitoring** option for activities that are defined inside a transaction of a long-lived process.

## Key concepts

The following is a set of concepts related to BAM:

### Business event response

Some business scenarios pose a threat or provide opportunities to business. In such a scenario, you must take action to handle the threats, or take advantage of the opportunities available. Business event response enables you to warn or alert users about such a scenario.

For example, an insurance company can build a reputation of being customer friendly while advertising a Service Level Agreement (SLA) that they process any claim within a period of five working days. However, if the company fails to meet the advertised SLA even for a few customers, it is a threat to the reputation of the company. Such a threat can be described as a business event response.

### Business measure

Use business indicators to measure some aspects of a business. These measures can be graph or KPI (Key Process Indicators) and are exposed as Web services with internal implementation of query against the business data.

An insurance company can build a measure to know all the claims which were approved the previous week.

### Process monitoring object

A process monitoring object is a set of associated attributes from a business process and related contextual application Web services. You must monitor it as a logical group. Each process monitoring object is persisted and used for the purpose of analysis by creating business measures on it.

A process monitoring object is built in the AppWorks Platform business process context, and nonprocess Web services can be defined to invoke on process events.

An insurance company has multiple phases in its claim handling process, such as preprocessing, review, and approval. Due to increased frauds in the industry for insurance claims, the industry may start maintaining a database of people who are suspected of making fraudulent claims. In a short interval, without changing its claim handling process, a company might want to alert an investigation officer when a claim is made by these people. For such a case, companies can build a process monitoring object called `Claim_` `Investigation`, which can have attributes from the claim handling process like customer name, claim amount, SSN, claim date, and claim ID. There can be attributes from Web service called `getFraudHistory(SSN)`, which is not part of the claim handling process. You can decide at which process event this Web service must be called in.

### **Key Performance Indicator**

Use the Key Performance Indicator (KPI) to monitor and capture critical information pertaining to a business process or activity. KPIs track any exceptions in an enterprise and trigger automatic actions to take immediate corrective actions. You can trigger the following actions:

- Send an email notification
- Invoke a new business process
- Call a Web service

These enhance business operations through continuous process improvement.

An insurance company can define a KPI to track the number of claims approved in the previous week, and can send a notification to the concerned authority if the number of claims approved is less than the minimum level.

### **Enterprise Data Object**

Enterprise Data Object (EDO) is the persistent object for data of enterprise applications like SCM, and CRM. There is a possibility that an enterprise would like to monitor these data objects, but they do not have any relation with any AppWorks Platform business processes. Enterprises can reuse BAM infrastructure for monitoring by synchronizing these data objects to the BAM system and then build business measures on them.

**Note:** Dashboards and KPI monitoring services can be built on EDO, but business event responses cannot be built because they require event support from enterprise applications. This is not supported by many applications. BAM supports EDOs, which are synchronized by using MDM component to the same hub database where the BAM hub is configured.

For a CRM system, `Customer_Details` might be a database table, and when this table is synchronized with the BAM Hub database through the MDM synchronization component, this table can be considered an Enterprise Data Object in the BAM context.

# Business activity monitoring overview

## Introduction

Business Activity Monitoring (BAM) provides an end-to-end, integrated solution for closed-loop monitoring, business event response management, and improvement of business processes. It empowers you with the global visibility of the business processes and the increased responsiveness to them, enabling you to take proactive action based on the actual information. It helps you monitor your company's business processes, identify failures or exceptions, and address them real-time.

BAM provides real-time alerts and notifications on critical events and exceptions, and a centralized performance dashboard. It offers a drill-down analysis to discover trends, patterns, and bottlenecks in enterprise performance.

In addition, BAM also supports monitoring of non-process data (Enterprise Data Objects) that leverages the same functionality as available for business processes. It provides real-time visibility into the effectiveness of overall business performance by monitoring key aspects and alerting users about deviations. This helps the firms to react with a level of agility which was previously unavailable through conventional approaches to business monitoring.

## Key Features

BAM has the following key features:

- Near real-time process information collection infrastructure
- Near real-time continuous process monitoring using process content
- Ability to combine contextual information of process for monitoring
- Near real-time monitoring based on process events
- Automatic resolution mechanism in case of exception situations observed during monitoring
- Monitoring external information sources
- Out-of-the box, pre-configured dashboards
- Ability to design custom dashboards
- Drill-down between views for root-cause analysis within custom dashboard
- Visualization through Microsoft Office Excel
- Operational data can be consumed by any Business Intelligence (BI) tool such as Cognos, Business Objects, Jasper and so on, for further analysis

## Key Concepts

### **Architecture**

The Architecture of BAM comprises various components, their properties, and the relationships between them. It enables you to deploy solution that can define, monitor, analyze business process and related external sources, and take appropriate actions in automated manner when required.

### **Administration**

Administration in BAM is a configuration process that allows you to configure the filtering options to synchronize and upload data from the Process database to the BAM database. Filtration of data has a considerable impact on the BAM database.

### **Artifacts**

Various artifacts help in displaying rich BAM capabilities from near real-time monitoring to periodic monitoring and visualization of data through dashboards. BAM has the following components:

- For near real time monitoring: Business Event Response Composer
- For periodic monitoring: Monitoring Object Modeler, Business Measure Editor.
- For Visualization: User Interface (Dashboard Designer)

### **Dashboards**

Dashboards in BAM depict the outcome of performance analysis in the form of graphs, dials and so on. Dashboards are of two types, Standard and Custom. The BAM Standard Dashboard helps you to analyze the information related to the process performance by providing some standard views on the process and activities. The Custom Dashboard displays the dashboards created through User Interface for a Business Measure or a KPI.

## **Business Activity Monitoring architecture**

Business Activity Monitoring (BAM) helps organizations identify critical threats and opportunities, and take corrective or preventive measures to counter them. To identify these threats and opportunities, business activities are monitored and analyzed with the help of predefined rules, and by invoking predefined actions.

To develop an effective BAM solution, define:

- Data or events, exceptional situations (objects to be monitored).
- Situations or instances when monitoring is needed.
- Actions to be taken in exceptional situations.
- Trends to observe from monitoring.

The architecture of BAM enables organizations to effectively deploy solutions, which can define, monitor, and analyze the business process activities and take appropriate actions in an automated manner, whenever required.

The section on the architecture of BAM contains:

- [Key concepts](#)
- [Information architecture](#)
- [Design architecture](#)
- [Deployment architecture](#)
- [Runtime architecture](#)
- [Architecture capabilities](#)

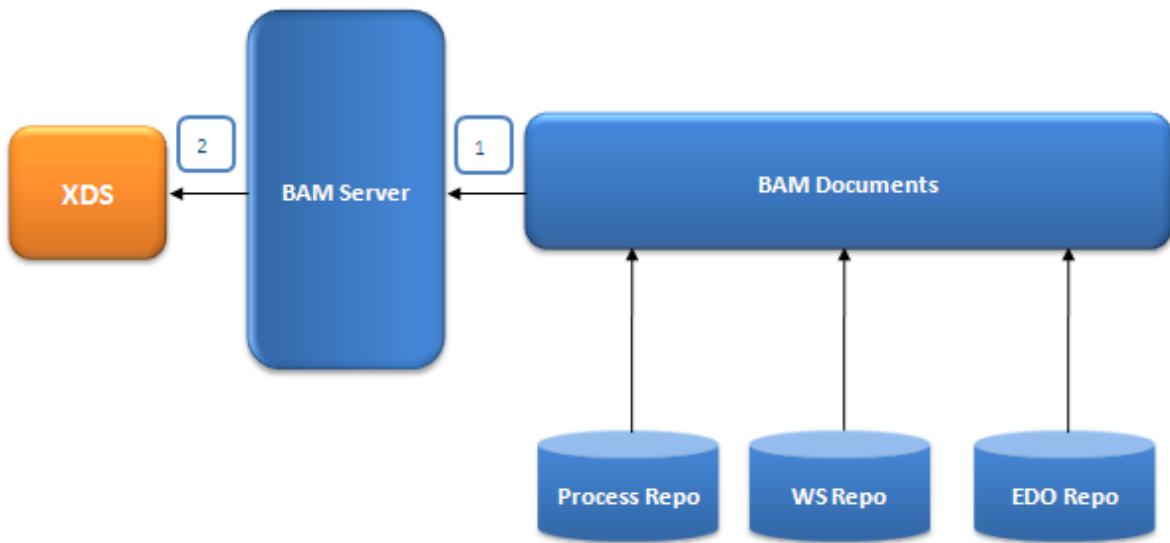
## Architecture capabilities

Monitoring capability is transcending the single business process boundary. BAM architecture enables you to monitor enterprise applications and provides the following capabilities:

Capabilities	Description
Proactive surveillance on process events	Proactively monitors different aspects of business process and associated application services.
Closed-loop monitoring	<ul style="list-style-type: none"> <li>■ Defines different kinds of corrective and preventive actions by invoking another business process, triggering emails, or invoking Web services.</li> <li>■ Defines business process flow based on real-time data from BAM Web services.</li> </ul>
Single dashboard and root-cause analysis	<ul style="list-style-type: none"> <li>■ Defines multiple dashboards on a single dashboard and drills down to different views.</li> <li>■ Performs the root-cause analysis by drilling down to specific business process instance data and the execution flow.</li> </ul>
Customization per organization	Enables an organization-level customization of the BAM content.
Extensible architecture	Consumes and supplies standard Web services. It leverages the AppWorks Platform rule engine to plugin any kind of action capabilities.

## Design architecture

Use the CWS design-time environment to define BAM design-time artifacts such as monitoring objects, business measures, KPI monitoring services, business event responses, and dashboards. In the earlier version of BAM, dashboards were created using the BAM workbench. Now, dashboards are created using the AppWorks Platform XForm designer. The design-time environment flow is as follows:



**Note:** CWS is the launch pad for the editors. Use it to model the BAM artifacts. The following editors are part of BAM:

Editor name	Description
Process Monitoring Object Editor	Process monitoring object on a business process, and the associated Web services attributes.
Business Event Response Editor	Business event responses, and the corresponding actions.
Business Measure Editor	SQL query-based Web services on the process monitoring object, and the EDO with the time period.
KPI Editor	Monitoring of KPI by defining parameters such as ranges, actions, and schedules.
Dashboard Designer	Earlier, the dashboards were defined using the BAM workbench. Now, they are defined using the AppWorks Platform XForms

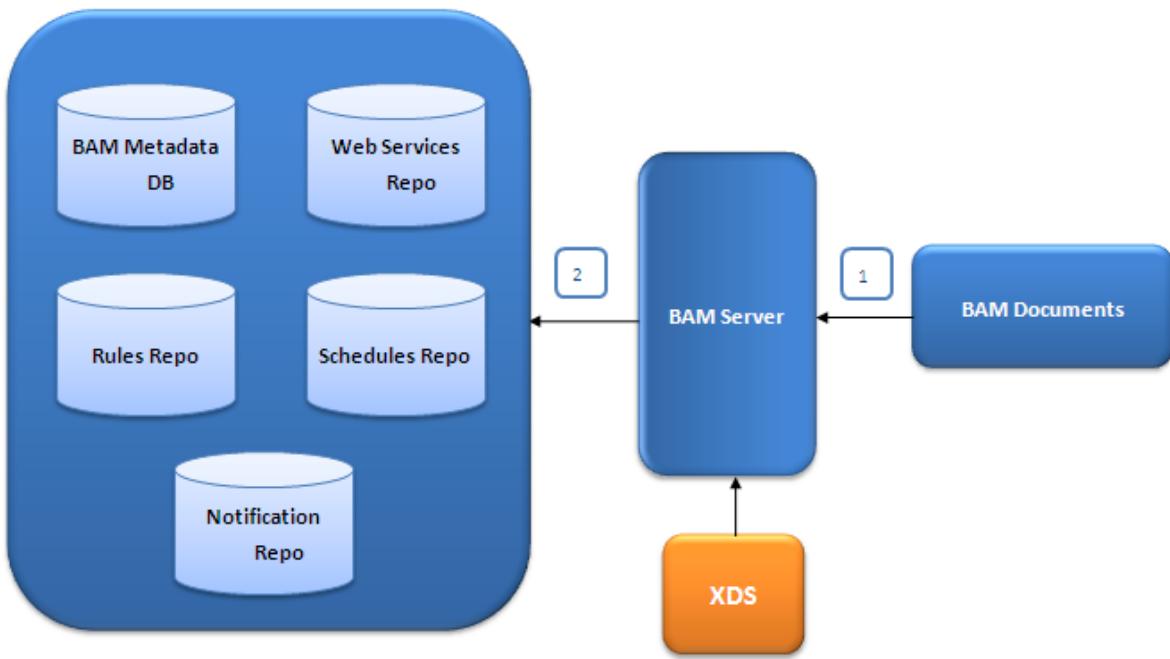
<b>Editor name</b>	<b>Description</b>
	designer. To define multiple views, dashboards are created using the business measure and KPI monitoring services. Select different layouts and drilldown different views.
BAM List views	BAM artifacts in the CWS artifact viewer. Use features such as tagging and search, to group the artifacts.

The following components are part of the design-time environment flow:

<b>Component</b>	<b>Description</b>
Processes Repository	Business processes that are published from the AppWorks Platform Business Process Management environment are stored in the process repository. The BAM editors retrieve business process details from here.
Web Services Repository	Web services in AppWorks Platform environment are stored in the Web services repository. BAM editors retrieve the stored Web services details from here.
Enterprise Data Objects Repository	BAM supports EDO, and these are synchronized with the BAM Hub database using the Cordys MDM component. Metadata of all published MDM entities is available in the EDO repository. This information is used in the business measure editor.
BAM Design-time Repository	Definitions of BAM artifacts that are created by the BAM editors, are stored in the XDS repository. BAM uses XDS as a design-time repository.

## Deployment architecture

Use the BAM deployment environment to deploy and publish the definitions of BAM artifacts created using editors in design time. The definitions of BAM artifacts are packaged as applications, deployed in the target environment, and published or configured for an organization. The deployment time flow is as follows:



BAM artifacts are selected from the artifact viewer and published or configured for an organization. The following runtime artifacts are created once the publish or configuration process is complete.

### **Process monitoring object**

A process monitoring object is provided as a set of attributes, with different data types. While defining queries on the business measure editor, you can assume that each attribute has a dedicated column in the database table reserved for each process monitoring object. However, internally all the process monitoring objects are stored in a generic database schema, since dynamic table creation is not permitted in many customer deployment environments. When a process monitoring object is published, its attributes, data type, and column map information are stored in the objects metadata repository in the BAM database.

### **Business measure**

Each business measure is published as a separate Web service and attached to the BAM monitoring service, available in that particular organization. This enables better enforcement of data security, as each BAM monitoring service processes data or events related to that organization only, though they can access the common BAM database.

Every business measure contains an SQL query as an internal implementation. While publishing the business measures, queries defined on the Process Monitoring Objects and EDOs are compiled to the SQL queries. These can be run on the BAM objects database, and EDO tables.

### **KPI monitor**

Monitoring of KPIs is configured through the KPI editor, where you can specify the time interval for the KPI data to be monitored. This is specified as:

- The parameters, ranges, and actions that must be triggered when a KPI value is part of that range.
- The scheduled intervals to monitor the KPIs. Each KPI monitor is created as a separate Web service from the data security perspective, making it easier to integrate with external tools.

When a KPI monitor is published, a Web service is created in the AppWorks Platform Web service repository and attached to the BAM monitoring service in that organization. The monitoring schedule is deployed with the AppWorks Platform scheduler. Rules are created with the defined ranges and actions. For firing of rules on the insertion of KPI monitor objects, with range definitions, a template is created for each KPI monitor. For the **type email** action, a Web service is created based on the email template.

### **Business event response**

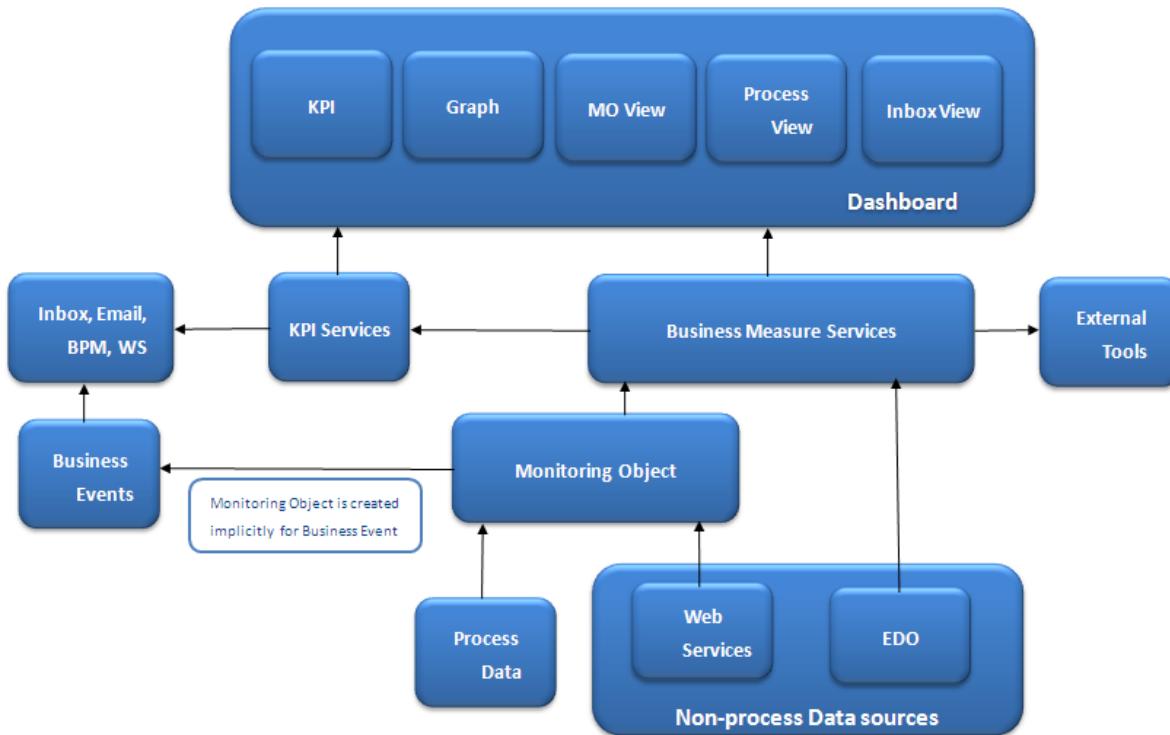
From the data architecture perspective, an implicit process monitoring object is created for each business event response. Based on the event condition definition, the corresponding event metadata is stored in the BAM database. For working of event rules, a template is created based on the business event response, with associated rules and actions. For the **type email** action, a Web service is created for each email template.

### **Dashboard**

Dashboard definition is now part of AppWorks Platform XForms designer. Once the dashboard is created by placing it on the business measures and KPI composite controls, assign the XForm to any required role to view it. You can assign the dashboards to any organization role. Therefore, access to the BAM dashboards is not dependent on any BAM application role. You require only the **everyOneInBAM** role to run the business measures, which are a part of the dashboard views.

## **Information architecture**

The BAM application supports multiple data providers and enables multiple consumers. Various sources of information, the processing mechanism, and consumers of the information are part of the information flow:



The BAM Information architecture flow is as follows:

### Information sources

- **AppWorks Platform business processes:** The message map elements of all the published or unpublished business processes can be the process monitoring object attributes. Business process data is synchronized from the Process database to the BAM database.
- **Web services:** All the AppWorks Platform Web services elements can be process monitoring object attributes. External Web services can be consumed using the UDDI connector.
- **Enterprise data objects:** All the Enterprise Data Objects (EDO), which are synchronized through Cordys MDM to the BAM hub database, can be used to build business measures.

### Information processing mechanism

- **Process monitoring object:** A process monitoring object is a composite object, which acts as container of business process and Web service attributes. It is filled and persisted when the process events happen.
- **Business measure:** Consists of all the query based Web services, which can be built either on process monitoring object or EDO. It supports the building of Web services, which can be directly consumed as KPI or graphs.
- **KPI monitoring services:** KPIs can be monitored at scheduled intervals and specified actions can be fired when the KPI is not in the specified range.

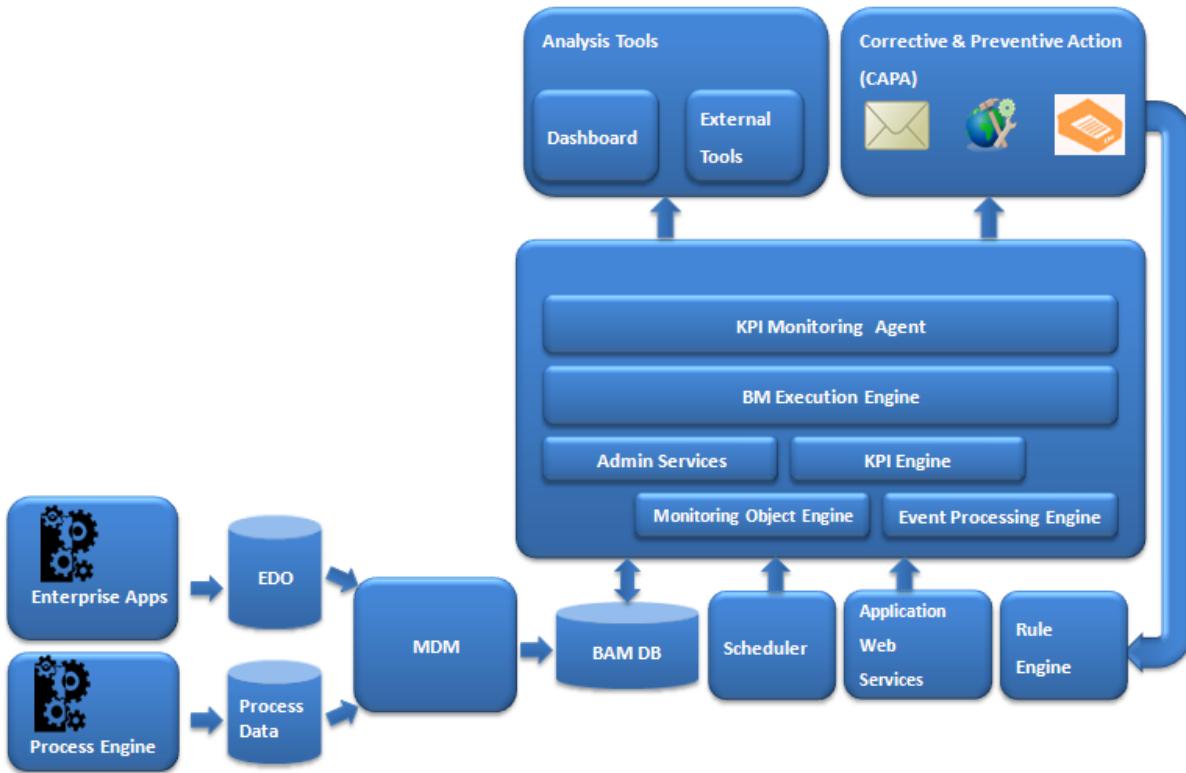
- **Business event responses:** Condition events can be built based on business process or Web services data, or business process or activity lead time. Specified actions can be fired when the conditions are met.

## Information consumers

- **AppWorks Platform BAM dashboard:** Create the dashboard using the XForm designer. You can:
  - Visualize graphs using the business measures.
  - Visualize KPI trends using KPIs.
  - View the process monitoring objects by default.
  - Launch specific process instance views.
  - Define the drilldown from one view to another.
- **External tools:** Any tool which can consume Web services can also potentially consume BAM Web services. Business measures and KPI monitoring services are published as AppWorks Platform Web services.
- **Action consumers:** When a business event response or KPI monitoring is triggered, the following actions can be defined:
  - Send an email, using the rich features of the AppWorks Platform Email template.
  - Send an alert to AppWorks Platform inbox.
  - Invoke another business process.
  - Invoke a Web service.

## Runtime architecture

From a real-time monitoring perspective on BAM, the right data should be available at the right time and harmonized to process it and relevant actions must be triggered. Different aspects of BAM runtime architecture are depicted in the following diagram:



## 1. Data collection and harmonization

AppWorks Platform BAM has three sources of data:

- AppWorks Platform business process data
- Application web services
- Enterprise data objects

AppWorks Platform business process data and web services data is harmonized as Process Monitoring object.

- Enterprise Data Object: The EDOs are synchronized using the Cordys MDM component. The Enterprise application database acts as the MDM spoke node and the BAM database acts as the MDM Hub node. When any changes happen in the application database, data is synchronized with the corresponding tables residing in the BAM database - data is synchronized from the spoke node to the Hub node.

## 1. Event processing

Process events are identified by the BAM server based on process state transitions in the database. When a process event occurs, The event processing engine determines whether it is a trigger event for any Business Event Response. A pre-processing of the event condition is done using Xpath expression evaluation to check if the event condition will be met in the rule engine and whether the event condition is going to be met then Business Event Response object is inserted in BAM repository to trigger rules and corresponding corrective and preventive actions (CAPA).

When the temporal (time-based) events are defined on process or activity lead time, an event monitor is started, which is embedded schedule in BAM server, with the maximum lead time defined as event monitoring time to check completion of process or activity. This concept is also known as proactive surveillance, where the system proactively does surveillance on happening of certain events.

You can define temporal condition along with the data conditions.

## **2. Business measure execution**

To execute Business Measure web services, fetch and execute the corresponding SQL query from the BAM meta database.

## **3. KPI monitoring**

The monitoring agent (Scheduler) monitors KPIs by invoking the KPI web service with parameters specified during design-time in the KPI Editor. The KPI web service is hosted in the BAM server. The BAM server invokes corresponding the Business Measure web services and output is given to the KPI engine where the expression is evaluated to calculate the KPI value. The KPI Monitor object, filled with the KPI value, is given to the rule engine, which evaluates the corresponding rules against the defined KPI ranges and corresponding corrective or preventive actions (CAPA) are triggered. KPI values are stored in the BAM repository for trend analysis.

## **4. Admin services**

Different admin services such as publish, unpublish, delete, and so on are available as part of BAM Admin services.

## **5. Dashboard**

The BAM dashboard charts are built using fusion charts, flex based technology created from the AppWorks Platform XForms designer. On drill-down from one view to another, parameters pass from the x-axis of source view to target view. Users can also drill-down to the default view of the Process Monitoring object and from there they can navigate to process instance graphical or grid view.

# Chapter 16

## Process Instance Manager

The Process Instance Manager (PIM) is a powerful interface that you can use to control process instances, gather information on processes, and examine minute aspects of a process instance.

The PIM can be accessed from the *My Applications* App Palette. Click  (Process Instance Manager) in the *My Applications* App Palette. The *Process Instance Manager* appears displaying a list of business processes that have process instances.

PIM displays two different views on the same data:

- Summarized View
- All Instances View

Both the views display the process instances in a table, depending upon your selection criteria. From the table, you can navigate to detailed instance data and perform operations to influence the running of a process instance.

Click  (Options) to switch between the two views.

You can use the PIM to:

- [Viewing process instances](#)
- [Viewing a process instance in graphical view](#)
- [Viewing subprocesses of a process instance](#)
- [Performing operations in the Process Instance Manager](#)
- [Viewing the message and error text of a process instance](#)
- View the Message Map of the process instance - See [Viewing a Process Instance Message Map](#) in the *AppWorks Platform Advanced Development Guide*.
- [Skipping an activity](#)
- [Resetting a process instance from graphical mode](#)
- [Monitoring rules in the Process Instance Manager](#)

**Note:**

- For process instances that are imported from an archive, the following operations cannot be performed: **Suspend**, **Resume**, **Terminate**, and **Show the Graphical**

**View.** See [Loading an archive](#).

- Configuring task level access - Depending on the access permissions assigned to your user role, certain options in the PIM screens may not be available to you for use. See [Configuring the access to task parts from Process Instance Manager](#).

## Modifying monitoring and crash recovery settings

This topic describes the procedure to modify monitoring and crash recovery settings for published business processes.

### Before you begin:

- You must have the Process Administrator role in AppWorks Platform Business Process Engine ISV package to modify the monitoring and crash recovery settings for a published business process.

**To modify the monitoring and crash recovery settings for a published process, do one of the following:**

### From deployed process models

1. On the **Welcome** page, click **Deployed Process Models**.

The Deployed Process Models dialog box opens. See [Performing operations on deployed process models](#).

2. Right-click the process to modify monitoring and crash recovery settings and select **Customize Configuration**.

Alternatively, select the check box of the deployed process, and click the enabled **Customize Configuration** on the toolbar. The *Monitoring and Crash recovery settings* window opens. See [Monitoring and crash recovery interface](#) for more information on the fields that appear on the *Monitoring and Crash recovery settings* dialog box.

3. Select the appropriate options for modifying monitoring and crash recovery settings in the *Monitoring and Crash recovery settings* window and click **Save**.

### Through ExecuteProcess Web service

```
<ExecuteProcess xmlns="http://schemas.cordys.com/bpm/execution/1.0">
  <type>[definition | instance]</type>
  <receiver>[ Unique identifier to the process ]</receiver>
  <source>[Custom instantiation source]</source>
  <modelSpace>[Organization | ISV]</modelSpace>
  <monitor level="BASIC | INTERMEDIATE | COMPLETE | COMPLETE_ON_ERROR">[ON | OFF]
  </monitor>
  <crashRecovery>[ ON | OFF ]</crashRecovery>
  <priority>[ 1 | 2 | 3 | 4 | 5 ]</priority>
  <message>[Any XML message (if required) by the process]</message>
</ExecuteProcess>
```

In the above ExecuteProcess Web service, consider the following:

```
<monitor level="BASIC | INTERMEDIATE | COMPLETE | COMPLETE_ON_ERROR">[ON | OFF]</monitor>
```

- \* BASIC denotes the process status
  - INTERMEDIATE denotes Start and End messages.
  - COMPLETE denotes Store complete process level information.
  - COMPLETE\_ON\_ERROR denotes Store complete process level information when the process aborts.

You have successfully modified the monitoring and crash recovery settings for a published business process.

## Viewing process instances

The Process Instance Manager (PIM) enables you to view individual process instances and a summarized view of each published business process. It depicts the total number of instances and the number of instances per status. It provides an overview of the running instances and their states, enabling you to take immediate action to suspend a running or a waiting instance, or restart an aborted instance.

You can view any of the following from the PIM:

- A single process from the business process modeling environment (design time).
- A single process from the business process deployment environment (run time).
- Summary of multiple process instances.

### To view the process instances of a single process from the business process modeling environment:

1. Select a starting point and click  (Business Process Model) to open an existing business process model. See Creating a document in the *AppWorks Platform Advanced Development Guide*.
2. Right-click in the business process modeler, and select **Business Process Execution > Show Process Instances**.  
The Instances by Process Definition window opens displaying the process instances of the selected process model. See [Instances by Process Definition interface](#).

### To view the process instances of a single process from the business process deployment environment:

1. On the Welcome page, click  (Process Instance Manager).  
The Process Instance Manager window opens with a list of business processes that have process instances. See Process instance manager interface in the *AppWorks Platform Advanced Development Guide*.

2. Click  (Find) to filter the process instances.  
The *Filter Process Instances* window opens. See *Filter process instances interface* in the *AppWorks Platform Advanced Development Guide* for information on the fields on this window.  
If you do not specify the filter criteria for process instances, the default filter options are considered.
3. Select the appropriate status of the business process.  
See *Statuses of a Process Instance* in the *AppWorks Platform Advanced Development Guide* for information on the statuses.
4. Click **OK**.  
The process instances that match the specified filter criteria are displayed.
5. Click the link that indicates the number of instances to drill down to the details.  
See [Monitoring and managing process instances](#) for more information on the drill down details.

**Note:**

- You can also download the process instances data into a Microsoft Excel file. To do so, click  (Export data to Excel) in the toolbar of the **Process Instance Manager** window.
- The Process Instance Manager always displays the complete count of the process instances in the current organization irrespective of the process authorization. However, on drill down you can only view the details of the process instances for which you are authorized. See *Defining runtime security* in the *AppWorks Platform Advanced Development Guide* for more details.

## Monitoring and managing process instances

The Process Instance Manager (PIM) displays information on the process instances, displayed based on the specified filter criteria. This view helps the process administrator to view running instances and their state, take decisions to suspend running or waiting processes, or restart ended instances of specific business processes. This view helps you to monitor and manage process instances.

**Before you begin:**

You must have the Process Administrator role in AppWorks Platform Business Process Engine application package to view instances of business processes by definition.

**To monitor and manage process instances:**

1. On the Welcome page, click **Process Instance Manager**.  
The *Process Instance Manager* window opens.
2. Perform required operations in the Process Instance Manager window. See [Performing operations in the Process Instance Manager](#).

3. Click **Find** on the toolbar to filter the process instances. If you do not specify the filter criteria for process instances, the default filter options are considered.  
The Filter Process Instances window opens. See *Filter process instances interface* in the *AppWorks Platform Advanced Development Guide*.
4. Type the selection criteria in the *Filter Process Instances* window.
5. Click **OK**.  
All process instances that match the specified criteria appear.
6. Click the hyperlinked attribute in the **Total** column to view the total list of process instances or the list of process instances which have a specific status.  
The *Instances by Process Definition* window opens. See *Instances by Process Definition interface* for more details about the fields and the operations that you can perform in this page.

## Viewing a process instance in graphical view

While executing a business process model, the process instance may go through a different set of execution states while executing the activities. Viewing the graphical representation of a process instance allow the users to visualize the progress of the process instance along with its activity details.

The graphical view shows the actual state of a process. For all the executed activities, the corresponding state icon is depicted at the bottom-right corner.

### Before you begin:

You must have the Administrator role.

### To see the graphical view of a process instance:

1. On the Welcome page, click  (Process Instance Manager).  
The Process Instance Manager window opens. See *Process Instance Manager Interface* in the *AppWorks Platform Advanced Development Guide*.
2. Click  (Find) on the toolbar.  
The Filter Process Instances window opens. If you do not specify the filter criteria for process instances, the default filter options are considered based on which the process instances are displayed.
3. Type the selection criteria. See *Filter Process Instances Interface* in the *AppWorks Platform Advanced Development Guide*.
4. Click **OK**.  
All process instances that match the specified criteria are displayed.
5. To view the process details, click the instances with the status.  
The *Instances by Process Definition* window opens.
6. Select the process instance, and click  (Show Sub-Processes).  
The Sub-Processes pane opens.

7. Click  (Show Graphical View) on the toolbar or right-click the process instance and select **Show Graphical View**.  
Alternatively, right-click the process instance and select **Show Graphical View**.

#### To see the graphical view of a process model:

1. On the Welcome page, click  (Deployed Process Models).  
The Deployed Process Models window opens displaying a list of all the deployed business process models. See [Performing operations on deployed process models](#)
2. Click  (Show Graphical View) on the toolbar or right-click the process instance and select **Show Graphical View**.  
Alternatively, right-click the process instance and select **Show Graphical View**.

#### Note:

- To view the previous process instance and navigate back to the current process instance of a published business process, click the required navigation icons on the activity associated with any one or more of the BPMN group constructs. For example, For Each, While, Until, Embedded Sub-process and Transaction.
- If a process instance has an aborted Transaction, to view the error text in its message map, right-click the Transaction construct and select **Show Error Text**.

#### Interpretation of graphical view of process instance

When a process instance is viewed in graphical view, the activities of the process are represented with various image and graphical notation based on their execution status.

- Single thick border around an activity denotes that the activity has been executed.
- Double walled green colored border around an activity denotes that the process execution is in that activity.
- Completed Symbol () in an activity denotes that the activity is in the **Completed** state.
- Pause Symbol () in an activity denotes that the activity is in the **Waiting** state.
- Suspended symbol () in an activity denotes that the activity is in the **Suspended** state.
- Red icon (with symbol similar to Close) () denotes that the activity has been terminated.
- Red colored border around an activity denotes that the execution of the activity has failed.
- Red colored broken line border around an activity and a Red icon (with a single diagonal line) in the activity denotes that the activity has been aborted.
- Blue colored broken line border around an activity and a Skip icon in the activity denotes that the activity has been skipped.

To view the Message Map, right-click the process graphical view, and then select  (Show Message Map).

The Message Map Data - View and Edit the Message map Data dialog box opens. You can do the following:

- To view all the messages, click the **All View** tab.
- To copy all the messages to the clipboard, click the **All View** tab, and then click  on the toolbar.
- To select and view a specific message, click the **List View** tab.
- To copy the currently displayed message to the clipboard, click the **List View** tab, and then click  on the toolbar.
- To navigate between the messages, click the navigation buttons on the toolbar.  
All the input and output messages of the activities in the process are displayed according to the selected option.

## Viewing activities of a process instance

### Before you begin:

You must have the Process Administrator role in the AppWorks Platform Business Process Engine application package to view activities of a process instance so that you may perform required operations on those process instances.

### To view activities of a process instance from Process Instance Manager:

1. On the Welcome page, click  (Process Instance Manager).  
The Process Instance Manager window opens displaying a list of business processes that have process instances. See Process Instance Manager Interface in the *AppWorks Platform Advanced Development Guide*.
2. Click  (Find) on the toolbar.  
The Filter Process Instances window opens. If you do not specify the filter criteria for process instances, the default filter options are considered based on which the process instances are displayed.
3. Type the selection criteria. See Filter Process Instances Interface *AppWorks Platform Advanced Development Guide*.
4. Click **OK**.  
All the process instances that match the specified criteria are displayed.
5. To view the process instance details, click the instances with the status.  
The Instances by Process Definition window opens.
6. Select the process instance and click  (Show Activities).  
The Activities pane opens.
7. To view the input message to the activity, click  (Show Input Message) on the toolbar.  
Alternatively, right-click the activity name and select **Show Input Message**.

8. To view the output message from the activity, click  (Show Output Message) on the toolbar. Alternatively, right-click the activity name and select **Show Output Message**.
9. To see the error message if the process aborted because of the activity, click  (Show Error Text). Alternatively, right-click the activity name and select **Show Error Text**.
10. To drill down to the sub-process details if the activity is a sub-process, click  (Show Sub-Process Details). Alternatively, right-click the activity name and select **Show Sub-Process Details**. Select the subprocess instance and click  (Show Activities).
11. To see the read-only view of the decision table, if the activity is of type **Decision table**, click  (Show Decision Table Details). The rule columns colored in green color denotes that those rules are successfully evaluated. The rule columns in red color denotes that those rules did not match the condition criteria defined for the given input message.

You have successfully viewed the activities of a process instance.

## Viewing subprocesses of a process instance

You can drill down to the subprocess details of a process instance from the Process Instance Manager (PIM). If a subprocess contains other subprocesses, the traversed path is displayed on top of the Sub Processes pane and the path can be drilled down.

### Before you begin:

You must have the Process Administrator role in AppWorks Platform Business Process Engine application package to view the subprocesses of a process instance.

### To view subprocesses of a process instance from the Instances by Process Definition interface:

1. On the Welcome page, click  (Process Instance Manager). The Process Instance Manager window opens with a list of business processes that have process instances.
2. Click  (Find) on the toolbar. The Filter Process Instances window opens. See [Filter process instances interface](#) in the *AppWorks Platform Advanced Development Guide*. If you do not specify the filter criteria for process instances, the default filter options are considered.
3. Type the selection criteria.
4. Click **OK**. All process instances that match the specified criteria are displayed.
5. To view the sub-process instance details, click the instances with the status . The Instances by Process Definition window opens. See [Instances by Process Definition interface](#).

6. Select the process instance and click  (Show Sub-Processes).  
The Sub-Processes window opens with the subprocess details.

**To view subprocesses of a process instance from All Instances View:**

1. On the Welcome page, click  (Process Instance Manager).  
The Process Instance Manager window opens with a list of business processes that have process instances.
2. Click  (Options) and select All Instances View.  
The Process Filters window opens displaying a list of all the saved process filters.
3. Click **OK**.  
All process instances that match the specified criteria are displayed.
4. To display the subprocesses, select the process instance and click  (Show Sub-Processes).  
The Sub-Processes window opens.
5. To drill down to the subprocesses of the subprocess instance, click  (Show Sub-Processes).

The subprocess details appear in the Sub Processes pane.

## Performing operations in the Process Instance Manager

To control the process instances during runtime, you need to perform certain operations in the Process Instance Manager.

**To perform operations in the Process Instance Manager:**

1. On the Welcome page, click  (Process Instance Manager). The Process Instance Manager appears, displaying a list of business processes that contain process instances.
2. Do one of the following:
  - Suspend: To pause or hold a waiting or running process instance, right-click the link that indicates the number of instances in:
    - Waiting state and select **Suspend Waiting Process Instances**.
    - Running state and select **Suspend Running Process Instances**.
  - Terminate: To stop a process instance that is not complete, right-click the link that indicates number of instances in:
    - Waiting state and select **Terminate Aborted Process Instances**.
    - Suspended state and select **Terminate Suspended Process Instances**.

- Debug ready state and select **Terminate Debug Ready Process Instances**.
  - Debug state and select **Terminate Debug Process Instances**.
- Resume: To restart a suspended process instance, right-click the link that indicates number of instances in:
- Suspended state and select **Resume Suspended Process Instances**.
  - Debug ready state and select **Resume Debug Ready Process Instances**.
3. Do one of the following:
- Suspend - To pause or hold a waiting or running process instance, right click the link that indicates the number of instances in
    - waiting state and select **Suspend Waiting Process Instances**.
    - running state and select **Suspend Running Process Instances**.
  - Terminate - To stop a process instance that is not completed, right click the link that indicates number of instances in
    - waiting state and select **Terminate Aborted Process Instances**.
    - suspended state and select **Terminate Suspended Process Instances**.
    - debug ready state and select **Terminate Debug Ready Process Instances**.
    - debug state and select **Terminate Debug Process Instances**.
  - Resume - To restart a suspended process instance, right click the link that indicates number of instances in
    - suspended state and select **Resume Suspended Process Instances**.
    - debug ready state and select **Resume Debug Ready Process Instances**.
4. Do one of the following:
- Suspend - To pause or hold a waiting or running process instance, right click the link that indicates the number of instances in
    - waiting state and select **Suspend Waiting Process Instances**.
    - running state and select **Suspend Running Process Instances**.
  - Terminate - To stop a process instance that is not completed, right click the link that indicates number of instances in
    - waiting state and select **Terminate Aborted Process Instances**.
    - suspended state and select **Terminate Suspended Process Instances**.
    - debug ready state and select **Terminate Debug Ready Process Instances**.
    - debug state and select **Terminate Debug Process Instances**.

- Resume - To restart a suspended process instance, right click the link that indicates number of instances in
  - suspended state and select **Resume Suspended Process Instances**.
  - debug ready state and select **Resume Debug Ready Process Instances**.
- Restart: To restart an aborted process instance, right-click the link that indicates number of instances in aborted state and select **Restart Aborted Process Instances**.
- Restart All: To restart all the aborted process instances for a particular model or over a selection of models.

**Note:** The **Restart** option works only when you select any specific aborted process instances. Use the **Restart All** option to restart all the aborted process instances that are related to the selected instance. The selection of such instances can be based on multiple process models.

5. You can do one of the following in the progress indication window:
  - Click **Start** to start process instances that are suspended, resumed, restarted, or terminated.
  - Click **Stop** to stop process instances that are running, resumed, restarted, or terminated.
  - Click **Restart** to restart process instances that are stopped or finished, resumed, or terminated.
  - Click **Finish** to finish the resumed, restarted, or terminated process instances.
6. You can do one of the following in the progress indication window.
  - Click **Start** to start process instances that are suspended, resumed, restarted, or terminated.
  - Click **Stop** to stop process instances that are running, resumed, restarted, or terminated.
  - Click **Restart** to restart process instances that are stopped or finished, resumed, or terminated.
  - Click **Finish** to finish the resumed, restarted, or terminated process instances.
7. You can do one of the following in the progress indication window.
  - Click **Start** to start process instances that are suspended, resumed, restarted, or terminated.
  - Click **Stop** to stop process instances that are running, resumed, restarted, or terminated.

- Click **Restart** to restart process instances that are stopped or finished, resumed, or terminated.
- Click **Finish** to finish the resumed, restarted, or terminated process instances.

## Viewing the message and error text of a process instance

### Before you begin:

- You must have the Process Administrator role in AppWorks Platform Business Process Engine application package to view the start message of a process instance and also to view the error text of an aborted process instance.

### To view the start message of a process instance:

1. On the Welcome page, click  (Process Instance Manager).  
The Process Instance Manager opens with a list of business processes that have process instances. See the Process instance manager interface in the *AppWorks Platform Advanced Development Guide*.
2. Click  (Find) on the toolbar.  
The Filter Process Instances window opens. If you do not specify the filter criteria for process instances, the default filter options are considered based on which the process instances are displayed.
3. Type the selection criteria. See the Filter process instances interface in the *AppWorks Platform Advanced Development Guide*.
4. Click **OK**.  
All the process instances that match the specified criteria are displayed.
5. To view the process instance details, click the instances with the status.  
The [Instances by Process Definition interface](#) window opens.  
Alternatively, click  (Show Sub-Processes) to display the sub-processes.  
The Sub-Processes pane opens.
6. Select the process or sub-process instance and click  (Show Start Message) on the toolbar. Alternatively, right-click the process instance and select **Show Start Message**.  
The Instance Message dialog box opens.
7. Click the **XML view** tab to view the XML structure of the start message.
8. Click the **Tree view** to view the start message in a tree view.
9. To view the error text of an aborted process instance, select the process instance, and click  (Show Error Text) on the toolbar.  
Alternatively, right-click the process instance and select **Show Error Text**. The Error text dialog box opens, displaying the activity details at which the process aborted.

The following table applies for an End event, where end type is **Error**:

Property in the End Event pane	Mapped to .... in Error Text in PIM
<b>Process Error Message</b>	<b>Fault message</b>
<b>Error Detail</b> <b>Note:</b> If Error Detail is not specified, Error Definition is mapped to Description in the error text in the PIM.	<b>Description</b>

## Editing the message map data of a process instance

### Before you begin:

You must have the Process Administrator role in AppWorks Platform Business Process Engine application package to view the message map of a process instance.

You can edit the message map of a process instance if it is in one of the following statuses:

- **Suspend** - Process instance that is suspended. Activities in a process instance cannot be executed till the process instance is resumed.
- **Aborted** - Process instance that is aborted. A process instance can be aborted due to some error, for example, failure to invoke a Web service.
- **Waiting** - Process instance that is waiting for a response from a manual activity or an incoming message.
- **Debug Ready** - Process instance that has reached a break point which was defined for the business process model during design time. It can be continued through the process debugger or it can be resumed.

### To view and edit the message map of a process instance

1. On the Welcome page, click  (Process Instance Manager).  
The Process Instance Manager window opens displaying a list of business processes that have process instances.
2. Click  (Find) on the toolbar.  
The Filter Process Instances window opens. If you do not specify the filter criteria for process instances, the default filter options are considered.
3. Type the selection criteria. See Filter Process Instances Interface in the *AppWorks Platform Advanced Development Guide*.
4. Click **OK**.  
All the process instances that match the specified criteria are displayed.
5. Click the instances with the status to view the process instance details.  
The Instances by Process Definition window opens. Alternatively, click  (Show Sub-

Processes) to display the subprocesses.

The Sub-Processes pane is displayed.

6. Select the required process instance and click  (Show Message Map) on the toolbar. Alternatively, you can right-click the process instance and select **Show Message Map**. The Message Map Data - View and Edit the Messagemap Data dialog box opens.
7. Click  (Edit) on the toolbar to edit the messages in the Message Map Data - View and Edit the Messagemap Data.

**Important:** You can view  (Edit) on the toolbar and edit the message map data only when the process instance is in one of the above defined statuses. Also, (Save),  (Validate),  (Refresh) and  (Undo) are displayed only when you click  (Edit).

8. Select the view to edit the messages. **List View** is a guided mode whereas **All View** is in unguided mode. Guided mode allows you to view the current messages in which changes are made. In **List View**, the message content is shown according to the message selected in left pane whereas, in **All View**, the complete message map is shown
  9. In **List View**, select the message to edit and make required changes to the message map in the Current message pane. Changes to Instance properties are not saved.
  10. Click  on the toolbar to refresh the message map.
  11. Click  on the toolbar to undo the changes made in the content of the **Current message**.
  12. In **List View**, select and right-click a message in the Messages pane, and do any one of the following:
    - **Insert Message:** Select this option to insert a message in the **Current message** pane content. Alternatively, to insert a new message select in the empty area in the **Messages** pane of the **List View** to add it to the message map.
    - **Rename Message:** Select this option to rename an existing message.
    - **Delete Message:** Select this option to delete an existing message.
  13. Click  on the toolbar to validate the content of the **Current message**.
  14. Click  to save the changes to the message map.
- Important:** The changes to the message map are saved only if there are no validation errors in the message map.
15. Click  (Copy Content of Current Edit Area) on the toolbar to copy the content of the **Current message** to the clipboard.

You have successfully edited the message map data of a process instance.

## Skipping an activity

When a business process runs, if there is an error in performing a service, the Process Engine ends the process with an error condition. See [Creating a custom error in the AppWorks Platform Advanced Development Guide](#). A process administrator can restart a process. See [Performing operations in the Process Instance Manager](#).

However, you cannot restart the process, if:

- the service is an insert activity and the insertion failed due to unique key violation in the backend,
- it is not possible to reach the backend system,
- there is an unforeseen problem in the execution of the activity, or
- to skip every artifact including subprocesses both from the **All Process Instances** and **Graphical** views.

In such cases, you can skip the abandoned activity and continue the process. If the output from the skipped activity is required for subsequent activities in the business process, the output message of the activity must be provided. The output message provided is updated in the **Message Map**, and the activity that caused the ending of the process is skipped and the business process continues.

### Before you begin:

You must have the Process Administrator role in AppWorks Platform Business Process Engine application package to skip an activity.

### To skip an activity:

1. On the Welcome page, click  (Process Instance Manager).  
The Process Instance Manager opens displaying a list of business processes that have process instances.
2. Click  (Find) on the toolbar.  
The Filter Process Instances window opens.
3. Type the selection criteria. See [Filter process instances interface in the AppWorks Platform Advanced Development Guide](#).
4. Click **OK**.  
All the process instances that match the specified criteria are displayed.
5. Click the instances to view the status.  
The Instances by Process Definition window opens.
6. Select the abandoned process instance to skip and click  (Skip Activity).  
Alternatively, right-click the process instance and select **Skip Activity**.  
If the activity to skip has an output message defined in the WSDL, then the Message dialog box opens from where the output message is captured.
7. Type the parameter values and click **OK**.  
The business process is continued.

You have successfully skipped the activity of the abandoned process instance.

## Adding a new process filter

You can add new process filters to filter process instances in the Process Instance Manager (PIM) when the existing filters are inadequate or if a new filter is to be defined.

### Before you begin:

- You must have the Process Administrator role in AppWorks Platform Business Process Engine application package to add a new selection filter.

### To add a new process filter:

1. On the Welcome page, click  (Process Instance Manager).  
The Process Instance Manager window opens.
2. Click  (Options) and select **Manage Process Filters**.  
The Process Filters window opens displaying a list view of all the saved process filters.
3. On the toolbar, click .  
The Create Process Filter window opens.
4. In the Select by Business Process Name groupbox, click .  
The Select Business Process dialog box opens.
5. In the Select Business Process dialog box, select a business process.
6. Specify the criteria to filter the process instances in the **Select by Activity in Process**, **Select by Instance Properties** and **Select by Process Identifier**.
7. Click **Save as** to save it as a new selection.
8. Click **Reset** to revert to the predefined selection criteria.
9. Select **Set as user default** to use this selection as the default selection criteria.
10. Click **OK**.

The selected criteria is added to the Process Filters list.

## Configuring the access to task parts from Process Instance Manager

An administrator can set the access levels of **Process Instance Manager** task parts for any user. All of these task parts are available to an administrator on the grid viewer and the graphical viewer of Process Instance Manager. Other users can see only those task parts that they have access to.

An administrator can give or revoke access for any of these task parts to a user. The following table lists these task parts. For instructions to set the access levels, see [Assigning tasks to users](#).

<b>Task parts</b>	<b>Descriptions</b>	<b>Icons</b>
Delete	Deletes a process	
Restart	Restarts an aborted process	
Suspend or Resume	Toggles between suspending a running process, or resuming a suspended one or one that is ready to be debugged	and
Terminate	Ends a running or a suspended one	
Debug	Opens the debugger	
Skip Activity	Skips the specified activity	
Instance Details	Makes available the graphical viewer and the process logs for the instance	and
Show Business Identifier values	Shows the business identifier values	
Message Data	Shows the input messages, output messages, queues for incoming messages, and error logs	, , , and
Edit Message Map Data	Shows the message map data for editing	
Use Filters	Shows the options to filter the results of a Process Instance search on	
Manage Filters	Enables the managing of filters that can be used to search for business process instances in Process Instance Manager	-
Excel Export	Exports all of the listed process instances to a Microsoft Excel file	
Forward and Delegate tasks	Forwards or assigns specific activities of a business process instance to the specified user	and

Task parts	Descriptions	Icons
Reset	Sets or removes a reset point from a business process instance	

## Monitoring rules in the Process Instance Manager

Rule monitoring helps in monitoring the running of the Decision Table by showing the rules that were run with color coding to denote if the condition for each rule was met.

**Note:** Monitoring of rules is possible only when the Decision Table is used as an Activity in BPM. Rule monitoring is not feasible for Web services on the Decision Table.

### Before you begin:

- Ensure that the Monitoring options is enabled while setting the properties of the relevant Activity of the BPM , so that the Activity Status, Input messages, and Output messages are monitored. By default, this option is disabled. See Monitoring tab section in Activity properties interface in the *AppWorks Platform Advanced Development Guide*.
- Ensure the Rule Management service container is in the **Started** state.

### To monitor rules in the Process Instance Manager:

1. On the **Welcome** page, click  (Process Instance Manager). The Process Instance Manager window opens with a list of business processes that have process instances. See Process Instance Manager interface in the *AppWorks Platform Advanced Development Guide*.
2. Open the Activities View for the required process instance in the Process Instance Manager.
3. Right-click the **Decision Table Activity** and select **Show Decision Table Details**. The Rule Monitoring dialog box opens, displaying the Decision Table with all the rules that were run.

All the rules whose conditions are not met are displayed in red and the rules that have passed are displayed in green.

## Resetting a process instance from graphical mode

In some cases, you may need to manually suspend a business process instance to modify the process or the values provided in the process. To again start running the suspended process, you can use the reset feature. This feature also enables you to reset the point of execution, that is you can select any activity from which to start running. Resetting a business process creates another instance with a new ID.

It is possible to reset a process and set it back in the execution path.

When the reset point is set at an activity that precedes two parallel execution paths, both the paths are reset.

#### **Before you begin:**

- You must have the process administrator role in the AppWorks Platform Business Process Engine application package to view and reset a process instance in the graphical mode.
- Ensure to suspend the process instance before you reset an activity.

#### **To reset a business process instance, if you are working in the graphical view:**

1. Click **<Organization> > Process Administrator > Process Monitoring > All Process Instances**.  
The All Process Instances window opens.
2. Select the process instance and click  (Show Graphical View) on the toolbar.  
Alternatively, right-click the process instance and select **Show Graphical View**. The business process instance appears in the graphical view in the read-only mode.
3. Click  (Reset) and click the activity starting from which you want to run the flow again.  
Alternatively, right-click the activity and select **Set Reset Point**. The reset icon appears on the selected activity.
4. Right-click the process instance and select **Resume**.  
Alternatively, select the **Resume** option from the **Process Options** menu in the toolbar.  
The Resume Process Instances dialog box opens.
5. To modify the data, click **Edit Message Map**.
6. Click **Start** to run the process instance flow.

The business process instance is reset and executed to completion.

## **Restarting a process instance from graphical mode**

In some cases, an activity in a business process instance may be aborted due to issues with the associated Web service operations. In such cases, you will need to resume the process after the relevant corrections have been made. You can do this by restarting the business process instance, which creates another instance with a new ID. It is possible to restart only the business process instances that are aborted or suspended.

#### **Before you begin:**

You must have the process administrator role in the AppWorks Platform Business Process Engine application package to view and restart a process instance in the graphical mode.

#### **To restart a business process instance in the graphical view:**

1. Click **<Organization> > Process Administrator > Process Monitoring > All Process Instances**.  
The All Process Instances window opens.

2. Select the process instance, and click  (Show Graphical View) on the toolbar.  
Alternatively, right-click the process instance and select **Show Graphical View**.  
The business process instance opens in the graphical view in the read-only mode.
3. Select **Restart** from the **Process Options** menu on the toolbar.  
Alternatively, right-click the business process model and select **Restart from here**.  
The Restart Process Instances dialog box opens. For information about the options available in this dialog box, see Restart Process Instances Interface in the *AppWorks Platform Advanced Development Guide*.

**Note:** If the restart point for a process instance is changed, the process instance goes to the suspended state. You can navigate forward in the process by using the **Skip activity** feature.

## Chapter 17

# Performing operations on deployed process models

The Deployed Process Models window helps you to view and perform certain operations on business process models that are already deployed from your current organization or from different organizations. This is the only window through which you can have a summarized view of the business process models that are otherwise not seen in **Workspace Documents** as they are loaded through the application packages and belong to different organizations.

Fields on the Deployed Process Models Interface:

Field	Description
Folder	Name of the folder in which the business process model is located.
Process Name	Name of the published business process model.
Last Published	Date and time at which the business process model was last published.
Published By	Name of the user who published the business process model.
Deployed In	Name of the organization where the business process model is deployed.
Configuration	Either Monitoring or Crash Recovery settings are customized for the model. If customized, the value for this field will be <b>custom</b> .
Description	Description of the published business process model.

### To perform the required operations on a deployed business process model:

1. On the Welcome page, click  (Deployed Process Models).  
The Deployed Process Models window opens.

To perform the operation	Right-click a process model or select a process model and
 (Show Graphical View)	Select <b>Show Graphical View</b> to view the execution sequence of the activities in a deployed process model.

To perform the operation	Right-click a process model or select a process model and
	<p>Alternatively, click  on the toolbar.</p> <p>The selected or deployed process model appears in a graphical view.</p>
 (View Instances by Process)	<p>Select <b>View Instances by Process</b> to monitor the process instances based on their business process.</p> <p>Alternatively, click  on the toolbar.</p> <p>The selected or deployed process model appears in the Instances by Process Definition window from where you may perform several other operations. See <a href="#">Instances by Process Definition interface</a>.</p>
 (Customize Configuration)	<p>Select <b>Configuration &gt; Customize</b>. Alternatively, click  on the toolbar to modify or overwrite the monitor options that were set at the time of creating the business process model.</p> <p>The Monitoring and Crash recovery settings window opens from where you may modify monitoring and crash recovery settings. See <a href="#">Monitoring and crash recovery interface</a>.</p> <p><b>Note:</b></p> <ul style="list-style-type: none"> <li>■ When a model exists both in the Shared space and the Organization space, customizing the model in Organization space reflects in the Shared space also. However, for the users of the Organization, the model that exists in the Organization space will only be considered for any new process instances. <ul style="list-style-type: none"> <li>• When a model exists both in Shared space and Organization space, you cannot update the customization made on the model present in Shared space directly.</li> </ul> </li> <li>■ When a model exists only in the Shared space, the customization made to that model will not impact users of any other organization.</li> <li>■ If any customization is done on a process model in an Organization, it will be considered automatically even if the process model is updated; users need not update the configuration every time the process model is updated.</li> <li>■ If a model exists either in the Shared space or in the Organization space, then deletion of the model also</li> </ul>

To perform the operation	Right-click a process model or select a process model and
	<p>results in deletion of the customization.</p> <ul style="list-style-type: none"> <li>■ When a process model is uninstalled from the Organization space and if the same model exists in the Shared space also, then the customization made to the model in the Organization space is not deleted; it is displayed against the model that exists in the Shared space. Users of the Organization can continue to rely on the custom configuration that is defined.</li> <li>■ It is possible to update the configuration for multiple models at the same time. <ul style="list-style-type: none"> <li>• When multiple models are selected, and if none of them have a predefined customization, then the default configuration of the model displayed first among the selected models will be considered.</li> <li>• When multiple models are selected, and if one or more customized models exist in that selection, the default configuration of the model which is already customized and appears first among the selected models will be considered.</li> </ul> </li> </ul>
 (View Audit Information)	Select <b>View Audit Information</b> to view audit information for a deployed process model. Alternatively, click  on the toolbar. The Artifacts Audit Information window opens, displaying audit information for the selected process model. See Parameters of artifacts audit information in the <i>AppWorks Platform API Reference Guide</i> .
 (Authorization)	Select <b>Authorization</b> . Alternatively, click  on the toolbar to set the authorization for a deployed process model. The Authorization window opens. Set authorization for the selected or deployed process model. See <a href="#">Setting authorization for deployed process models</a> .

You have performed the required operations on a deployed business process model.

## Setting authorization for deployed process models

Business process models can be deployed directly from an application package in the runtime environment. Therefore, authorization for these process models cannot be set in the design time environment. A process administrator can monitor and authorize business processes in runtime by setting authorization for deployed (published) business processes.

The only relevant runtime authorization settings are instantiation authorization per role or per use.

#### **Before you begin:**

- You must have the Process Administrator role in the AppWorks Platform Business Process Engine application package to set authorization for deployed (published) business process models.
- You must have published a business process model.

#### **To set authorization for deployed process models:**

1. On the Welcome page, click  (Deployed Process Models).  
The Deployed Process Models window opens displaying a list of deployed business process models.
2. Right-click a business process model and select  (Authorization). The Authorization dialog box opens.
3. Click  on the toolbar.  
An empty record is displayed for setting authorization.
4. In the Role/User column, click  to browse, and select a role or the name of a user.  
The Select a Role or an User dialog box opens.

**Note:** If you have set authorization at the user level, you cannot set authorization at the role level.

5. In **Select Role or User**, select either the **Role** or **User** options appropriately.  
Depending upon the option that you selected, either a list of roles or a list of users are displayed.
6. Select the role or the user to authorize, and click **OK**.  
The selected role or user appears in the Authorization dialog box.
7. Select the required **Role/User** for authorization, and select the **Instantiate** option.
8. Click .

The user or the user having the selected role is authorized to instantiate the business process model.

Authorization is set for a deployed process model.

# Chapter 18

## Archiving process instances

Archiving helps in storing the published process instances that are no longer active or cannot be restarted from the database at a different location. The storage mechanism depends upon the archive policy defined by the administrator. At a later stage, the administrator can load the archived data back into Admin Cockpit to retrieve the business process details. Using the Process Archival Manager feature, the administrator defines an archive policy to archive the data in the Admin table of the live database. See [Defining an archive policy](#). The archive policy definition must ensure that the following are present:

- The criteria for selecting the business processes that need to be archived
- The criteria for triggering the archive policy

When you execute an archive policy, the process instances matching the archive policy's filter criteria are archived to the file system along with corresponding activities and business identifiers data. The archived file also contains a copy of the corresponding deployed process model data.

Upon restoring an archive, the process instances and their related data are imported back in to the repository. However, if the corresponding deployed process model data does not already exist in the repository, it is loaded along with the process instances.

When you unload an imported archive, the entire imported data is cleared from the relevant repositories.

**Note:** In case of older archives, for example, archives created on AppWorks Platform installations prior to the Cordys BOP 4.1 release, the process identifiers associated with the process instances are mapped back to the business identifier repository. When you unload an imported archive file, both the process instances and business identifiers are cleared from the Process Instance Repository and Business Identifier Repository respectively.

### Types of archive policies

There are two types of archive policies: manual and scheduled.

When an archive policy definition is set to Manual, the administrator needs to trigger the archiving process manually. When an archive policy definition is set to Scheduled, the system triggers the archiving process automatically as per the set schedule, provided the archive policy is activated. It is also possible to manually trigger a scheduled archiving policy whenever required, without altering or disturbing the schedule. For more information on archiving, see the following topics:

- Defining an archive policy
- Activating or deactivating an archive policy
- Triggering an archive policy
- Loading an archive
- Viewing loaded archive details

## Defining an archive policy

Use the Process Archival Manager feature to define an archive policy by setting the:

- Criteria to select the business processes to be archived.
- Triggering type to archive those business processes.

### Before you begin:

You must have the Process Administrator role in the AppWorks Platform Business Process Engine application package to define an archive policy.

### To define an archive policy:

1. On the Welcome page, double-click  (Process Archival Manager).  
The Archive Policies table displays the details of archive policies that have been defined and saved, while the Archive Log View table displays the detailed log for the triggered policies.
2. Click  in the Archive Policies table.  
The new archive policy dialog box opens.
3. Type the required information to define the criteria for the archive policy. See New policy interface in the *AppWorks Platform Advanced Development Guide* for more information on the fields in this dialog box.
4. Click **Save** to save the archive policy.  
The details of the archive policy you created appear in the Archive Policies table.

## Activating or deactivating an archive policy

### Before you begin:

- You must have the role of a process administrator in AppWorks Platform Business Process Engine application package to activate or de-activate an archive policy.
- An archive policy can be activated only when the **Policy Type** is Scheduled.

### To activate or deactivate an archive policy:

1. On the Welcome page, double-click  (Process Archival Manager).  
The Archive Policies and Archive Log View grids appear.

2. In the Archive Policies table/grid, right-click a policy of type **Scheduled** and select **Activate**.

The archive policy is activated.

**Note:**

- If you activate a scheduled archive policy that has been saved without setting the schedule, a prompt appears to set the schedule when you activate the policy.
  - You can activate only one archive policy per organization. One simple reason is that if you activate multiple policies, the policies may overlap and you may not be sure of what data goes into which archive.
3. To deactivate a policy, right-click an already activated policy in the **Archive Policies** table and select **Deactivate**.

The archive policy is deactivated.

## Triggering an archive policy

After you define an archive policy, you can trigger it immediately or at a later period. For a scheduled archive policy, the archiving occurs as per the scheduled archiving settings. However, for manual archive policies you have to manually trigger the policy.

**Before you begin:**

You must have the process administrator role in AppWorks Platform Business Process Engine application package to trigger an archive policy.

**To trigger an archive policy:**

1. On the Welcome page, double-click  (Process Archival Manager). The Archive Policies and Archive Log View grids are displayed.
2. Right-click an archive policy in the **Archive Policies** table and select **Run Now**.  
The archive policy is performed. The archive file that is generated is displayed in the **Archive Log View** table.
3. Download the archive policy by clicking the archive file displayed in the **Archive Log View** table.

## Loading an archive

The process of importing data from an archive is called loading.

**To load an archive:**

1. On the Welcome page, click  (Restore Process Instances).  
The Restore Process Instances - Restore archived process instances page opens displaying a list of loaded archive files available on the server.

2. To upload a new archive file from the local system to the server, click  (Import Archive).  
The Select Archive window opens displaying a list of available archives.
3. Select an archive from the list.
  - If the organization space of the selected archive is different from that of the current organization, select the **Change organization of the archived data to current organization** option.  
Ensure that the database configuration details of Business Process Service Container is different from the database configuration details used to create the archive.  
Alternatively, click the browse icon in **Upload**, select an archive from your computer, click **Upload**.

**Note:** If the size of the archive file is large, then upload it to the server manually to  
 <web root directory>\cpc\admin\archive.

4. Click **OK**.

You have successfully loaded the selected archive.

## Viewing loaded archive details

### Before you begin:

You must have the Process Administrator role in AppWorks Platform Business Process Engine application package to view the loaded archive details.

### To view loaded archive details:

1. On the Welcome page, double-click (Restore Process Instances).  
The Restore Process Instances - Restore archived process instances page opens displaying the log and import details of the loaded archives.
2. Select an archive from the list of loaded archives.  
The **Archive Details** and the **Import Details** tabs for the selected archive are displayed on the right pane.
3. Click the **Archive Details** tab to view the log details for the selected archive.  
The following information is displayed.

Field	Description
Instance Count	The number of records archived.
Database source	The database server from which the archive was created.
Machine Name	The name of the host.
Version	The version of the product form which the archive was generated.
Triggered User	The user who triggered the archive.

Field	Description
Triggered At	The time at which the archive was triggered.
End Time	The time at which the execution of the archive policy ended.

4. Click the **Import Details** tab to view the import details for the selected archive. The following information is displayed.

Field	Description
Loaded By	The details of the user who loaded the archive.
Unload	Click to unload the archive file.
Last Modified	The date on which the archive was last modified.
Status	The state of the loaded archive.
Resume	Click to resume the import process.

# Chapter 19

## Monitoring case instances

After running a case Model, you may want to monitor its case instances to view the state of each activity and the status of the case instances. For example, an activity of a case instance may exist in different states such as initial state or final state whereas, a case instance may go through different statuses such as new, in progress, ended, suspended, resumed, terminated, or completed.

Case Instance Monitoring (CIM) is done to ensure that all the activities within the case model are acted upon by the case workers and completed as planned. Case instance monitoring is also done while debugging to test the model and resolve errors in the model, if any.

You can view any of the following from the CIM:

- A single process from the Case Modeling environment (design time)
- A single process from the Case Model deployment environment (runtime)
- Summary of multiple case instances

### Before you begin:

- Execute a case model. See Executing a case model in the *AppWorks Platform Advanced Development Guide*.

### To view the case instances of a single case model from the case modeling environment:

1. Select a starting point and click  (Case Model) to open an existing Case Model. See Creating a document in the *AppWorks Platform Advanced Development Guide*.
2. Right-click on the Case Modeler and select **Case Model Execution > Show Case Instances**.

The Instances by Case Definition window displays the case instances of the selected Case Model. See [Instances by case definition interface](#).

### To view the case instances of a single process from the case model deployment environment:

1. On the **Welcome** page, click  (Case Instance Manager).

The Case Instance Manager displays a view of all executed case models, indicating the total number and number of new case instances for each model.

2. To view the case instances, double-click the <Case Model>. A graphical view of the latest case instance for the selected case model is displayed in the **Overview** tab by default.
3. To view all Case instance details, click on the **Detail View** tab for the selected Case Model. All Case instances for the selected Case Model are displayed. See [Manage case instances interface](#) for information about the fields and operations that can be performed from the **Detail View** tab.
4. Right-click a Case instance and provide any one of the following monitoring operations, if required:
  - **View Activity Details:** Select to view Case activity details. Refer to Case Activities Interface for more information on the type of information that you can view.
  - **View Case Data:** Select to view the Case data.
  - **Suspend Case:** Select to suspend the Case instance. For example, you can suspend all Case instances which are not in **Suspended, Aborted , Closed** or **Terminated** status.
  - **Terminate Case:** Select to terminate the Case instance. For example, you can terminate all Case instances, which are not in **Aborted , Closed**, or **Terminated** status.
  - **Close Case:** Select to close the Case instance. For example, you can close all Case instances which are not in **Suspended, Aborted , Closed**, or **Terminated** status.
  - **View Case Error:** Select to view error for the Case instance when a Case instance is in **Aborted** status.
  - **Resume Case:** Select to resume Case instance monitoring when a Case instance is in **Suspended** status. Alternatively, select required Case instances check box and click on the toolbar icons as needed to perform the above monitoring operations.

Case instances of a Case Model are monitored.

## Instances by case definition interface

Fields in the Instances by case definition interface in the Case Instance Manager page:

Field	Description
Status	Current business status of the case instance.
Current Functional State	The business status of a case can be part of the current functional state, which can also be a milestone, for example, a case can be closed (completed), but the business status can be approved or rejected. <ul style="list-style-type: none"><li>▪ For a terminated case, the state at which the case has got terminated is displayed.</li></ul>

Field	Description
	<ul style="list-style-type: none"> <li>■ For a suspended case, the state at which the case has got suspended is displayed.</li> <li>■ For a closed case, the state at which the case has come to an end is displayed.</li> </ul> <p><b>Note:</b> Default state is displayed for the activity that is not associated with any state.</p>
Started On	Start day, date and time at which the case instance was started. For example, Tuesday, November 11, 2008 3:48:33 PM
Started By	Name of the case worker who started the case instance. For example, Luca Raggi.
Acted On	Day, date, and time at which the case instance was acted upon. For example, Tuesday, November 11, 2008 3:48:33 PM
Acted By	Name of the case worker who acted upon the case instance. For example, Luca Raggi.
Due On	Day, date, and time at which the case instance is due to be closed. For example, Monday, November 14, 2008 3:48:33 PM.

### Operations in the Instances by case definition interface

Operations that can be performed in the Instances by case definition interface:

<b>To perform operation...</b>	<b>Right-click a process instance or select a process instance and...</b>
Suspend	<p>Select <b>Suspend</b> to suspend the case instance.            Alternatively, click  on the toolbar.</p> <p><b>Note:</b> You can suspend a case instance that have the status <b>New</b> or <b>InProgress</b>.</p>
Resume	<p>Select <b>Resume</b> to resume the execution of the process instance.            Alternatively, click  on the toolbar.</p> <p><b>Note:</b> You can resume only the process instances that have the status <b>Suspended</b>.</p>
Restart	<p>Select <b>Restart</b> to restart a process instance.            Alternatively, click  on the toolbar.</p> <p><b>Note:</b> You can restart only the process instances that have the status <b>Aborted</b>.</p>
Terminate	<p>Select <b>Terminate</b> to terminate a process instance.            Alternatively, click  on the toolbar.</p> <p><b>Note:</b> You can terminate a process instance if it is not</p>

<b>To perform operation...</b>	<b>Right-click a process instance or select a process instance and...</b>
	<b>Complete.</b>
Close Case	Select <b>Close Case</b> to close the case instance. <b>Note:</b> You can close all the case instances, which are not in the <b>Suspended, Aborted, New, or Terminated</b> status.
View Activity Details	Select <b>View Activity Details</b> to view Case activity details. See Case Activities Interface in the <i>AppWorks Platform Advanced Development Guide</i> for more information on the type of information that you can view.
View Case Data	Select <b>View Case Data</b> to view the case data.
View Case Error	Select <b>View Case Error</b> to view error for the case instance when a case instance is in the <b>Aborted</b> status.

# Chapter 20

## Business Activity Monitoring administration

BAM administration provides options to monitor the data processed for BAM and to manage KPIs of the type trend.

The tabs in BAM administration:

- **Settings** - Assists in configuring the settings to synchronize and upload the data, and also other advanced settings used by the BAM engine.
- **Trend manager** - Assists in enabling or disabling the trend settings related to KPIs of the type trend.

The settings defined through the BAM administration page are applicable only for the organization through which they are set.

### Settings tab

The controls on the **Settings** tab of the BAM administration page are enabled only when you deploy the AppWorks Platform BAM MDM model package.

These settings are categorized as follows:

#### Synchronize

Synchronization refers to the near real-time transfer of the business process instances data of all or a set of selected business processes to BAM. The data present at every event such as PROCESS\_START, PROCESS\_END, ACTIVITY\_START, and ACTIVITY\_END during the business process instance execution is read from the instances tables and propagated to BAM tables in the database.

By default, BAM synchronization is not enabled when you create a new organization. As the organization does not contain a BAM service container, synchronization is not supported for that organization.

Synchronization can be enabled or disabled by selecting or clearing the **Enable Synchronization** option.

- **Enabling BAM synchronization** - With the **Enable Synchronization** option, you can select to synchronize either all or a set of business processes for a specific organization. Data synchronization is enabled in the organization when the BAM service container is started in the system organization.  
**Note:** If an organization other than the system organization has a BAM service container already created in it, start the service container to enable the synchronization.
- **Disabling BAM synchronization** - During synchronization, data is transferred from business process instances tables to BAM tables.

## Upload

Uploading refers to transferring the historical business process instances data of the selected business processes to BAM tables. The data is transferred from business process instances tables to BAM tables in the database.

Upload activity is enabled in the system organization and other organizations when you deploy the AppWorks Platform BAM MDM model package. The organizations other than the system organization do not need the BAM service container to be created in the organization, provided the service container in the system organization is running. The administrator must trigger the upload activity to upload the historical instances data of the selected business processes to BAM.

Selecting at least one business process is mandatory to trigger the upload action. The selected business processes are considered only for the upload iteration and not for synchronization. You can also provide an optional time frame input to upload the historical data. The time frame selection is based on the business process instances creation date and includes both from and to dates.

The list below contains the possible time frame options:

- Past events - This option enables recording of the data for all the completed BAM events (PROCESS\_START, PROCESS\_END, ACTIVITY\_START, ACTIVITY\_END for completed instances) based on the status of the uploaded business process instances.
- Latest events - This option enables recording of the data only for the latest BAM events (PROCESS\_END and ACTIVITY\_END for completed instances) based on the status of the uploaded business process instances.

See [Uploading a business process](#) for more information on how to upload the data for the selected business processes in the organization.

Administrators can view the logs for further details on upload execution. The log viewer displays the list of the tables that are considered while uploading the data. For more information, see [Viewing MDM logs](#).

**Tip:** It is recommended to upload the historical data from the BAM Administration page instead of the MDM model browser.

## Configure advanced settings

Synchronization and uploading of a process transfers the data from the Business Process Instances related tables to the BAM tables. The standard dashboard collects the reporting data from the BAM tables. In addition to the BAM standard dashboard, the BAM engine fills the data for the Process Monitoring Object and Business Event Response documents.

Through **Advanced Settings**, you can monitor and control the data for these documents.

The configuration for the advanced settings:

- **Process monitoring objects** - With this option, you can enable or disable the data filling for the Process Monitoring Objects. This data is considered as the input data for BAM monitoring.
- **Business event response** - This option enables or disables the data filing and evaluates the conditions to trigger appropriate actions for the Business Event Responses, which is visualized as SLA monitoring in BAM.

See [Configuring advanced monitoring options](#) for a detailed explanation on how to configure the settings.

**Note:** If you do not select the **Process monitoring objects** or **Business event response** options, respective filling of the monitoring data and SLA monitoring will not occur in the organization.

On selecting at least one of the above options, the settings that are configured for uploading (Past events and Latest events) will be considered by the BAM engine for recording the monitoring data and enabling SLA monitoring. When these advanced settings are not selected, the BAM engine will not consider the upload-specific settings for the upload. The Process Monitoring Object and Business Event Response documents must be created before uploading the instances data.

**Note:**

- Data monitoring and SLA monitoring will not occur if the Process Monitoring Object and Business Event Response documents are created after triggering the upload. In such cases, the upload must be triggered again.
- The advanced settings can be enabled only when the BAM service container is created in the organization.

**Tip:** It is recommended to create the BAM service container in the organization. After creating it, enable the options available in the Settings tab to record the BAM monitoring data.

## Trend Manager tab

**Note:** Ensure that the **Scheduling** service container is started, to view the KPI trend information on the **Trend Manager** tab.

The **Trend Manager** tab provides an overview of the KPIs of the type trend that are available for the organization. The Administrator can enable or disable the trend settings for the available KPI definitions.

Each entry in the table represents a unique KPI definition that is of the type trend. Using  (Enable Trend) and  (Disable Trend), you can enable or disable the trend calculations relating to the selected KPI definition. When the associated scheduler is disabled, it will not capture the related data, and therefore the related trend plot is depicted without any data for that period.

The administrator can also use **Column Chooser** option available to view more details about the deployed KPI definitions that are of the type trend. The following columns are available in the display table:

- KPI ID - Unique identifier for the KPI
- KPI Name - Name of the KPI provided during the design time
- Folder - Path where the KPI was created in design time
- Schedule ID - Unique identifier for the associated schedule template
- Schedule Event ID - Unique Identifier for the schedule instance
- Space - Displays the schedule space. Either **Shared** or **Organization** is displayed based on the space where the deployed definition is available.
- Status - Current status of the associated schedule. Either **Active** or **Inactive** is displayed based on the current status of the associated schedule definition.

The **Trend Manager** tab also provides search functionality. The administrator can search, enable, or disable trends based on the KPI ID, KPI Name, Space, and Status.

## Configuring advanced monitoring options

While deploying the AppWorks Platform Business Activity Monitoring package during installation, create a BAM service container in the system organization.

To configure the advanced monitoring options in non-system organization, the BAM service container must be created and started in the organization. On starting the service container for the first time, both the advanced monitoring options are enabled.

### Before you begin:

- The AppWorks Platform BAM MDM Model package must be deployed.
- Ensure that you create and start the BAM service container in the organization.
- You must have an Administrator role to view the  (BAM Administration) task on the Welcome page.

### To change the advanced monitoring options:

1. Select or clear the **Process Monitoring Objects** check box to enable or disable the recording of the monitoring data.

2. Select or clear the **Business Events Response** check box to enable or disable the evaluation of conditions to trigger the configuration activities.
3. Click .  
The advanced monitoring configuration is saved for the organization.

## Disabling BAM synchronization

### Before you begin:

- AppWorks Platform BAM MDM Model package must be deployed.
- You must have an Administrator role to view  (BAM Administration) task on the Welcome page.

### To disable synchronization in any organization:

1. On the Welcome page, click  (BAM Administration).  
The BAM Administration page is displayed.
2. Clear **Enable Synchronization**.  
The synchronization options are disabled.
3. Click  on the toolbar.

Synchronization is disabled for that organization.

### Note:

- To disable synchronization in a non-system organization, BAM service container is not required, provided the BAM service container in the system organization is started.
- To delete BAM service container, in a non-system organization, when advanced monitoring is not required, perform the above mentioned steps before deleting the service container.

## Enabling BAM synchronization

While deploying the AppWorks Platform Business Activity Monitoring package during AppWorks Platform installation, the BAM service container is created in the system organization. On starting the service container, synchronization is enabled in the system organization.

### Before you begin:

- AppWorks Platform BAM MDM model package must be deployed.
- You must have an Administrator role to view  (BAM Administration) task on the Welcome page.

**To enable synchronization in a non-system organization:**

You can enable synchronization in a non-system organization by using one of the following ways:

**By creating a service container:**

1. Create the BAM service container in the required organization. See Creating a Service Group for BAM in the *AppWorks Platform Advanced Development Guide* for more information.
2. Start the service container to enable the synchronization for all the business processes.

**Without creating a service container:**

1. On the Welcome page, click  (BAM Administration).  
The BAM Administration page is displayed.
2. Select **Enable Synchronization**.
3. Click  to enable synchronization.

You can select all or a set of business processes for synchronization as follows:

- To enable synchronization for all the processes in that organization, select **Synchronize all processes**.
- To enable synchronization only for the selected business processes, select **Synchronize only selected processes**.
  - Open the BAM administration page.  
The list displays the selected business processes that were saved earlier (if any).
  - To add more business processes, click .  
The list of available business processes in the organization is displayed.
  - Select the required business processes from the list.
  - Click **OK**.  
The selected processes are added to the list.
  - To remove any process from synchronization, select the processes and click .
  - Click  on the toolbar.

The list is updated with the new set of business processes that are configured for synchronization.

**Note:** To enable synchronization in an organization other than system organization, BAM service container is not required, provided the BAM service container in the system organization is started.

# Uploading a business process

## Before you begin:

- AppWorks Platform BAM MDM Model package must be deployed.
- You must have an Administrator role to view  (BAM Administration) task on the Welcome page.

## To upload business process instances data to the BAM tables:

1. On the Welcome page, click  (BAM Administration).  
The BAM Administration page opens.
2. Select one of the options from **Past events** and **Latest events** for uploading the data.
3. Click **Start Uploading**.  
The list displays the selected business processes that were saved earlier (if any).
4. Add or remove the required business process.  
The historical data of the selected business processes is uploaded.

**Note:** Selecting atleast one business process is mandatory to trigger the upload activity.

If required, provide input for the time frame. Time frame is based on the process instances creation time as listed below:

- To upload all the instances related to the selected business processes, do not select any time frame.
- To upload the instances related to the selected business process created from a specific date (inclusive) till current date, select only from-date.
- To upload the instances related to the selected business process created from the beginning till a specific date (inclusive), select only to-date.
- To upload the instances related to the selected business process created between a date range (both inclusive), select both from and to dates.

5. Click **Upload**.

Uploading is triggered with the selected business processes.

**Note:** The selected list of business processes is applicable to the current upload iteration only. This list is not considered for synchronization.

6. To view the upload related logs, click **View log**. For more details, see [Viewing MDM logs](#).

## Note:

- You are recommended to upload the historical process instances data from BAM Administration, than from the MDM Model Browser.

- To trigger an upload in non-system organization, BAM service container is not needed, provided the BAM service container in the system organization is started.

# Chapter 21

## Managing and monitoring MDM

After designing an MDM model, it must be published. Only a published model can be used for run time activities.

After successful configuration and runtime deployment of the MDM model, you would expect the software to execute the desired behavior without manual intervention. However, in most cases, the presence of knowledge workers to handle unexpected data situations is unavoidable. In addition, unexpected events could impact the smooth execution. These could then result in error situations that would depend upon human intervention to rectify and set things moving again.

Some data administration activities need to be performed periodically to ensure that desired performance levels are achieved. Data might have to be archived; log files may have to be purged under a scheduled policy regime.

Also, changing conditions have to be managed. Data loads might increase or decrease depending upon the business requirements. More users might need to be given access to the managed data objects. These require administrative efforts to ensure that as the data management requirements scale, the underlying execution platform also scales in sync.

Monitoring and management is thus an important governance activity aimed at ensuring the desired business or IT objectives are being continuously delivered by the Data management program executed using AppWorks Platform MDM.

The chief activities involved in this phase of the program are:

- Tracking the execution progress and looking for any anomalies and out-of-bound situations.
- Handling exceptions, including debugging as part of error analysis.
- Analysis such as Audit Trail or History tracking of data objects.
- Periodic verification and modification of settings that govern administrative tasks.
- Other administrative activities such as adding new users, assigning or modifying user permissions for data access.

For more information,

- on MDM admin tasks, see [MDM admin settings](#).
- on MDM Monitoring and Management tasks, see [MDM data steward cockpit](#).

## MDM admin settings

This topic describes the settings that can be modified in the MDM Admin Settings window.

### Before you begin:

You must have the role of an administrator to perform administrative tasks.

MDM administrator is responsible for creating service containers such as MDM Publisher and MDM Service.

The primary role of the administrator is to ensure the smooth deployment of MDM. The MDM administrator can also define the activities that must be logged, the duration for which logs must be stored, and the different conditions for which notifications must be sent. Similarly, the administrator can also define the time period for storing state history related information.

Logs are generated while the data is being synchronized. Logs can be used for monitoring purposes. The MDM Admin Settings window allows you to manage the settings for both regular data synchronizations and manual synchronizations done in batches. You can also change the settings to send notifications to the AppWorks Platform Inbox or any regular email inbox. In Admin Settings, you can select the number of days after which the logs can be purged. Similarly, the number of days after which the logs for state history can be purged can also be defined.

The following are the tasks that can be performed through the MDM Admin Settings window:

- [Assigning MDM proxy user](#)
- [Configuring MDM repository](#)
- [Configuring conditions for receiving notifications](#)
- [Configuring time period for storing logs](#)
- [Downloading database scripts](#)

Additionally, you can set the following properties:

- **Allow Modified Fields Only:** Whenever you update a record in a database table, MDM sends the updated record with all the columns instead of sending only the columns that were updated. If you are updating different columns of a particular record from two or more sources, the previous content is overwritten by the latest content, leading to data loss and data inconsistency as the entire record is updated. Therefore, to ensure that only the modified columns are propagated during sniffing, before publishing the model, select the **Allow Modified Fields Only** option. This option is not selected by default.
- **Retain Target Values for Empty Source Values:** Select this option to retain the values in the target elements if the corresponding source elements contain empty values. This option is not selected by default.

## Assigning MDM proxy user

MDM proxy user is a proxy (person authorized to act for another) for someone else. During data synchronization, the service containers perform most of the core work in the background and do not involve any explicit user requests. Any service container in the SOA grid works on a user context. Therefore, the proxy user is used for all the daemon work that these service containers execute. In case of exceptions, this user is again used to publish the notifications.

1. On the Welcome page, click  (MDM Admin Settings).  
The MDM Admin Settings window opens.
2. In **Configuration > Select MDM Proxy User**, select a user.

**Note:** The selected user must have the Administrator role assigned to him.

3. Click  to save.

## Configuring MDM repository

After publishing, the MDM models are stored in the MDM Repository. The publishers, that is the spoke and hub, and the MDM service retrieve runtime information of the model from this repository.

An MDM repository can be configured in the following ways:

- **Central MDM Repository:** In the central repository, the spoke and the hub share a single repository and point to the same database. The MDM models in different organizations of AppWorks Platform can use a single MDM repository. The central MDM repository enables sharing of the MDM repository among multiple hubs in various organizations.
- **Distributed MDM Repository:** For a distributed MDM repository, there are separate MDM repositories for the spoke and hub publishers. If the databases are available at different locations, the publishers are configured to the local databases instead of pointing to the central MDM repository.

### To store MDM content in the repository:

1. On the **Welcome page > My Applications**, click  (MDM Admin Settings).  
The MDM Admin Settings window opens.
  2. By default, the **Use Central MDM Repository** check box is selected. Ensure that the **Use Central MDM Repository** check box is selected only to store the MDM runtime content in the central MDM repository.
- Note:** If the **Use Central MDM Repository** check box is not selected, a distributed MDM repository is created for the model.
3. Click  to save.  
The information related to MDM is stored in the MDM repository.

## Configuring conditions for receiving notifications

This topic describes the procedure to configure Cordys MDM to enable users with appropriate roles to receive notifications whenever synchronization or bulk operation occurs.

### To configure notifications for the synchronization activities:

1. On the **Welcome page > My Applications**, click  (MDM Admin Settings).  
The MDM Admin Settings window opens.
2. Go to **Synchronization** in the Notifications group box.
  - To receive notification in AppWorks Platform Inbox, select the **Through Inbox** option.
  - To receive notification through email, select the **Through Mail to Recipients** option. Enter the email ID in the adjacent field. To provide multiple email IDs, separate them with commas (,).
3. Click  to save.

The conditions for receiving notifications are configured.

### To configure notifications for bulk activities:

1. On the **Welcome page > My Applications**, click  (MDM Admin Settings).  
The MDM Admin Settings window opens.
2. To configure notifications for bulk activities such as upload and download, go to **Bulk Operations** in the Notifications group box.
  - To receive notification in AppWorks Platform Inbox, select the **Through Inbox** option.
  - To receive notification through email, select the **Through Mail to Recipients** option. Enter the email ID in the adjacent field. To provide multiple email IDs, separate them with commas (,).
3. In **Sender Mail ID**, enter an email ID.  
Any email notification is sent through this email address.
4. Click  to save.

The conditions for receiving notifications are configured.

### Note:

- The Sender Mail ID box is enabled only if you select the **Through Mail to Recipients** option either under **Synchronization**, or **Bulk Operations**, or both.
- Notifications are delivered to users who have administrator credentials.
- To restore the settings to default, click **Restore Default**.

## Configuring time period for storing logs

Over a period of time, as the MDM logs start accumulating, they fill up the database. Extensive accumulation of logs may eventually impact synchronization, and if left unattended, may even stop data synchronization. As an administrator, you are responsible for periodical backups of the logs and freeing the database. MDM provides an option to automatically delete logs older than a user-defined time period. As an administrator, you can set this time period to store the logs.

### To configure time period for storing logs:

1. On the **Welcome page > My Applications**, click  (MDM Admin Settings).  
The MDM Admin Settings window opens.
2. By default, MDM logs only unsuccessful transactions during synchronization. To log successful transactions, select the **Log Successful Transactions during Synchronization** check box under **Log Settings**.
3. Select the **Purge Logs Older than Days** check box in the Log Settings group box to delete the logs, and provide the number of days during which the logs must be retained. If you do not want to delete logs, clear the check box.

**Note:** By default, logs are deleted after seven days.

4. Select the **Purge State History Logs Older than Days** check box in the Log Settings group box to delete logs of the purge state history, and provide the number of days during which the state history logs must be retained. If you do not want to delete logs of the purge state history, clear the check box.

**Note:** By default, state history is deleted after seven days.

5. Click  to save.

The time period to maintain the logs by MDM is defined.

## Downloading database scripts

When the model containing data stores is published, certain database scripts are run on the actual database of the data store. While designing the MDM model, if the **Download Database Scripts** option is enabled in the Data Store property, then these scripts are not executed on the database automatically, and are saved in a file. A Database Administrator (DBA) can download the scripts, review them, and execute them manually on the database. This is useful to control each and every alteration made to the database.

### To download database scripts:

1. On the Welcome page, click  (MDM Admin Settings).  
The MDM Admin Settings window opens.
2. Select **Download Database Script**.  
A page with **Download Database Script** is displayed, and a table with the **Model Key**

and **Script Type** columns is displayed. The models are listed in the **Model Key** column. Based on the MDM model state, the **Script Type** can be **Publish** or **Unpublish**.

**Note:**

- When the models are published, then scripts of the type **Publish** are available for download to be executed on the database.
- When the models are unpublished, then scripts of the type **Unpublish** are available for **Download** to be executed on the database.

The model database scripts are downloaded. The downloaded scripts are available as a .zip file. You can save the files in a specified location or view them.

**Note:**

- To delete the model scripts if they are no longer required, click  (Delete) on the toolbar.
- To retrieve the latest changes to the scripts, click  (Refresh) on the toolbar.

## Managing MDM models

An MDM model contains spokes and hub that represent the various data stores within the organization. An MDM model depicts the relationship of the spokes with the hub. The published or run-time MDM models are used for synchronizing data across different data stores. These models are obtained by publishing the design-time models.

Following management tasks can be performed on a published model.

- **Unpublishing an MDM Model:** A published model can be unpublished if you do not want to continue working with the model. If you have to change the way data is synchronized then you must edit the design-time model and re-publish it. If a published model is unpublished, edited and then published, it is possible that in the meantime data to be synchronized may be lost. Therefore, ensure that a model is unpublished or re-published depending upon the situation. For information on un-publishing an MDM model, see [Unpublishing an MDM model](#).
- **Perform Runtime Operations:** The run-time activities like Upload data from the requisite spoke data entities to the appropriate hub data entities, can be done. Similarly, Download of data from hub to spoke entities can be performed. For information on Runtime Operations, see [Performing runtime operations using an MDM model](#).

## Viewing the runtime MDM model in a dashboard

The Dashboard <MDM Model> window lists the number of objects in different states that belong to an entity. This window displays the following information that is continuously updated:

- The number of data entities in the selected hub data store
- The total number of messages associated with a selected hub entity
- The errors occurred in the state engine for the selected entity
- The errors occurred in Match Object for the selected entity that is governed by an object life cycle model
- The number of errors that occurred during synchronization
- The count of instances in their respective states under the Business Object Instances pane
- The count of instances in the hub entity that have matches coming in, in their respective states under the Match Object Instances pane
- The number of errors occurred during the recent upload of records under the Recent Upload Status pane

**To view the runtime MDM model in a dashboard:**

1. On the Welcome page, click  (MDM Data Steward Cockpit).  
The MDM Data Steward Cockpit window opens with a list of published MDM models.
2. You can view the summary of the published MDM model in any one of the following ways:
  - Click the published MDM model, and click  (View Dashboard) from the tool bar.
  - Right-click the published MDM model and select  (View Dashboard).  
The Dashboard - <MDM Model> window opens with different states of the entity of the selected MDM model.
3. To view summary, select the hub entity in **Select Hub Entity**.  
The Runtime Errors pane opens with the details of the number of errors occurred in the selected hub entity:
  - during synchronization
  - in the State Engine
  - match object errors

The Business Object Instances pane opens with a graph depicting the count of instances in the respective states, and the Match Object Instances pane opens with a graph depicting the count of matched instances in their respective states.

**Note:** The number of errors that occur in the State Engine and the number of match object errors that occur in the selected hub entity are displayed in the Runtime Errors pane, only if the **Enable Business Object Lifecycle** check box is selected while creating the hub entity and if there are any errors in the State Engine and in the match object.

4. Use the Dashboard - <MDM Model>:
  - To view the total number of entities in the hub, see the number beside the **Total Entities In The Hub**.
  - To view the messages in the AppWorks Platform Inbox, click the number beside **Total Messages**.
  - To view synchronization errors, click the number beside **Sync Errors**, under the Runtime Errors pane.  
The Log Viewer appears with the log details. See [Viewing MDM logs](#).
  - View the errors in the State Engine. See [Viewing state errors](#).
  - View the Match Object Errors that the Candidate Match state model generates while performing match operation. See [Viewing match object errors](#).
  - View the business object instances in a particular state. See [Viewing business object instances](#).
  - View the matched object instances in a particular state. See [Viewing the matched object instances](#).
  - To view the current status of upload status, under Recent Upload Status view the status, total and failed records of the selected hub entity.  
Click the status, the Log Viewer opens, with the log details. See [Viewing MDM logs](#).

5. To refresh the Dashboard - <MDM Model> window, click  (refresh).

### ***Viewing state errors***

The State Errors window displays the errors in the hub entity that was selected in the cockpit. Errors can occur in business instance objects or candidate (matched) instance objects.

#### **To view state errors:**

1. On the Welcome page, click  (MDM Data Steward Cockpit).  
The MDM Data Steward Cockpit window opens with a list of published MDM models.
2. You can view state errors in any one of the following ways:
  - Click the published MDM model to view the state errors, and click  (View Dashboard) from the tool bar.
  - Right-click the published MDM model and select published MDM model to view the state errors and select  (View Dashboard).  
The Dashboard - <MDM Model> window opens with the graphical runtime view of the selected MDM model.
3. To view summary, select the hub entity in **Hub Entity**.  
The summary of the selected hub entity is displayed. If there are any state errors for the selected entity, a graph depicting the number of errors occurred in the State Engine is displayed in the Runtime Errors pane.

4. Click the **State Errors** bar graph.  
The State Errors window opens with a list of rows displaying the errors in the selected hub entity, the primary keys of the records, the time at which the error occurred, the exception details, and so on.
5. To rectify the exceptions, select a row .  
The exception details is displayed in the Exception Message section.
6. To view more details about the exception, click **More Details**, and then rectify the exception.
7. To view the complete instance of the record, select the rectified exception instance (row) and click (Click to view the instance of the selected record).  
The Business Object Instance window opens. You can modify the values of an instance and then manually trigger an event.
8. To view complete details of the instance, select the row, and then click (Click to view the selected record).  
The State Instance window opens with the complete details of the record.
9. Click to refresh the State Exceptions window.

## **Viewing match object errors**

### **To view match object errors:**

1. On the Welcome page, click (MDM Data Steward Cockpit).  
The MDM Data Steward Cockpit window opens with a list of published MDM models.
2. You can view match object errors in any one of the following ways:
  - To view the match object errors, click the published MDM model, and then click (View Dashboard) from the tool bar.
  - Right-click the published MDM model for which you want to view the match object errors and select View Dashboard.  
The Dashboard - <MDM Model> window opens with the graphical runtime view of the selected MDM model.
3. To view summary of the selected hub entity, select the hub entity from the list next to Hub Entity.  
A graph depicting the match object errors for the selected entity opens in the Runtime Errors pane.
4. Click the bar representing the Match Object Errors.  
The Match Exceptions window opens with a list of rows depicting the errors, the primary keys of the records, the time at which the error occurred, and so on.
5. To rectify the exceptions, select the row.  
The exception details is displayed under the Exception Details section.
6. To view more details about the exception, click **More Details**.  
Rectify the exception.

7. To view the complete details of the instance, select the row for which you have rectified the exception and click (Click to view the instance of the selected record). The Match Instance details window opens. For more information on the fields, see Matched Instance Details Interface in the *AppWorks Platform Advanced Development Guide*.
8. To view the complete details of the selected record, select the row for which you have rectified the exception and click (Click to view the selected record). The State Instance window opens with complete details of the record.

## **Viewing business object instances**

The Business Object Instance window displays the instances of a hub entity in a particular state.

1. On the Welcome page, click (MDM Data Steward Cockpit). The MDM Data Steward Cockpit window opens with a list of published MDM models.
2. You can view the business object instances in any one of the following ways:
  - Click the published MDM model for which you want to view the business object instances, and click (View Dashboard) from the tool bar.
  - Right-click the published MDM model to view the business object instances and select (View Dashboard). The Dashboard - <MDM Model> window opens with different states of the entity of the selected MDM model.
3. To view summary, select the hub entity from the list. The Business Object Instances pane opens with a graph depicting the count of instances for error and non-error records in the respective states. You may note that if there are any errors for the selected entity then **State Errors:** <number of errors>, and **Match Object Errors:** <number of errors> is also displayed.
4. To view the instances in the state, click the bar representing the state, under the graph of the Business Object Instances section. The **Business object instance view** window opens with all the details of the clicked instances belonging to the state.

### **Business object instance view**

The Instance View window displays the list of business object instances of a selected state. For any state, instances can be categorized as error and non-error records. An Instance View has options to filter a set of records based on the states and error category. You can select a set of records and delete those instances from runtime.

<b>Field</b>	<b>Description</b>	<b>Actions</b>
Filters	Select the required state and	In the <b>Select a State</b> list, to

Field	Description	Actions
	<p>category and click <b>Get Instance</b>. The business object instances are displayed in the Instance View pane on the right based on the filter selected in this pane.</p>	<p>view select the required state of the business object instance.</p> <p>Based on your requirement, in <b>Category</b>, select one of the following categories:</p> <ul style="list-style-type: none"> <li>■ <b>All Records</b> - All records are displayed.</li> <li>■ <b>Error Records</b> - Only error records are displayed.</li> <li>■ <b>Non-error Records</b> - Records with no errors are displayed.</li> </ul>
Instance View	<p>Displays the business object instance view. To perform actions on the instances use the icons on the toolbar.</p>	<p>To open an instance, select the instance to view the business object instance and click  (Open Selected Instance).</p> <p>The Business Object Instance window opens. You may edit the business object instances and manually trigger an event. See Business object instance interface in the <i>AppWorks Platform Advanced Development Guide</i> for more information.</p> <p>To show state history, select the instance to view the history of the business object instance and click  (Show State History).</p> <p>The State History window opens. See Business object instance state history interface in the <i>AppWorks Platform Advanced Development Guide</i> for more information.</p>

Field	Description	Actions
		<p>To show candidate pairs, select the instance to view the matched objects of the instance and click  (Show Candidate Pairs).</p> <p>The Match Object Instances window opens. See <a href="#">Viewing the matched object instances</a> for more information.</p> <p><b>Note:</b> You can view the matched objects of the instance only if you have enabled the Data Quality option in the State Engine of the model while creating a data entity. See <a href="#">MDM data entity interface</a>.</p>
		<p>To delete instance, select the instances to delete and click  &gt; <b>Selected Records</b>.</p> <p>To delete all instances, click  &gt; <b>All Records</b>.</p> <p>In the confirmation dialog box, click <b>Yes</b> to delete the selected or all the records.</p>
		<p>To resend error records, select the error records to resend and click (Resend Error Records) &gt; <b>Selected Records</b>.</p> <p>In the confirmation dialog box, click <b>Yes</b> to resend all the error records.</p> <p>Click (Resend Error Records) &gt; <b>All Records</b>.</p> <p>In the confirmation dialog box, click <b>Yes</b> to move the records to the next valid state.</p> <p><b>Note:</b> This option is enabled</p>

Field	Description	Actions
		<p>only for the error records.</p> <p>To refresh, click  .</p> <p>The Instance View window refreshes and displays the latest instances.</p>

## Viewing the matched object instances

The Matched Object Instances window displays the matched instances in a particular state. You can view the matched objects of the instance only if you have enabled Data Quality in the State Engine for the model, while creating a data entity. See [MDM data entity interface](#).

### To view the matched object instances:

1. On the Welcome page, click  (MDM Data Steward Cockpit).  
The MDM Data Steward Cockpit window opens with a list of published MDM models.
2. You can view the matched object instances in any one of the following ways:
  - a. To view the matched object instances, click the published MDM model and then click  (View Dashboard) from the tool bar.
  - b. To view the matched object instances, right-click the published MDM model and select  View Dashboard.  
The Dashboard - <MDM Model> window opens with different states of the entity of the selected MDM model.
3. To view summary, select the hub entity from the list.  
The Match Object Instances pane opens with a graph displaying the count of matched instances in their respective states.

**Note:** If there are any errors for the selected entity then **State Errors:** <number of errors>, and **Match Object Errors:** <number of errors> also appear along with the Business Object Instances pane. See [Viewing the matched object instances](#).

4. Click the bar that represents the state under the graph of the Match Object Instances, to view the list of records that have matched with the incoming business instance.  
The Match Object Instances window opens with a list of records that have matched with the incoming business instance. See Matched object instance interface in the *AppWorks Platform Advanced Development Guide* for more details.
5. To view the matched instance details, select the instance (row), and then click  (View selected match object).  
The Match Instance details window opens. You may edit the business object instances and manually trigger an event. See Matched instance details interface in the *AppWorks Platform Advanced Development Guide* for more details.

6. To view the history of the business object instance, select the instance (row), and then click  (View selected match object history).  
The State History window opens. See Matched object instance state history interface in the *AppWorks Platform Advanced Development Guide* for more details.
7. To refresh the Matched Object Instances window, click .

## **Viewing audit trail details**

An audit trail is a trail of data changes for a particular record. Each and every change to a record in the master data repository is tracked. Such information can be used to audit master data.

**Note:** You can view the audit trail for a record only if the **Enable Business Object Lifecycle** option is enabled in the MDM Model while providing details for the hub data entity properties. See Creating a Data Entity in the *AppWorks Platform Advanced Development Guide*.

### **To view audit trail details:**

1. On the Welcome page, click  (MDM Data Steward Cockpit).  
The MDM Data Steward Cockpit window opens with a list of published MDM models.
2. You can view the details of audit trails in any one of the following ways:
  - Click the published MDM model for which you want to view details of the audit trail, and click  (View Dashboard) from the tool bar.
  - Right-click the published MDM model and select published MDM model for which you want to view details of the audit trail and select  View Dashboard.  
The Dashboard - <MDM Model> window opens with the graphical runtime view of the selected MDM model.
3. Click  (Audit Trail and Registry Viewer).  
The Audit Trail and Registry Viewer window opens.
4. In the **Show filters for** pane, select the **Audit trail** option.
5. In the **Display Instances of** pane, select the following:

<b>Field name</b>	<b>Description</b>
Business Object	To view the summary, select the business object.
Modified From	To view the changes made on the object instances entered in the particular state, select the start date.
Modified To	To view the changes made on the object instances in the particular state, select the end

Field name	Description
	date.
States	To view the summary, select the state of the object instance.

6. Click  to add key values.

The key field that you have specified as primary key while creating a data entity is displayed under **Key** along with **Value**. See [MDM data entity interface](#).

7. In **Key**, type the value to view the instance details.

You can use wild characters as the value of the key field.

**Note:** Click **Reset** to clear the above settings.

8. To retrieve the audit trail details of the selected business object instance, click **Get instances**.

The Instance View window opens on the right pane with the details of the clicked instances and belonging to the state.

9. The audit trail details of the object instances are displayed.

**Note:** Click  to save the search criteria, and reuse the search options later by selecting it from the **Select filter** list.

## Viewing registry details

A registry is a collection of entries. In the context of MDM a registry holds a collection of links which are pointing to the corresponding records in different systems for this master data record.

1. On the Welcome page, click  (MDM Data Steward Cockpit).

The MDM Data Steward Cockpit window opens with a list of published MDM models.

2. You can view the details of registry in any one of the following ways:

- To view details of the registry, click the published MDM model, and click  (View Dashboard) from the tool bar.

- To view details of the registry, right-click the published MDM model, select a published MDM model, and then select  View Dashboard.

The Dashboard - <MDM Model> window opens with the graphical runtime view of the selected MDM model.

3. Click  (Audit Trail and Registry Viewer).

The Audit Trail and Registry Viewer window opens.

4. In the **Show filters for** pane, select the **Registry** option.

5. In the **Display Instances of** pane, select the following:

Field	Description
Query By	To query the registry by the <b>PrimaryKey</b> or the <b>RegistryId</b> .
Hub Entity	The hub entity to query.
Spoke Data Stores	The data store to query. <b>Note:</b> <b>Spoke Data Stores</b> is not displayed if you select <b>RegistryId</b> in <b>Query By</b> .
Spoke Entities	To further query the registry to the spoke entity level. <b>Note:</b> <b>Spoke Entities</b> is not displayed if you select <b>RegistryId</b> in <b>Query By</b> .
Primary Key Value	The primary key value. You can use wild characters instead of typing the whole primary key value. <b>Note:</b> <b>Primary Key Value</b> is not displayed if you select <b>RegistryId</b> in <b>Query By</b> .
Registry ID	The registry identifier. You can use wild characters instead of typing the whole registry identifier value.

**Note:** Click **Reset** to clear the above settings.

#### 6. Click **Show Links**.

The registry details appear on the right pane and the associated data store information is displayed at the bottom.

**Note:** Click  to save the search criteria. To reuse the search options later, select the search criteria in **Select filter**.

## Performing runtime operations using an MDM model

The MDM model in AppWorks Platform assists you to synchronize the data between different data stores.

### Before you begin:

Ensure that the MDM model for which data has to be synchronized is published to the runtime.

### To synchronize data:

- Upload data from the requisite spoke data entities to the appropriate hub data entities. See [Uploading data to the hub](#) for additional information.

**Note:** If a spoke uses comparator based sniffing, uploading data is mandatory even if the spoke does not contain data. If the data is uploaded for the first time, a snapshot of the data is taken which is used for comparison after a specified time interval. Ensure

that upload is done before the schedule time as defined in the sniffer for the changes to be sniffed. If upload is done at a later point after the schedule has begun, the spoke service container needs to be restarted for synchronization.

2. If you wish to transfer the data from hub to the spoke, download the data. See [Downloading data to the spoke](#) for more details.
3. Manually synchronize the data at regular intervals for which the automatic synchronization is disabled. See [Manually synchronizing a data store](#).
4. If required, modify the synchronization settings of each subscription. You can enable or disable sniffing and synchronization for each subscription. This is essential for all subscriptions arising out of those data stores for which the **Save SQL scripts in a file** check box was selected while creating the model. See [Changing synchronization settings of a data entity](#) for details on changing synchronization settings.
5. You can modify the cursor size at run time. See [Modifying cursor settings](#) for details on modifying cursor size settings.

## ***Uploading data to the hub***

Uploading is the process of copying data from the spoke to the hub according to the rules specified in the model.

### **To upload the data to the hub:**

1. On the Welcome page, click (MDM Data Steward Cockpit).  
The MDM Data Steward Cockpit window opens with a list of published MDM models.
2. You can upload data in any of the following ways:
  - Click the published MDM model, and click (Perform Runtime Operations) from the tool bar.
  - Right click the published MDM model and select Perform Runtime Operations.  
The Runtime View - MDM Model window opens with the graphical runtime view of the selected MDM model.
3. Do one of the following:
  - Select the spoke entity or the spoke data store to upload the data to (hub) and click (Upload from Spokes to Hub).  
The Upload window opens with the list of relevant subscriptions to the hub data entities.
  - If you select a hub entity and click (Upload from Spokes to Hub), then all the relevant subscriptions to that hub entity are listed.

- If you select the hub data store and click  (Upload from Spokes to Hub), then all the relevant subscriptions to all the entities in the hub data store are listed.

- Click a subscription.  
A property sheet is displayed at the bottom.
- Set the following properties:

Property	Description
Sync With Spokes	To synchronize the data from hub to all the subscribed spokes after uploading the data, select the <b>Sync With Spokes</b> option.
Enable State Flow	If you want the data to be uploaded to go through State Flow before updating in the hub, when a State Engine is defined on a hub entity, select the <b>Enable State Flow</b> option.
Use Lookup	When this option is selected, upload always registers the conflicting data instead of overwriting it.
Threads	If you have specified a Web service with input parameters on the spoke or hub entity to read data, then only the Threads, Parameter, XPath, and Value columns are displayed. Threads facilitate to upload data in batches. The names of the thread specified in the Web service appear under the Threads column.
Parameter	The input parameter names to specify the range of values in the batch are displayed under the Parameter column.
XPath	The XPath of the input parameter in the request is displayed. For example, if '/' is displayed, it indicates that it is the immediate child of the request.
Value	Specify the range values. For example, specify the range values as range1: 0-100, range2: 101-200 and so on.
Use Filter	To filter specific records based on a rule, select this option. The rule filter window is displayed where you can set the required rules. This field appears in the property sheet of the Upload and Download functions.

- Upload the data in any one of the following ways:
  - To individually upload data for each subscription, select the subscription and click  (Start Upload).
  - To sequentially upload data:
    - Select the **Upload in sequence (Use the up and down buttons to alter the sequence)** option. The  or  arrow (or both the arrows) is enabled.
    - Arrange the subscriptions to upload data using  and  arrows.

- c. Click  (Start Upload).  
The upload process starts.

A notification appears prompting to view the MDM logs. Click **Yes** to view the upload logs. The MDM Log Viewer page opens. See [Viewing MDM logs](#) for more information.

The data from the spoke to the hub is uploaded.

## **Downloading data to the spoke**

Downloading data means copying data from the hub to the spoke. While downloading, the rules that are specified in the model are followed.

### **To download data to the spoke:**

1. On the Welcome page, click  (MDM Data Steward Cockpit).  
The MDM Data Steward Cockpit window opens with a list of published MDM models.
2. You can download data in any one of the following ways:
  - Click the published MDM model, and click  (Perform Runtime Operations) from the tool bar.
  - Right-click the published MDM model and select  Perform Runtime Operations.  
The Runtime View - MDM Model window opens with the graphical runtime view of the selected MDM model.
3. Select the spoke entity or the spoke data store to download data from (hub), and click (Download from Hub to Spoke(s)).  
The Download window opens with the list of subscriptions to the data store.  
If you select a hub entity and click  (Download from Hub to Spoke(s)), then all the relevant subscriptions to that hub entity are listed.  
If you select the hub data store and click  (Download from Hub to Spoke(s)), then all the relevant subscriptions to all the entities in the hub data store are listed.
4. Click a subscription.  
A property sheet is displayed at the bottom displaying the **Use Filter** option.
5. Select this option to filter specific records based on a rule. It displays rule filter window where you can set the required rules.
6. You may download data in any one of the following ways:
  - To individually download data for each subscription, select the subscription and click  (Start Download).
  - To sequentially download data:
    - a. Select the **Download in sequence (Use the up and down buttons to alter the sequence)** option.  
The  or  arrow (or both the arrows) is enabled.
    - b. Arrange the subscriptions to download data using  or  arrow.

- c. Click  (Start Download).  
The download process starts.

A notification opens prompting to view the MDM logs. Click **Yes** to view the download logs. The MDM Log Viewer page opens. See [Viewing MDM logs](#) for more information.

The data from the hub to the spoke is downloaded.

### **Manually synchronizing a data store**

**To manually synchronize the data stores with the hub for the ones that are not configured to synchronize automatically:**

1. On the Welcome page, click  (MDM Data Steward Cockpit).  
The MDM Data Steward Cockpit window opens with a list of published MDM models.
2. You can access the run-time view of the published MDM model in any one of the following ways:
  - a. Click the published MDM model, and click  (Perform Runtime Operations) from the tool bar.
  - b. Right click the published MDM model and select  (Perform Runtime Operations).  
The Runtime View - <MDM Model> window opens with the graphical runtime view of the selected MDM model.
3. Click the data store for which you have cleared the **Automatically receive data** check box and click  (Manually Synchronize) from the tool bar. See [MDM data store interface](#).  
The Manually Synchronize Interface dialog box opens with a list of buckets that are available to download for the selected data store. See Manually synchronize interface in the *AppWorks Platform Advanced Development Guide*.
4. Select the check boxes against the buckets to download and click **OK**.

The data store is manually synchronized with the hub.

### **Changing synchronization settings of a data entity**

This topic describes the process to change synchronization settings of a data entity in a published Cordys MDM model.

#### **Before you begin:**

- Before you modify the synchronization settings for a data entity, ensure that the publisher for that data entity is working.
- Ensure that you manually reset the synchronization settings for those entities in data stores with \*Save SQL scripts in the file option selected.

#### **To change synchronization settings of a data entity:**

1. On the Welcome page, click  (MDM Data Steward Cockpit).  
The MDM Data Steward Cockpit window opens with a list of published MDM models.

2. Access the runtime view of the published MDM model in any one of the following ways:
  - Click the published MDM model, and click  (Perform Runtime Operations) from the tool bar.
  - Right-click the published MDM model and select  Perform Runtime Operations. The Runtime View - <MDM Model> window opens with the graphical runtime view of the selected MDM model.
3. Click the data entity to change the synchronization settings, and click  (Change Synchronization Settings) from the tool bar. The Synchronization Settings dialog box opens with all the subscriptions related to that data entity and their synchronization settings. See Synchronization settings interface in the *AppWorks Platform Advanced Development Guide*.
4. Select the **Sniff Changes**, **Synchronize** check boxes to select the required synchronization options and click **OK**. The synchronization settings for the data entity are modified.

### **Modifying cursor settings**

Cursor size is the number of records MDM has to fetch while performing bulk operations such as Upload during data synchronization. You can define the cursor size of a data entity while selecting Web services to read and update data at design-time of the data integration model. There may arise situations where there is a need to change the cursor size during run-time. For example, the number of individual records that are fetched are huge and the computer may run out of memory. In such cases, you may wish to modify the cursor size dynamically without republishing the data integration model. You can modify the cursor size at run-time also.

**Note:** The cursor size set in the runtime is not retained if the model is republished. If the model is republished then the cursor size is reverted to the design-time cursor size.

#### **Before you begin:**

Ensure that the publisher for the data entity for which to modify the cursor settings is working.

#### **To modify cursor size for the data entity:**

1. On the **Welcome** page, click  (MDM Data Steward Cockpit). The MDM Data Steward Cockpit window opens with a list of published MDM models.
2. You can access the runtime view of the published MDM model in any one of the following ways:
  - Click the published MDM model, and click  (Perform Runtime Operations) from the tool bar.
  - Right-click the published MDM model and select  Perform Runtime Operations. The Runtime View - <MDM Model> window opens with the graphical runtime view of

the selected MDM model.

3. Click the data entity to modify the cursor settings and click  (Modify Cursor Settings) from the tool bar.
4. The Update Cursor Size for <Data Entity> dialog box opens with cursor size defined for the Web services to read and update data associated with the data entity.
5. Type the cursor size as required.
6. Click .

The cursor size for the data entity is modified.

## Unpublishing an MDM model

In the MDM Data Steward Cockpit you can un-publish a published MDM model.

1. On the Welcome page, click  (MDM Data Steward Cockpit).  
The MDM Data Steward Cockpit window opens with a list of published MDM models.
2. Click the published MDM model, and click  (Unpublish Selected Model) from the tool bar.  
The Confirm Un-publish window opens with the message, 'Do you want to unpublish the selected model <MDM Model name>"?
3. Choose
  - **Yes**, if you want to unpublish the MDM model
  - **No**, to stop publishing the model
  - **Cancel**, to cancel publishing the modelIf you choose Yes, the MDM - Unpublish Model window opens.  
The unpublish process starts and the status is displayed at the same location.

A successful unpublish is indicated by  whereas errors are indicated by .

To know details of the unpublished document, click **Details**.

The details appear in the **Details** tab.

**Note:** If the **Enable Business Object Lifecycle** and **Enable Registry** or **Enable Data Quality** options are enabled for any of the hub data entities, then the corresponding tables, that is the State Instance Table Name, Registry Table Name, and Match Object Table Name tables and their content, are not removed when the MDM Model is unpublished.

4. After making corrections, publish the document again.

**Note:** To republish an MDM Model without making any changes to the model, click **Clean Build Output** on the CWS toolbar and then publish.

## Viewing MDM logs

MDM facilitates monitoring of critical processes that are associated with it. To take corrective and preventive actions, you can view logs. As an MDM user, you can monitor the logs of the following activities:

- Bulk operations
- Synchronization
- Publishing

### Before you begin:

You must have the role of a MDM administrator or MDM monitor for viewing MDM logs.

### To view MDM logs:

1. On the Welcome page, click (MDM Data Steward Cockpit).  
The MDM Data Steward Cockpit window opens with a list of published MDM models.
2. You can view MDM logs in any one of the following ways:
  - Click the published MDM model, and click (View Logs) from the tool bar.
  - Right-click the MDM model and select View Logs.  
You can do the following in the Log Viewer - MDM Model window.

Operation	Description
Synchronization	<p>Select the filter.</p> <ul style="list-style-type: none"> <li>• <b>Subscriptions</b> - Select the subscription. If no subscription is selected, all the subscriptions from the spoke to the hub are selected.</li> <li>• <b>Operation Name</b> - Select the type of operation performed on the database.           <ul style="list-style-type: none"> <li>• <b>All</b> - All logs are displayed irrespective of the type of operation performed.</li> <li>• <b>Specific</b> - Select one or more types of operation: <b>Insert</b>, <b>Update</b>, and <b>Delete</b>.</li> </ul> </li> <li>• <b>Status</b> - Select the status of the operation.           <ul style="list-style-type: none"> <li>• <b>All</b> - All logs are displayed irrespective of the status of the operation.</li> <li>• <b>Specific</b> - Select one or more types of status: <b>Completed</b>, <b>Failed</b>, and <b>Pending</b>.</li> </ul> </li> <li>• <b>Time</b> - Select the time of the operation.</li> </ul>

Operation	Description
	<ul style="list-style-type: none"> <li>• <b>All</b> - All logs are displayed irrespective of the time of the operation.</li> <li>• <b>Display for the last</b> - To view logs from a time period of a specific length that moves continuously, which can be year, month, week or day.</li> <li>• <b>Between</b> - To view logs for the range falling within the specified From and To dates.</li> </ul>
Publishing	<ul style="list-style-type: none"> <li>• <b>Status</b> - Specify the status of the operation.           <ul style="list-style-type: none"> <li>• <b>All</b> - All logs are displayed irrespective of the status of the operation.</li> <li>• <b>Specific</b> - Select types of status: Completed, Stopped, and Started.</li> </ul> </li> <li>• <b>Time</b> - Select the time of the operation.           <ul style="list-style-type: none"> <li>• <b>All</b> - All logs are displayed irrespective of the time of the operation.</li> <li>• <b>Display for the last</b> - To view logs from a time period of a specific length that moves continuously, which can be year, month, week or day.</li> <li>• <b>Between</b> - To view logs for the range falling within the specified From and To dates.</li> </ul> </li> </ul>
Bulk Operations	<p>You can select the following filters:</p> <ul style="list-style-type: none"> <li>• <b>Subscription Details</b> - Select the subscription. If no subscription is selected, all the subscriptions from the spoke to the hub are selected.</li> <li>• <b>Status</b> - Select the status of the operation.           <ul style="list-style-type: none"> <li>• <b>All</b> - All logs are displayed irrespective of the status of the operation.</li> <li>• <b>Specific</b> - Select types of status: Completed, Done with errors, Stopped, and Started.</li> </ul> </li> <li>• <b>Time</b> - Time of the operation.           <ul style="list-style-type: none"> <li>• <b>All</b> - All logs are displayed irrespective of the time of the operation.</li> <li>• <b>Display for the last</b> - To view logs from a time period of a specific length that moves continuously, which can be year,</li> </ul> </li> </ul>

Operation	Description
	<p>month, week or day.</p> <ul style="list-style-type: none"> <li>• <b>Between</b> - To view logs for the range falling within the specified From and To dates.</li> </ul>

3. To view the logs, click **Get Logs**.

In the right pane, logs window opens with Process ID, Status, Start Time, and End Time, with the following options on the logs window.

Icons	Options	Description
	Delete	The selected logs are deleted.
	Purge	All logs are deleted.
	Refresh	The latest changes reflect in the UI.
	Show/Hide Filters	The Log Filters pane is displayed or hidden.

**Note:** To clear the selections made in the previous step, click **Reset**.

If more than one type of log is selected, the logs are displayed in tabs. Double-click a log to view its details in More Details - <Type of log>.

## MDM data steward cockpit

This topic describes the monitoring and managing MDM models using the data steward cockpit.

### Before you begin:

You must have the role of an analyst to perform monitoring and management tasks.

A data steward cockpit delivers real-time, on-demand access to runtime instances of underlying data objects. The cockpit serves as the single point of entry for a data steward to perform human tasks associated with data integration. The role of a data steward is to ensure that the master data is clean, consistent, and accurate.

Depending upon the activities modeled, these might include handling data exceptions, performing data edits, acting upon notifications and so on. The cockpit also delivers insight into the events that caused the changes on a data object instance and the overall trace of the lifecycle of the object till date. Data is managed and monitored in the MDM Data Steward Cockpit.

- For managing data in MDM cockpit, see [Managing MDM models](#).
- For monitoring Data in MDM Cockpit, see [Monitoring MDM models](#).

## Monitoring MDM models

Cordys MDM also facilitates monitoring of critical processes that are associated with it. Monitoring critical processes enables you to take corrective as well as preventive action. As an MDM user with an analyst role, you can perform monitoring activities by viewing the MDM logs.

You can perform the following monitoring tasks on a published model:

- [Viewing the runtime MDM model in a dashboard](#)
- [Viewing MDM logs](#)
- [Viewing runtime details](#)

### ***Viewing runtime details***

You can monitor sniffers configured for a design time model from the **View Runtime Details** page. This page provides the sniffer details such as sniffer status, schedule, and so on.

#### To monitor sniffers:

1. On the Welcome page > **My Applications**, click  (MDM Data Steward Cockpit). The MDM Data Steward Cockpit window opens with a list of published MDM models.
2. To view the runtime details of the published MDM model, click the **View Runtime Details** icon from the toolbar.
3. In **Select Data store**, select the data store to view the sniffer details.  
All the sniffers related to the data store are displayed in the pane below.
4. Select one of the following to display the data entities based on your requirements:
  - **Show All** - All the data entities.
  - **Show Active** - The data entities having active sniffers.
  - **Show Inactive** - The data entities for which the sniffers are inactive that is, either stopped or the schedule window has expired.
  - **Show Scheduled** - The data entities for which the schedule (active and inactive) is enabled.
5. Based on the above selection, the data entity table displays the following data:
  - **Data Entity** - The entities present for the selected data store.
  - **Sniffer Type** - The type of sniffer defined for the entity while designing the model. If the Sniffer type is **Comparator**, the **Last Snapshot Date** field is displayed

additionally. It displays the last executed sniffer time.

- **Sniffer Status** - Status of sniffer: active, inactive, or stopped.
  - **Scheduled** - If the schedule is enabled in the design time: **Yes** or **No**.
6. Click the required record to display the sniffer details on the right side:
    - **Audit Table Name** - Displays the name of the audit table provided during the design time
    - **Is Schedule Active Now** - Displays if the schedule is currently active or not
    - **Next Schedule** - Displays the next schedule time
    - **Service Container DN** - Displays the service container to which the data entity is configured

**Note:** For a hub data store, the sniffer details are not applicable.

## Viewing MDM logs

MDM facilitates monitoring of critical processes that are associated with it. To take corrective and preventive actions, you can view logs. As an MDM user, you can monitor the logs of the following activities:

- Bulk operations
- Synchronization
- Publishing

### Before you begin:

You must have the role of a MDM administrator or MDM monitor for viewing MDM logs.

### To view MDM logs:

1. On the Welcome page, click (MDM Data Steward Cockpit).  
The MDM Data Steward Cockpit window opens with a list of published MDM models.
2. You can view MDM logs in any one of the following ways:
  - Click the published MDM model, and click (View Logs) from the tool bar.
  - Right-click the MDM model and select View Logs.  
You can do the following in the Log Viewer - MDM Model window.

Operation	Description
Synchronization	Select the filter.

Operation	Description
	<p><b>Subscriptions</b> - Select the subscription.  If no subscription is selected, all the subscriptions from the spoke to the hub are selected.</p> <ul style="list-style-type: none"> <li>• <b>Operation Name</b> - Select the type of operation performed on the database. <ul style="list-style-type: none"> <li>• <b>All</b> - All logs are displayed irrespective of the type of operation performed.</li> <li>• <b>Specific</b> - Select one or more types of operation: <b>Insert</b>, <b>Update</b>, and <b>Delete</b>.</li> </ul> </li> <li>• <b>Status</b> - Select the status of the operation. <ul style="list-style-type: none"> <li>• <b>All</b> - All logs are displayed irrespective of the status of the operation.</li> <li>• <b>Specific</b> - Select one or more types of status: <b>Completed</b>, <b>Failed</b>, and <b>Pending</b>.</li> </ul> </li> <li>• <b>Time</b> - Select the time of the operation. <ul style="list-style-type: none"> <li>• <b>All</b> - All logs are displayed irrespective of the time of the operation.</li> <li>• <b>Display for the last</b> - To view logs from a time period of a specific length that moves continuously, which can be year, month, week or day.</li> <li>• <b>Between</b> - To view logs for the range falling within the specified From and To dates.</li> </ul> </li> </ul>
Publishing	<ul style="list-style-type: none"> <li>• <b>Status</b> - Specify the status of the operation. <ul style="list-style-type: none"> <li>• <b>All</b> - All logs are displayed irrespective of the status of the operation.</li> <li>• <b>Specific</b> - Select types of status: Completed, Stopped, and Started.</li> </ul> </li> <li>• <b>Time</b> - Select the time of the operation. <ul style="list-style-type: none"> <li>• <b>All</b> - All logs are displayed irrespective of the time of the operation.</li> <li>• <b>Display for the last</b> - To view logs from a time period of a specific length that moves continuously, which can be year, month, week or day.</li> <li>• <b>Between</b> - To view logs for the range falling within the specified From and To dates.</li> </ul> </li> </ul>

Operation	Description
Bulk Operations	<p>You can select the following filters:</p> <ul style="list-style-type: none"> <li>• <b>Subscription Details</b> - Select the subscription. If no subscription is selected, all the subscriptions from the spoke to the hub are selected.</li> <li>• <b>Status</b> - Select the status of the operation.             <ul style="list-style-type: none"> <li>• <b>All</b> - All logs are displayed irrespective of the status of the operation.</li> <li>• <b>Specific</b> - Select types of status: Completed, Done with errors, Stopped, and Started.</li> </ul> </li> <li>• <b>Time</b> - Time of the operation.             <ul style="list-style-type: none"> <li>• <b>All</b> - All logs are displayed irrespective of the time of the operation.</li> <li>• <b>Display for the last</b> - To view logs from a time period of a specific length that moves continuously, which can be year, month, week or day.</li> <li>• <b>Between</b> - To view logs for the range falling within the specified From and To dates.</li> </ul> </li> </ul>

3. To view the logs, click **Get Logs**.

In the right pane, logs window opens with Process ID, Status, Start Time, and End Time, with the following options on the logs window.

Icons	Options	Description
	Delete	The selected logs are deleted.
	Purge	All logs are deleted.
	Refresh	The latest changes reflect in the UI.
	Show/Hide Filters	The Log Filters pane is displayed or hidden.

**Note:** To clear the selections made in the previous step, click **Reset**.

If more than one type of log is selected, the logs are displayed in tabs. Double-click a log to view its details in More Details - <Type of log>.

# Chapter 22

## BAM standard dashboard

Business Activity Monitor (BAM) standard dashboard provides an overview of all the process instances available in the current organization and their related drill-downs to enable the administrators to view the details about the same. It consists of five types of data views.

Type	Overview	Description
Operational dashboard	An Operational dashboard provides an overview of all the process instances in the current organization.	<p>The operational data can be viewed in the following ways:</p> <ul style="list-style-type: none"><li>■ Total number of instances for all the processes in the current organization. This data can be grouped in terms of time period such as hours, days, weeks, months, and years.</li><li>■ Total number of instances for each status for all the processes in the current organization.</li><li>■ Top 'X', where 'X' is a positive integer referring to the processes having the highest number of instances for each status.</li></ul> <p>The available statuses on a process are Complete, Waiting, Aborted, Suspended, and Terminated.</p> <p>The data is populated automatically on the Operational dashboard when a BAM standard dashboard is opened from the Welcome page. By default, the data related to the last one week is reflected on the dashboard. This can be increased or decreased by modifying the <b>Time Frame</b> option available at the top of the dashboard and clicking </p>

Type	Overview	Description
		(Refresh). See <a href="#">Operational-level views</a> for more information on composite controls used on the Operational dashboard.
Process dashboard	A Process dashboard provides the overview of a specific process model in the current organization. The data monitored on the Process dashboard is process-specific.	<p>The process data can be viewed in the following ways:</p> <ul style="list-style-type: none"> <li>■ Total number of the process instances in a specific period. The related data can be grouped in terms of hours, days, weeks, months, and years.</li> <li>■ Average cycle time for all the instances of the process.</li> <li>■ Total number of the process instances in the current organization for each status. The available statuses on the process are Complete, Waiting, Aborted, Suspended, and Terminated.</li> <li>■ Total number of the activity instances for a process for each status in the current organization. The available statuses on the activity are Complete, Waiting, and Aborted.</li> <li>■ Average cycle time of each activity for all the instances of a process.</li> </ul> <p>A cycle time of a process instance is the time that it has taken to complete, calculated as the time between process start time and process end time. The data related to the cycle time can be viewed in the units of seconds, minutes, hours, days, weeks, months, and years.</p> <p><b>To view the data on the Process dashboard:</b></p> <ol style="list-style-type: none"> <li>1. In <b>Process Selection</b>, select <b>Process Name</b>.</li> <li>2. Provide the time frame and click  (Refresh).</li> </ol>

Type	Overview	Description
		<p>This data can also be drilled down from the Operational dashboard.</p> <p>3. Click the process populated in the <b>Top Processes for Status</b> pie-chart of the Operational dashboard. The data related to that process is populated on the Process dashboard.</p> <p>See Process-level views in the <i>AppWorks Platform Advanced Development Guide</i> for more information on the composite controls used on the Process dashboard.</p>
Instance dashboard	An Instance dashboard provides the top 'X', a positive integer that refers to the instances per status for a particular process model having the highest lead time in the current organization.	<p>A lead time of a process instance is the time that it has taken for processing. The data related to the lead time can be viewed in the units of seconds, minutes, hours, days, weeks, months, and years for the following process instances:</p> <ul style="list-style-type: none"> <li>■ For a process instance having any one of the following statuses such as Complete, Suspended, Aborted, Terminated, the lead time is the difference between the process end time and the process start time.</li> <li>■ For a process instance having a Waiting status; the lead time is the difference between the current time and the process start time.</li> </ul> <p>The lead time can also be grouped in terms of seconds, minutes, hours, days, weeks, months, and years.</p> <p><b>To view the data on the Instance dashboard:</b></p> <ol style="list-style-type: none"> <li>1. Click the data point on the instance lead time graph to obtain a graphical representation of the process status.</li> <li>2. Select the number of instances to be displayed such as the lead time</li> </ol>

Type	Overview	Description
		<p>units and status, and click  (Refresh). This data can also be drilled down from the Process dashboard.</p> <p>3. Click the status populated in the <b>Process instances per status</b> bar chart of the Process dashboard. The data related to that status is populated on the Instance dashboard.</p> <p>See <a href="#">Instance-level views</a> for more information on the composite controls used on the Instance dashboard.</p>
Activity dashboard	An Activity dashboard provides an overview of a specific activity for the process in the current organization.	<p>The activity data can be viewed in the following ways:</p> <ul style="list-style-type: none"> <li>■ Total number of the activity instances for each Status. The available statuses per activity are Complete, Waiting, and Aborted.</li> <li>■ Total number of tasks for each user based on the task status in the current organization. The available statuses per task are Complete, Waiting, Aborted, Skipped, and Suspended.</li> <li>■ Average cycle time for the activity in all the instances of a process.</li> <li>■ Average cycle time for the activity in all the instances for each user.</li> </ul> <p>The cycle time of an activity is the time that it has taken to complete; calculated as the time between the activity end time and the activity start time.</p> <p>The data related to the cycle time can be viewed in units of seconds, minutes, hours, days, weeks, months, and years and can also be grouped in terms of time period such as hours, days, weeks, months, and years.</p>

Type	Overview	Description
		<p><b>To view the data on the Activity dashboard:</b></p> <ol style="list-style-type: none"> <li>1. In <b>Activity Selection</b>, select <b>Process Name, Activity Name</b>.</li> <li>2. Provide the time frame and click  (Refresh). This data can also be drilled down from the Process dashboard .</li> <li>3. Click either the status data of an activity populated on the <b>No. of Instances per Activity by Status</b> bar-chart or the <b>Cycle time</b> point of an activity in the <b>Cycle time per Activity</b>.</li> </ol> <p>See Activity-level views in the <i>AppWorks Platform Advanced Development Guide</i> for more information on the composite controls used on the Activity dashboard.</p>
User dashboard	A User dashboard provides an overview of tasks assigned to a particular user. The data monitored on the User dashboard is user-specific.	<p>The user data can be viewed in the following ways:</p> <ul style="list-style-type: none"> <li>■ Number of tasks completed, scheduled, and outstanding for a specific process for the user.</li> <li>■ Number of tasks completed, scheduled, and outstanding for all the processes in the current organization for the user.</li> </ul> <p>The above data can also be grouped in terms of the time period such as hours, days, weeks, months, and years.</p> <p><b>To view data on the User dashboard:</b></p> <ol style="list-style-type: none"> <li>1. In <b>User Selection</b>, select <b>Process Name, Activity Name</b>, and <b>User Name</b>.</li> <li>2. Provide the time frame and click  (Refresh). This data can also be drilled down from the Activity dashboard.</li> </ol>

Type	Overview	Description
		<p>3. Click the required user data populated in either the <b>No. of Tasks per User by Status</b> or <b>Cycle Time for Activity per User</b>' bar chart.</p> <p>See User-Level Views in the <i>AppWorks Platform Advanced Development Guide</i> for more information on the composite controls used on the User dashboard.</p>



# Part III

# JMX

## Chapter 23

# Managing operations through Java Management Extensions

Java Management Extensions (JMX) is a Java-based technology that provides tools for managing and monitoring devices (printers), applications and other service-driven networks. The resources that must be managed are represented by managed components (JMX Managed Bean). See <http://www.oracle.com/technetwork/java/javase/tech/javamanagement-140525.html>.

The JMX architecture enables exposing managed components through various protocols. See [JMX architecture](#). You can add the standard RMI protocol and other connectors such as protocol adaptors. In AppWorks Platform, connectors are available for the SNMP and SOAP protocols.

## Applications

Consider an e-commerce site where online orders are placed for a variety of products. To measure the accuracy in the site performance, continuous monitoring of certain runtime parameters is essential. For example, the administrator might need to monitor the number of hits and change the configuration settings accordingly.

Operational management is the process where the administrator continuously monitors and manages the runtime behavior of the system by accessing the data related to the state of the system, performance statistics, and user activities.

Adding JMX support to the tools enables creation of a direct plug-in with the Java-enabled programs. Several large management tools from other organizations have added JMX support to their tools.

Besides commercial tools, free management consoles such as Sun ships JConsole (part of Java 5), MC4J, JManage, and XtremeJ (Eclipse Plug-in) are also available. See [JConsole](#).

## Management interface

The management interface exposes the business components of an application that require monitoring and management. It collects performance figures and reads the settings from different components of the application. Developers can develop components that

administrators can manage and monitor from any management console that supports JMX. See [Management interface](#) and [Implementing Java Management Extensions in applications](#).

Managed Beans (MBeans) provides a component hierarchy:

- Attributes: Named, typed values that can be read or written. Attributes can be of primitive types such as `int` and `string` or complex types such as structures and tables. Read-only attributes often expose status information while read or write attributes define settings.
- Operations: Methods that can be invoked through JMX and are defined with a name, parameter list, and a return type.
- Notifications: Alerts that notify the administrator about major application events. AppWorks Platform alerts are exposed as JMX notifications.

The Mbeans of JConsole exposes managed components of AppWorks Platform, which must be monitored and managed through JMX.

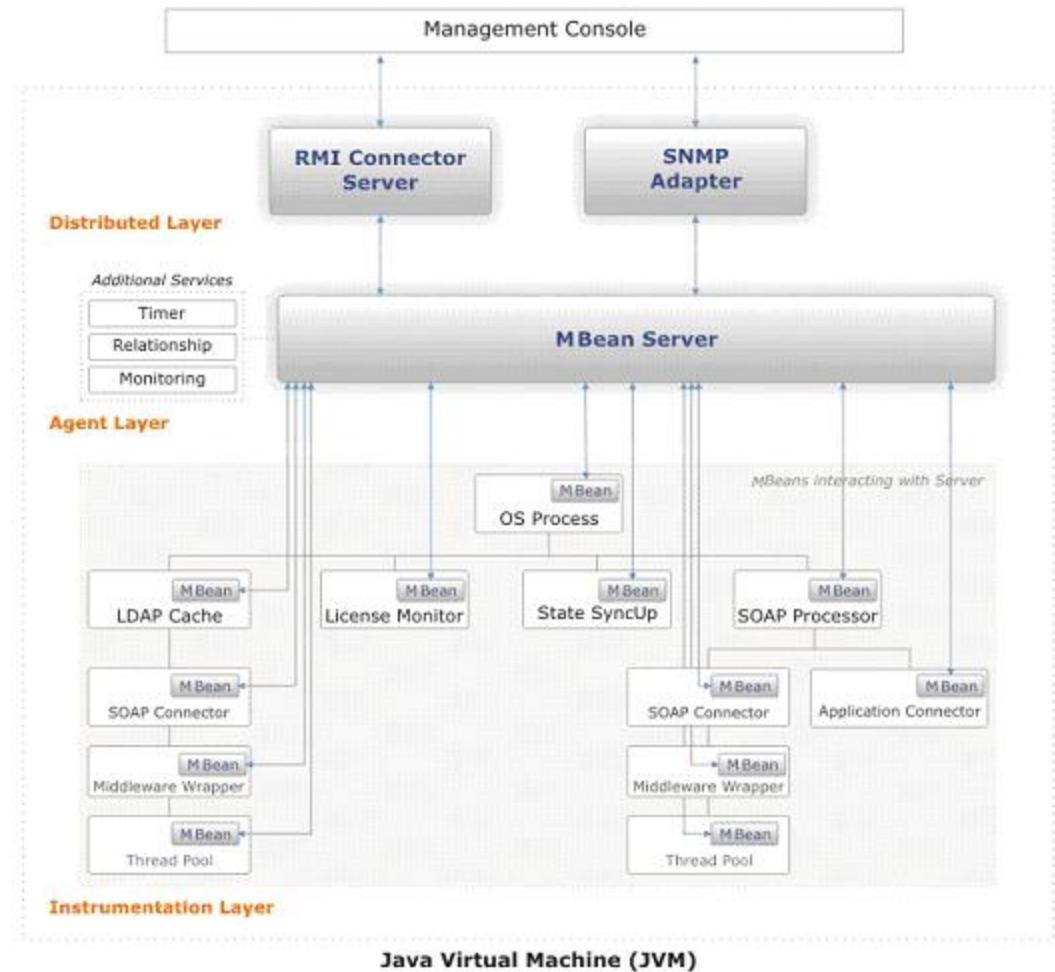
For more information, see [Connecting to JMX through JConsole](#), [Monitoring managed components](#), and [Managed component types](#).

# Chapter 24

## JMX architecture

The JMX architecture comprises the following three layers:

- Distributed layer
- Agent layer
- Instrumentation layer



## Distributed layer

The distributed layer exposes the JMX agents to the management applications through protocols such as SNMP, adapters, and connectors. Different connector types expose the API agent to other distributed technologies such as RMI technology.

## Agent layer

The agent layer, which holds the runtime environment of the Mbean server consists of adaptors and connectors that provide the managed component services such as monitoring an application, loading the managed components, and defining relationships.

The MBean Server maintains a directory of the registered services. The management framework detects only the registered managed components. Also, the MBean server handles the management messages that are sent between registered managed components.

## Instrumentation layer

The instrumentation layer contains the managed components and their manageable resources such as service, middleware, and SOAP connector. The managed components and the association between them are displayed in the instrumentation layer.

## Chapter 25

# Implementing Java Management Extensions in applications

The concept of a management interface draws its inference from an application resource that provides some value to the user. An application resource here refers to the business component within an application or an infrastructure component such as a cache, a queue, or the entire application. An application resource in the context of JMX (Java Management Extensions) encapsulates only those resources that must be managed.

Each such resource that requires management must provide a management interface. A management application must monitor and control the attributes and operations exposed by a management interface.

The management interface concept can be applied to AppWorks Platform that supports operational management through JMX. The management interface can collect performance figures from different components of AppWorks Platform and read their settings.

This management interface comes as a handy tool to the AppWorks Platform application developers for developing applications that can be managed and monitored from any management console that supports JMX.

The above definition can be substantiated using an example of a queue that must be managed in an application. A queue is temporarily used to store logical units of work until they are taken away from the queue for further processing. This is done asynchronously through a multi-threaded design. Therefore, the queue must also be thread-safe. Consider that the application is characterized by a single supplier thread and a single consumer thread. The queue is assumed to be thread-safe, ensuring that the user or the supplier waits until an item is added to the queue if it is empty. On similar lines, the consumer that picks the items from the queue for further processing waits to remove an item if the queue is empty.

The next step is to identify specific information that can assist in monitoring the application. In the context of the queue, the significant information that could require monitoring is the time duration for which a supplier waits to add an item to the queue because the queue is full and a consumer waits to remove an item from the queue because the queue is empty. Therefore, this information can be exposed as two attributes on the management interface of the queue.

This section contains code samples and best practices:

- [Managed components](#)
- [JMX settings for managed components](#)
- [JMX counters](#)
- [Alerts](#)
- [Problems](#)
- [Operations](#)
- [InfoAttribute](#)

## Alerts

Alerts inform an administrator or administration application about major application events. Alerts in AppWorks Platform are exposed as JMX notifications that can be monitored from any JMX management tool. These alerts are published through Log4J to be forwarded as an SMS or email message through appenders.

### Defining an alert

The `defineAlert` method helps define an alert for a managed component in an application. This method returns an `IAlertDefinition` instance that the application must retain. While defining an alert, specify one of the following alert levels that are issued:

- **ERROR:** when a fatal error occurs
- **WARNING:** as a signal to a potentially upcoming error alert
- **INFORMATIONAL:** when a major informational event occurs (starting or stopping a service) or a warning condition clears

The following example illustrates different types of alerts:

```
// Defining an alert
IAlertDefinition m_indexCorruptAlert = managedComponent.defineAlert
(AlertLevel.ERROR, CoBOCMessages.INDEX_CORRUPT_ALERT_MESSAGE,
CoBOCMessages.INDEX_CORRUPT_ALERT_DESCRIPTION);
```

### Issuing an alert

To issue an alert, call the `issueAlert` Web service operation.

Issuing an alert:

- delegates the alert to the alert system for further processing depending on the consumers
- issues a JMX notification

```
try {
    ...
    ...
} catch (IndexCorruptException ex) {
    // Issue an alert
    m_indexCorruptAlert.issueAlert(ex, indexName);
}
```

## JMX counters

Counters are defined on a managed component. They expose performance data to systems managers and health analysis programs. Performance counters created for managed components are exposed as JMX attributes. The `createPerformanceCounter` method creates different types of counters. CounterFactory acts as a factory to create different types of performance counters.

### Types of counters

Counters can be of many types. For information about statistics of each counter type, see [JMX counter statistics](#).

#### **Value counter**

A value counter is similar to a thermometer that exposes an internal value, which is retrievable anytime. A thermometer has no statistics. To calculate an average temperature, you must sample the thermometer.

A regular value counter requires that the counter's `setCurrentValue` is invoked when the counter's value changes. This is not a very common counter type. The property-based value counter is easier to use.

#### **Example**

```
// Defining the counter
IValueCounter ctr = (IValueCounter) m_managedComponent.createPerformanceCounter(
    "temperature",
    Messages.CURRENT_TEMP_DESC,
    CounterFactory.VALUE_COUNTER);
// Setting a value on it
ctr.setCurrentValue(32);
```

#### **Property-based value counter**

A property-based value counter is functionally identical to a value counter, but it invokes the getter method of the related property when queried for a value.

**Example (taken from com.eibus.util.threadpool.Dispatcher)**

```
// Defining the counter
m_managedComponent.createPropertyBasedValueCounter(
"numIdleWorkers",
Messages.NO_IDLE_WORKERS_DESC
"numIdleWorkers",
this);
```

***Event occurrence counter***

An event occurrence counter registers each occurrence of an event. For example, it can track the number of rain showers per day. The counter has a configurable moving average window in which it calculates the number of events per second. It also tracks the number of events since the counter reset.

To add an event to an event occurrence counter, call `addEvent`.

**Example (taken from com.eibus.directory.soap.Cache)**

```
// Defining the counter
IEventOccurrenceCounter m_lookupCounter =
(IEventOccurrenceCounter) m_managedComponent.createPerformanceCounter(
"lookups",
Messages.CACHE_LOOKUPS_DESC",
CounterFactory.EVENT_OCCURRENCE_COUNTER);
// Adding an event to the counter
m_lookupCounter.addEvent();
```

***Event value counter***

An event value counter registers the value of each event occurrence. For example, it can track the amount of rain per shower. Apart from the statistics collected by an event occurrence counter, it tracks the total value since the counter reset and uses the moving average window to calculate the minimum, maximum, standard deviation, and average event value (amount of rain).

To add an event to an event value counter, call `addEvent`.

**Example (taken from com.eibus.web.tools.download.Download)**

```
// Defining the counter
IEventValueCounter m_downloadSizeCounter = (IEventValueCounter)
managedComponent.createPerformanceCounter(
"downloadSize",
Messages.DOWNLOAD_SIZE_DESC,
CounterFactory.EVENT_VALUE_COUNTER);
// Adding an event to the counter
m_downloadSizeCounter.addEvent(fileSize);
```

## Timer event value counter

The Timer event value counter is a variant of the event value counter. The counter value is the event duration in milliseconds.

To add an event to a timer event value counter:

- Before the event occurs, call `start`, which returns a long value.
- After the event occurs, pass the long value in the `finish` call.

### Example (taken from com.eibus.web.isapi.WebApplication)

```
// Defining the counter
ITimerEventValueCounter m_requestHandlingDurationCounter =
(ITimerEventValueCounter)
m_managedComponent.createPerformanceCounter(
"requestHandlingDuration",
Messages.REQUEST_HANDLING_FREQUENCY_DESC,
CounterFactory.TIMER_EVENT_VALUE_COUNTER);
// Adding an event to the counter
long startTime = m_requestHandlingDurationCounter.start();
service(ecb);
m_requestHandlingDurationCounter.finish(startTime);
```

## Using JMX counters

You can use JMX counters provided in the AppWorks Platform components for the following purposes:

### Diagnosing performance issues

Occasionally, business applications encounter performance issues such as high transaction response times and unresponsive service containers. To diagnose such issues, AppWorks Platform provides JMX counters in each component covering all the critical processing layers.

Diagnosing performance issues can be classified into the following:

Diagnosis	Description
Basic	This requires basic knowledge of AppWorks Platform including knowledge about the AppWorks Platform components involved in business transactions and other business components such as the database with which the AppWorks Platform components interact during the business transactions.
Advanced	This requires basic and advanced knowledge of AppWorks Platform including the AppWorks Platform subcomponents, available MBean objects, and corresponding JMX counters.

## Common performance problems

Following are the commonly encountered performance problems.

Problem	Solution
High transaction response times	<p>Check the <code>ctrv_requestHandlingTimer_current</code> counter value in all the service containers that are participating in slow response time transactions and isolate the service container. The following list describes few instances of its occurrence and suggested workarounds.</p> <ul style="list-style-type: none"> <li>■ Service container itself is contributing to the high response time: Check the JMX counters of subcomponents, for example, counters related to database queries and caches.</li> <li>■ Resources with which this service container is interacting are performing slowly, for example, database servers: Check the JMX counters of the corresponding resources.</li> </ul> <p>If a combination of the described instances occur, use a combination of the described workarounds.</p>
High database response times	<p>Check the <code>ctrv_readConnectionUsageDuration_current</code> and <code>ctrv_updateConnectionUsageDuration_current</code> counters of the database connection pool MBean available in the corresponding service container. They specify the time taken by database read and update queries respectively. Attention is required if the time taken increases continuously or is more than expected.</p>
Service container memory issues - Memory leaks	<ul style="list-style-type: none"> <li>■ Check the <code>ctrv_virtualMemoryUsage_current</code> (for Windows OS) or <code>ctrv_residentMemoryUsage_current</code> (for Linux OS) counters. If the value continuously increases with the usage, it indicates a memory leak or poor garbage collection.</li> <li>■ Check the layer responsible for the memory increment of AppWorks Platform NOM, Java heap, or other native layer. For the AppWorks Platform NOM memory, check the <code>ctrv_NOMNodePoolMemory_current</code> counter.</li> </ul>
Service container memory issues - High memory utilization	<ul style="list-style-type: none"> <li>■ Check the <code>ctrv_virtualMemoryUsage_current</code> (for Windows OS) or <code>ctrv_residentMemoryUsage_current</code> (for Linux OS) counters and verify whether any values are higher than expected.</li> <li>■ Check if the high memory utilization is due to slow garbage collection.</li> </ul>
Service container memory issues - Insufficient memory	<p>Occasionally, the default memory settings might be insufficient. Check the <code>ctrv_virtualMemoryUsage_current</code> (for Windows OS) or <code>ctrv_residentMemoryUsage_current</code> (for Linux OS) counters</p>

<b>Problem</b>	<b>Solution</b>
settings	and take appropriate actions.
Service container memory issues - NOM document leaks	<p>Check if the <code>ctrv_NOMNodePoolMemory_current</code> and <code>ctrv_numOfNOMNodes_current</code> counter values continuously increase with the NOM usage. If yes, it indicates a memory leak in the NOM usage.</p> <p><b>Note:</b> Leaks might be related to AppWorks Platform or the custom logic developed for the business application.</p>
Insufficient cache sizes	<ul style="list-style-type: none"> <li>■ For LDAP cache, check if the <code>ctrv_cacheMisses_persecond</code> counter value is high or increasing with the usage. If yes, then consider expanding the LDAP cache size with the <code>bus.ldap.cache.maxSize</code> property in the <code>wcp.properties</code> file.</li> <li>■ For some service containers, <code>ctrv_numCacheMisses_current</code> captures cache misses for a service container. A high value of this indicates that the current service container cache is insufficient with the current usage.</li> </ul>
Insufficient database connections	<ul style="list-style-type: none"> <li>■ Check if the <code>ctrv_readConnectionWait_current</code> and <code>ctrv_writeConnectionWait_current</code> values are greater than zero. If yes, it indicates that the connection pool size is insufficient and must be increased.</li> </ul>
High response times from an external Web server	<ul style="list-style-type: none"> <li>■ Check for the <code>ctrv_sendAndWaitTimer_current</code> value in the SOAP Connector Mbean of the UDDI service container to know the response time when a request is sent to the external Web server.</li> </ul>

## Frequently used JMX counters

### Common to all service containers

<b>Collection name</b>	<b>Counter</b>	<b>Description</b>	<b>Needs attention if</b>
Service container CPU time	<code>ctrv_currentCPUTime_current</code>	CPU time in the number of clock-ticks being consumed (this counter value must be converted into %)	a high CPU utilization is observed for prolonged periods

<b>Collection name</b>	<b>Counter</b>	<b>Description</b>	<b>Needs attention if</b>
		of CPU utilization)	
Service container memory utilization	ctrv_virtualMemoryUsage_current	Virtual memory usage	it is continuously increasing; indicates risk of a crash
	ctrv_residentMemoryUsage_current	Resident memory usage	it is continuously increasing; indicates risk of exhausting memory and leads to a crash
NOM memory counters	ctrv_NOMNodePoolMemory_current	NOM node pool memory (specified in bytes)	it is continuously increasing; indicates risk of exhausting memory and leads to a crash
	ctrv_NOMStringPoolMemory_current	NOM string pool memory	it is continuously increasing; indicates risk of exhausting memory and leads to a crash
	ctrv_numOfNOMNodes_current	Number of NOM nodes	it is continuously increasing; indicates risk of exhausting memory and leads to a crash.
NOM memory counters	ctrv_numOfXPathObjects_current	Number of XPath objects	it is continuously increasing; indicates risk of exhausting memory and leads to a crash
	ctrv_numOfXPathExpressionObjects_current	Number of XPath expression objects	it is continuously increasing; indicates risk of exhausting

<b>Collection name</b>	<b>Counter</b>	<b>Description</b>	<b>Needs attention if</b>
			memory and leads to a crash
	ctrv_numOfNOMDocuments_current	Number of current NOM documents	it is continuously increasing; indicates risk of exhausting memory and leads to a crash
	ctrv_NodeObjectInstanceCount_current	Node object instance count	it is continuously increasing; indicates risk of exhausting memory and leads to a crash
Request handling time	ctrv_requestHandlingTimer_current	Request processing time	it displays a high value or is continuously increasing; indicates poor performance
	ctrv_requestHandlingTimer_persecond	Number of requests processed per second	it displays a low value or is continuously decreasing; indicates poor performance.
	ctrv_requestHandlingTimer_average	Average request processing	it displays a high value or is continuously increasing; indicates poor performance.
Cache	ctrv_cacheMisses_persecond	Cache misses per second	it displays inadequate cache size
	ctrv_cacheHits_persecond	Cache hits per second	the cache hits are zero cache
	ctrv_removes_persecond	Cache removes per second	it displays a high value; a high

<b>Collection name</b>	<b>Counter</b>	<b>Description</b>	<b>Needs attention if</b>
			value indicates inefficient cache
	ctrv_lookups_persecond	Cache lookups per second	it displays a low value; a low value indicates that the cache is not utilized efficiently
DB connection pool (applicable to those service containers with DB connectivity, for example, BPM, Notification)	ctrv_readConnectionUsageDuration_current	Time to read database queries	it is increasing continuously or takes more time than expected
	ctrv_writeConnectionUsageDuration_current	Time to update database queries	it is increasing continuously or takes more time than expected
	ctrv_readConnectionWait_current	Application thread waiting time to get a connection for the DB connection pool	it displays a positive value; a zero value indicates that the configured DB connection pool size is sufficient and a positive value indicates that the DB connection pool size is insufficient
DB connection pool (applicable to those service containers with DB connectivity, for example, BPM, Notification)	ctrv_writeConnectionWait_current	Application thread waiting time to get a connection for the DB write connection pool.	it displays a positive value; a zero value indicates that the configured DB connection pool size is sufficient and a positive value indicates that the DB connection pool size is insufficient

## Monitoring health and performance

Administrators continuously monitor the health and performance of the critical service containers involved in the business to take proactive actions and avoid unwanted problems.

For information about monitoring AppWorks Platform, see [Managing operations through Java Management Extensions](#).

**Note:** Performance issues can be related to AppWorks Platform or the custom logic in the business application that is developed on top of AppWorks Platform. Therefore, while diagnosing issues, it is important to isolate the problem either to the AppWorks Platform layer or the custom logic layer.

JMX management can be done through commercial management tools such as IBM (Tivoli), CA (Unicenter), HP (OpenView), and BMC (Patrol) or through free management tools, for instance, the JConsole. See [Connecting to JMX through JConsole](#).

### Most used service container-specific counters

Service container/subcomponent	Counter	Description
BPM	ctrv_numAbortedProcesses_current	Number of aborted processes. It needs attention if the value is increasing with the usage.
	ctrv_numActiveThreads_current	Number of active threads of the process engine that are responsible for processing long-lived flows.
BPM	ctrv_numFreeThreads_current	Number of free process engine threads that are responsible for processing long-lived flows.
	ctrv_numCompletedProcesses_current	Number of completed processes.
	ctrv_numTerminatedProcesses_current	Number of

<b>Service container/subcomponent</b>	<b>Counter</b>	<b>Description</b>
		terminated processes.
	ctrv_waitingRequestsQueueSize_current	Number of awaiting requests. If it is continuously increasing, it indicates that the service container might crash soon. Either increase the queue size or investigate the cause of the slow request processing.
Rule	ctrv_ruleActionExecutionRate_current	Number of rule actions executed per second.
Rule	ctrv_dispatcherQueueSize_current	Number of awaiting rules in the dispatcher queue. It needs attention if the value is continuously increasing.
Rule	ctrv_ruleExecutionRate_current	Number of rules executed per second.
WS-Apps	ctrv_noOfActiveTransaction_current	Number of active transactions. It needs attention if it is increasing continuously.
WS-Apps	ctrv_numberOfOpenCursors_current	Number of open cursors. It needs attention if it is increasing continuously.

<b>Service container/subcomponent</b>	<b>Counter</b>	<b>Description</b>
Notification	ctrv_noOfMessageDispatchedToMailBox_current	Displays the number of messages dispatched to the mail box.
	ctrv_noOfMessagesDispatchedToInbox_current	Displays the number of messages dispatched to the Inbox.
	ctrv_noOfMessagesDispatchedToTaskIndexer_current	Displays the number of messages dispatched to the task indexer.
	ctrv_noOfMessagesInQueue_current	Displays the number of messages in the queue
UDDI	ctrv_sendAndWaitTimer_current	Time taken to receive the response from the external server. It needs attention if there are high response times.

For information about the AppWorks Platform JMX counters, see [Managed component types](#). The [AppWorks Developer community](#) provides additional resources that you may find helpful.

## Operations

The operations defined on a managed component are exposed as JMX operations in the managed bean. For example, the operations for the managed component of the monitor application connector include start, stop, and reset. The input of the operation accepts the Distinguished Name (DN) of the service container as the parameter and performs the respective operations on the service containers.

Based on the impact, the operations are classified into the following types:

- ACTION: The operation performs the write function and modifies the managed bean by writing a value or changing a configuration.
- ACTION\_INFO: The operation performs both read and write functions.
- INFO: The operation performs the read function and retrieves information.
- UNKNOWN: The nature of the operation is unknown.

### Defining an operation

An operation is defined by invoking `defineOperation` on a managed component. The operation is implemented through a Java Web service operation where static and non-static methods can be used.

The following is an example of a static Web service operation in which `Cache.class` is passed as an argument.

```
m_managedComponent.defineOperation ("clearCache",Messages.CACHE_CLEAR_DESC,"clearCache",
Cache.class,null,OperationImpact.ACTION);
```

The following is an example of a non-static Web service operation. The object in it is passed as an argument because the Web service operation is non-static.

```
m_managedComponent.defineOperation
("getSize",Messages.CACHE_READ_DESC,"getSize",this,null,OperationImpact.INFO);
```

## Problems

A problem is the state of the component when it does not function in an expected manner. The management subsystem allows applications to perform the following tasks:

- Define a problem with the cause and clear alert messages.
- Raise a defined problem when the problem occurs.
- Resolve the problem through a problem resolver handler.

### Defining a problem

The `defineProblem` method defines a problem for a managed component in an application. It returns an `IProblemDefinition` instance that the application must retain to raise the problem.

```
// defining a problem
IProblemDefinition m_problemDef = defineProblem()
```

```
Messages.CONNECTION_LOSS_RAISE_ALERT_MESSAGE,
Messages.CONNECTION_LOSS_RAISE_ALERT_DESCRIPTION,
Messages.CONNECTION_LOSS_CLEAR_ALERT_MESSAGE,
Messages.CONNECTION_LOSS_CLEAR_ALERT_DESCRIPTION);
```

## Raising a problem

The `Raiseproblem` method raises a problem. When calling the method, the component must pass a problem resolver that implements `IProblemResolver`. The problem resolver thread periodically invokes the `resolve` method and verifies whether it can resolve the problem.

```
// raising a problem
long recheckInterval = 120*1000;
m_problemDef.raiseProblem(new MyProblemResolver(),
recheckInterval,
null);
```

You can also raise a problem with arguments.

```
// raising a problem with arguments
long recheckInterval = 120*1000;
m_problemDef.raiseProblem(new MyProblemResolver(),
recheckInterval,
new Date());
```

## Resolving a problem

The problem resolver periodically calls the `resolve` method and verifies whether it can resolve the problem. After calling the method, `IProblemStatusEventListener` is passed after which the problem resolver invokes methods on this interface and informs the management subsystem about the problem status.

```
//resolving a problem
class MyProblemResolver implements IProblemResolver
{
public void resolveProblem(IProblemStatusEventListener
problemStatusEventListener)
{
try
{
reconnect();
problemStatusEventListener.notifyResolved();
}
catch (ConnectionFailureException ex)
{
problemStatusEventListener.
```

```
    notifyNotResolved(recheckInterval);
}
}
}
```

## JMX settings for managed components

Settings are the knobs that tweak and tune the system performance. They are exposed as read/write or write-only attributes of JMX. Examples of settings are pool size and maximum queue length.

### Setting types

Based on their implementation, settings can be of two types.

#### Hot Settings

In a hot setting, the change persists through the settings persister and invokes the related setter Web service operation.

When defining a hot setting, providing the setter Web service operation is required to ensure that the setting is applied in the running process.

You can select the setting to be read/write or write-only by choosing the appropriate Web service operation to define the setting. Usually, passwords use write-only settings.

#### Cold settings

In a cold setting, the change is applied only after restarting the service container. This setting type cannot be applied in the runtime execution of the process.

## Settings persistence

The settings persistence handler stores and retrieves the settings from the `wcp.properties` file or the `bussoapprocessorconfiguration` in LDAP.

`LDAPBusConfigSettingsPersistenceHandler` supports retrieving and storing the settings in the configuration file of the LDAP bus service container. The changes in the settings are logged under the `com.eibus.management.SettingDefinition` category.

## Defining settings

A collection of settings is required to define the settings. The Web service operation `{ {getSettingsCollection} }` of the managed component retrieves the settings collection. When no parameters are passed, the default settings collection is retrieved.

## Hot setting

A hot setting requires a setter Web service operation that is invoked when the setting value is changed. You must define the name of the setter Web service operation in the format `set<<propertyImplName>>`.

See the sample code for the setter Web service operation for `propertyImplName` `minConcurrentWorkers`.

```
public void setMinConcurrentWorkers(int minConcurrentWorkers)
```

This setter Web service operation is invoked before the setting value is stored in LDAP.

See the sample taken from `com.eibus.util.threadpool.Dispatcher` that depicts the definition of a hot setting.

```
ISettingsCollection settings = m_managedComponent.getSettingsCollection();  
  
If (settings != null)  
{  
    // There is a settings collection defined,  
    // so we can define settings.  
    int minConcurrentWorkers = (Integer) settings.defineHotSetting(  
        "minConcurrentWorkers",  
        Messages.MIN_CONCURRENT_WORKERS_COUNTER_DESC,  
        "minConcurrentWorkers",  
        this,  
        null,  
        m_minConcurrentWorkers,  
        0,  
        Integer.MAX_VALUE);  
}
```

## Cold setting

A cold setting does not require any setter Web service operation.

See the sample from `com.eibus.util.threadpool.Dispatcher` that describes the definition of a cold setting.

```
ISettingsCollection settings = m_managedComponent.getSettingsCollection();  
If (settings \!= null)  
{  
    // There is a settings collection defined,  
    // so we can define settings.  
    int maxPendingWorks = (Integer) settings.defineColdSetting(  
        "maxPendingWorks",  
        Messages.MAX_PENDING_WORKS_COUNTER_DESC,  
        null,  
        defaultMaxQueueSize,  
        0,  
        Integer.MAX_VALUE);  
}
```

## Write-only setting

See the sample from `com.eibus.applicationconnector.sql.DBConnectionPool` that describes the definition of a write-only setting.

```
ISettingsCollection settings = m_managedComponent.getSettingsCollection();
If (settings != null)
{
    // There is a settings collection defined,
    // so we can define settings.
    String password = (String) settings.
        defineWriteOnlyColdSetting("password",
        Messages.DB_PASSWORD_DESC,
        (String) legacyNames.get("password"),
        String.class);
}
```

## Complex-type setting

When defining a complex setting, ensure that the name of the complex-type setting is `<jmxApplications>`. Each `<settingValue>` represents an instance of the collection of settings and `<name>` identifies the value of the setting in the list. OpenText recommends that the name of the attribute be unique in structure.

Consider a service container with the following `bussoapprocessorconfiguration`.

```
<configurations ...
:
:
<configuration
implementation="com.eibus.applicationconnector.management.ManagementConnector"
htmfile="/cordys/wcp/admin/applicationconnector/management.htm">
*<jmxApplications>*
*<settingValue>*
*<name>Test</name>*
*<hostname>rnd0825</hostname>*
*<port>1099</port>*
*<jmxuser>hewa</jmxuser>*
*<jmxpassword>aGV3YQ==</jmxpassword>*
*</settingValue>*
*<settingValue>*
*<name>AnotherTest</name>*
*<hostname>rnd0825</hostname>*
*<port>1099</port>*
*<jmxuser>hewa</jmxuser>*
*<jmxpassword>aGV3YQ==</jmxpassword>*
*</settingValue>*
*</jmxApplications>*
</configuration>
:
:
</configurations>
```

In the code, `jmxApplications` is exposed as a complex setting that returns `NodeObject`.

```
ISettingsCollection settings = m_managedComponent.getSettingsCollection();

If (settings != null)
{
// There is a settings collection defined,
// so we can define settings.
NodeObject jmxApplications = (NodeObject)
settings.defineColdSetting(
"jmxApplications",
Messages.JMX_APPLICATIONS_DESC,
null,
NodeObject.class);
}
```

## Legacy setting

The settings for busoapprocessorconfiguration of the service container have been modified after the Cordys CU2 release. A legacy property supports the previous version of the settings. This property does not apply to new settings.

According to the new framework, the properties related to a service container must exist as child elements to `<configurations>`.

For example, you must add the settings such as `spyPublish` and `cancelReplyInterval` as child elements to the `configurations` node.

```
<configurations>
<spyPublish>false</spyPublish>
<cancelReplyInterval>30000</cancelReplyInterval>
<configuration implementation="com.eibus.applicationconnector.xmlstore.XMLStore"/>
</configurations>
All the application connector related settings must exist as child elements to
<configurations><configuration>.
For example the setting {{root}} is related to {{xmlstore}} application connector
and is present as child element to configuration node.
<configurations>
<spyPublish>false</spyPublish>
<configuration implementation="com.eibus.applicationconnector.xmlstore.XMLStore">
<root>D:\Program Files\Cordys\WCP\XMLStore</root>
</configuration>
</configurations>
```

The sample code describes the procedure to define the settings that contain legacy settings.

```
// Old configuration
<configurations cancelReplyInterval="30000">
<configuration implementation="com.eibus.applicationconnector.xmlstore.XMLStore"/>
```

```
</configurations>
// New configuration
<configurations>
<cancelReplyInterval>30000</cancelReplyInterval>
<configuration implementation="com.eibus.applicationconnector.xmlstore.XMLStore"/>
</configurations>
```

### Defining a setting with a legacy property

See the sample code that is derived from `com.eibus.soap.Processor` that describes the definition of a setting with a legacy property.

```
ISettingsCollection settings = m_managedComponent.getSettingsCollection();
long cancelReplyInterval = ((Long)
settings.defineHotSetting("cancelReplyInterval",
Messages.CANCEL_REPLY_DESC, "cancelReplyInterval",
requestMonitor,
LDAPBusConfigSettingsLegacySettingProvider.PREFIX +
"<configurations>@cancelReplyInterval",
new Long(30000),
new Long(0),
new Long(3600000))).longValue();
```

The settings provider reads the legacy properties from LDAP and WCP properties. In the above sample, the suffix `LDAPBusConfigSettingsLegacySettingProvider.PREFIX + "<configurations>@cancelReplyInterval"` represents that the settings are read from LDAP, that is, the `bussoapprocessorconfiguration` of the service container.

In the above code, the symbol '@' indicates that the property exists as an attribute. Since the `cancelReplyInterval` setting is defined using the managed component of the service container, the settings provider searches for the `cancelReplyInterval` attribute for the `<configurations>` element.

For example, if the legacy property is given as

`LDAPBusConfigSettingsLegacySettingProvider.PREFIX + "<configurations><cancelReplyInterval>"`, the settings provider searches for the `<cancelReplyInterval>` element as a child to `<configurations>`.

Initially, the settings provider searches for the settings in the new structure; if the settings are not found, it performs a search in the legacy path.

## InfoAttribute

Based upon the property of the class or the object, informational attribute is defined at the level of a managed component. The type of this attribute is read-only and can be retrieved through JMX.

**Example - 1**

The following example defines an informational attribute with the argument `PropertyDescriptor`.

```
void defineInfoAttribute(java.lang.String name, ILocalizableString
description, java.beans.PropertyDescriptor propertyDescr, java.lang.Object
sourceObject)
```

The following are the parameters.

<b>Parameter</b>	<b>Description</b>
<code>name</code>	Name of the informational attribute.
<code>description</code>	Description of the informational attribute, which must not be empty.
<code>propertyDescr</code>	Value for the informational attribute.
<code>sourceObject</code>	Source object on which to invoke the getter method. If the method is static, <code>sourceObject</code> refers to the class.

**Example - 2**

The following example defines an informational attribute of a managed component with the argument `propertyName`.

```
void defineInfoAttribute(java.lang.String name, ILocalizableString
description, java.lang.String propertyName, java.lang.Object sourceObject)
```

The following are the parameters.

<b>Parameter</b>	<b>Description</b>
<code>name</code>	Name of the informational attribute.
<code>description</code>	Description of the informational attribute, which must not be empty.
<code>propertyName</code>	Name of the property that supplies value for the informational attribute.
<code>sourceObject</code>	Source object on which to invoke the getter method. If the method is static, <code>sourceObject</code> refers to the class.

# Chapter 26

# Management interface

The management interface of AppWorks Platform is implemented as an abstraction layer for JMX, which supports concepts such as components, settings, info properties, counters, operations, alerts, and problems.

## Components

The components in AppWorks Platform are the application connectors that are exposed as JMX Managed Beans (MBeans).

## Settings

Settings are the knobs used by the administrator to tune the performance and scalability of the system. Conceptually, settings are the properties of the application connectors that are exposed as read/write or write-only attributes. Some of the settings that the administrator can configure are connection pool size, LDAP Cache, and thread pool size.

Settings are displayed under the Attributes node of MBean in the JMX Management tool. For a detailed description of the settings of each component, see [Managed component types](#).

Settings are classified into the following types.

- Hot: Hot settings are implemented immediately after the change and do not require a restart of the service container.
- Cold: Cold settings are implemented only after the restart of the service container.

## Info properties

Info properties are properties of managed objects that are only informational to the administrator. The info properties are exposed as read-only attributes and are available under the Attributes node of the JMX Management tool. The info properties are exposed as simple attributes that are displayed as `info_<propertyName>`. For example, the status of a service container is displayed as `info_status`.

## Counters

In the context of JMX, counters are exposed as read-only attributes and are available under the Attributes node of the JMX Management tool. For a detailed description of the counters, see [JMX counters](#).

The counters are exposed as complex type attributes that are displayed as `ctr_<counterName>`. For example, `cacheSize` is displayed as `ctr_cacheSize`. The statistics are exposed as `ctrv_<counterName>_<statistic>`.

Following are the counter types and the statistics for each counter type where `xxxx` is the placeholder for the counter name.

Counter type	Attributes	Example
Value Counter	<ul style="list-style-type: none"> <li>■ <code>ctr_xxxx</code>: Complex object bundling all the counter values.</li> <li>■ <code>ctrv_xxxx_current</code>: Value of the current counter.</li> </ul>	Counter Name: <code>totalCPUTime</code> <ul style="list-style-type: none"> <li>■ <code>ctr_totalCPUTime</code></li> <li>■ <code>ctrv_totalCPUTime_current</code></li> </ul>
Event Occurrence Counter	<ul style="list-style-type: none"> <li>■ <code>ctr_xxxx</code>: Complex object bundling all the counter values.</li> <li>■ <code>ctrv_xxxx_occurrences</code>: Number of events since the counter reset.</li> <li>■ <code>ctrv_xxxx_persecond</code>: Number of events per second in the moving average window.</li> </ul>	Counter Name: <code>NOMDocumentCreation</code> <ul style="list-style-type: none"> <li>■ <code>ctr_NOMDocumentCreation ctrv_NOMDocumentCreation_occurrences</code></li> <li>■ <code>ctrv_NOMDocumentCreation_persecond</code></li> </ul>
Event Value Counter	<ul style="list-style-type: none"> <li>■ <code>ctr_xxxx</code>: Complex object bundling all the counter values.</li> <li>■ <code>ctrv_xxxx_average</code>: Average value of the event in the moving average window.</li> <li>■ <code>ctrv_xxxx_current</code>: Last value of the last event.</li> <li>■ <code>ctrv_xxxx_maximum</code>: Maximum event value in the moving average window.</li> <li>■ <code>ctrv_xxxx_minimum</code>:</li> </ul>	Counter Name: <code>sendAndWaitTimer</code> <ul style="list-style-type: none"> <li>■ <code>ctr_sendAndWaitTimer</code></li> <li>■ <code>ctrv_sendAndWaitTimer_current</code></li> <li>■ <code>ctrv_sendAndWaitTimer_occurrences</code></li> <li>■ <code>ctrv_sendAndWaitTimer_persecond</code></li> <li>■ <code>ctrv_sendAndWaitTimer_average</code></li> <li>■ <code>ctrv_sendAndWaitTimer_minimum</code></li> <li>■ <code>ctrv_sendAndWaitTimer_maximum</code></li> </ul>

Counter type	Attributes	Example
	<p>Minimum event value in the moving average window.</p> <ul style="list-style-type: none"> <li>■ <code>ctrv_xxxx_occurrences</code>: Number of events since the counter reset.</li> <li>■ <code>ctrv_xxxx_persecond</code>: Number of events per second in the moving average window.</li> <li>■ <code>ctrv_xxxx_stddev</code>: Standard deviation of the event values in the moving average window.</li> <li>■ <code>ctrv_xxxx_total</code>: Total event value since the counter reset.</li> </ul>	<ul style="list-style-type: none"> <li>■ <code>ctrv_sendAndWaitTimer_stddev</code></li> <li>■ <code>ctrv_sendAndWaitTimer_total</code></li> </ul>

## Operations

In AppWorks Platform, following are the default operations for all the managed components:

- `getDefinedProblems`: Returns all the defined problems on the managed component in the form of a table.
- `getOpenProblems`: Returns all the open problems on the managed component in the form of a table.

## Alerts

Alerts are designed to inform the administrator or the administration application about the major events in the application. For example, failure of the database connection and starting and stopping of the service container are considered as alerts.

Alerts in AppWorks Platform are exposed as JMX notifications, which can be monitored from any JMX management tool. In addition, these alerts are published through Log4J and can be forwarded as SMS or email messages. See [Understanding the alert system](#).

## Problems

*A problem is defined as the state of the component in which the functioning of the component does not occur in the expected manner. The defined problems can be viewed by invoking `getDefinedProblems`. The open problems can be viewed by invoking*

*getOpenProblems from the operations. See [Working with problem registry in the AppWorks Platform Advanced Development Guide](#).*

# Chapter 27

## JConsole

JConsole, which is part of the Java Development Kit (JDK) is a graphical monitoring tool to monitor the Java Virtual Machine (JVM) and Java applications on a local or remote computer.

JConsole uses underlying features of JVM to provide information on performance and resource consumption of applications running on the Java platform. You can use the `jconsole` command to start JConsole.

The JConsole interface consists of six tabs:

- Overview: Provides a graphical view of the data related to JVM such as CPU usage, heap memory usage, threads, and classes.
- Memory: Displays graphical information about memory usage.
- Threads: Displays names and states of the threads used.
- Classes: Displays information about classes such as number of loaded classes and total number of classes unloaded.
- VM Summary: Briefs about the JVM data such as connection name, operating system, and architecture.
- MBeans: Contains the managed components of AppWorks Platform that must be monitored and managed by the administrator. See [MBeans](#).



For information about JConsole, see <http://docs.oracle.com/javase/6/docs/technotes/guides/management/jconsole.html>.

## Connecting to JMX through JConsole

You can connect to JConsole to manage and monitor your applications that implement the Java Management Extensions (JMX) functionality. The management interface of AppWorks Platform is implemented as an abstraction layer for JMX, which supports concepts such as components, settings, counters, operations, alerts, and problems. This topic explains the usage of JConsole to connect to JMX.

The JMX connections are identified using a URL that contains a reference to the Remote Method Invocation (RMI) registry. The RMI registry stores the manageable processes that are registered by JMX. From AppWorks Platform 16 onward, service containers can run under the monitor or in the TomEE application server. This implies that there are two RMI registries in which service containers are registered. The application server where service containers run determine the RMI registry in which the JMX counters are found.

The RMI registry that is hosted inside the monitor, by default, connects to the network port 1099. The port, which is used for the RMI registry hosted in the application server is configured in the application server configuration.

**Note:** Counters for the Web gateway and for those service containers that run in the application server (TomEE) are available in the TomEE RMI registry. In these cases, the JMX connection point specified in TomEE must be used to connect to the TomEE RMI registry. For information about how to deal with JMX specifically in the TomEE application server, see [Managing Apache TomEE](#) and [Enabling JMX Remote in TomEE](#).

## Obtaining the JMX address URL

To connect to JConsole, you require a JMX address URL of the managed component. This depends on whether the service container runs in TomEE or under the Monitor.

If the service container runs in TomEE, the JMX information is available under the `Application_Server` Mbean. You can connect to TomEE using the server name and port number, for example: `srv-pp-prod1:9090`

See the JMX address URL pattern for a specific service container in TomEE.

```
service:jmx:rmi:///jndi/rmi://<TomEE hostname>:<TomEE RMI server
port>/jmxrmi/Application_
Server/<organizationname>%23<servicegroupname>%23<servicecontainername>
```

If the service container runs under the Monitor, you can obtain the JMX address URL in two ways:

### Obtaining the URL from the log files

You can obtain the JMX address URL from the log files, which are present in the `<<AppWorks
Platform_installdir>>/Logs` folder.

After starting or restarting the service container, the log entry for that service container is added to the log file. You can note the JMX address URL from the log entry of the service container.

For example, the following sample shows the first log entry of the WS-Apps service container when started or restarted.

```
<log4j:event logger="com.eibus.management.ManagedComponent"
timestamp="1309772693355" level="INFO" thread="main"> <log4j:message><! [CDATA[JMX
address:
service:jmx:rmi:///jndi/rmi://127.0.0.1:4465/cordys/system%23maxdbservice%23maxdb] >
</log4j:message> <log4j:MDC><! [CDATA[host=CIN10660L processid=12268]]></log4j:MDC>
<log4j:locationInfo method="createJMXProtocolConnectors"
file="ManagedComponent.java" line="1244"/> </log4j:event>
```

From the log entry, you can note the JMX address URL.

```
service:jmx:rmi:///jndi/rmi://127.0.0.1:4465/cordys/system%23maxdbservice%
23maxdb
```

## Forming the URL manually

Form the URL manually using the following format:

```
service:jmx:rmi:///jndi/rmi://<hostName>:<rmiRegPort>/cordys/<ProcessName>
```

where:

<hostName>	Name of the host to connect.
<rmiRegPor t>	Port number of the RMI registry that you can obtain from the <code>wcp.properties</code> file located at: <code>&lt;&lt;AppWorks_Platform_installdir&gt;&gt;/config</code> . The default port number of the RMI registry in the monitor is 1099. If you use a non-default port, you must register it in the <code>wcp.properties</code> file using the <code>com.eibus.management.rmiregistry.port=</code> property.
<ProcessNam e>	Type of process. It can be of three types: <ul style="list-style-type: none"><li>■ Monitor: <code>service:jmx:rmi:///jndi/rmi://srv-ind-dvm19jw:1099/cordys/Monitor</code></li><li>■ OS process: Name of the OS process with the spaces replaced by underscores. For Combined OSProcess1, the URL is: <code>&lt;service:jmx:rmi:///jndi/rmi://srv-ind-dvm19jw:1099/cordys/Combined_OSProcess1</code></li><li>■ Service container: Combination of the organization name in lower case, service group, and the service container with spaces replaced by underscores and the # character replaced by %23. For example, for the Business Process Management service container, the URL is: <code>service:jmx:rmi:///jndi/rmi://127.0.0.1:1099/cordys/system%23business_process_management %23business_process_management</code></li></ul>

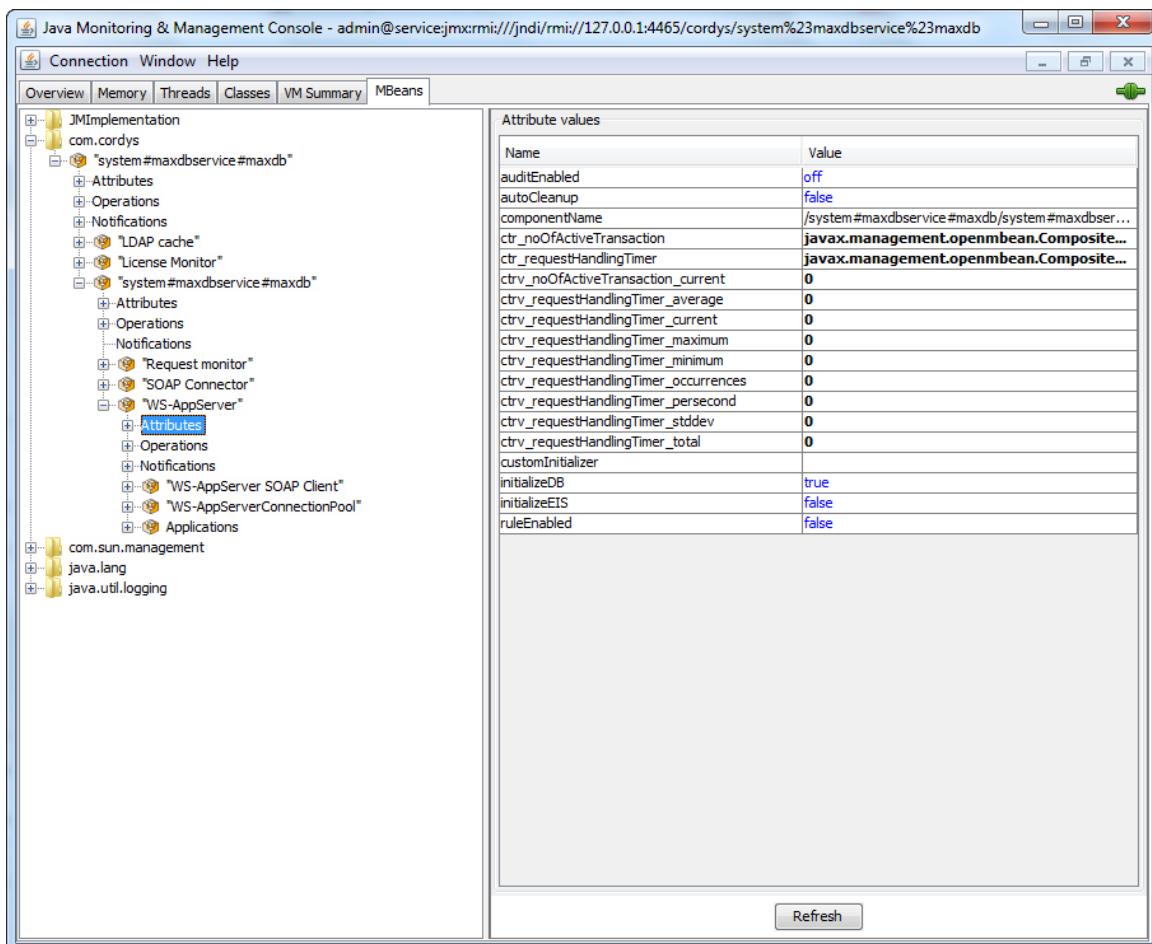
## Connecting to JMX using JConsole

To connect to JMX using JConsole, apart from the JMX address URL, you require a username and password. The user name and password details are specified in the `wcp.properties` file:

```
com.eibus.management.jmxUser=  
com.eibus.management.jmxPassword= <Encoded base64 password>
```

### To connect using JConsole:

1. In the command prompt, type <Your JDK home>/bin/jconsole, and then start the JConsole.  
The JConsole: New Connection window opens.
2. Select **Remote Process**, and then type the JMX address URL and the **Username** and **Password** as configured.
3. Click **Connect**.  
The JConsole interface window opens.
4. Click the **MBeans** tab, and then expand **com.cordys**.  
The managed components of AppWorks Platform are listed.



After connecting to JMX through JConsole, you can monitor and manage the components of AppWorks Platform. See [Monitoring managed components](#).

# Chapter 28

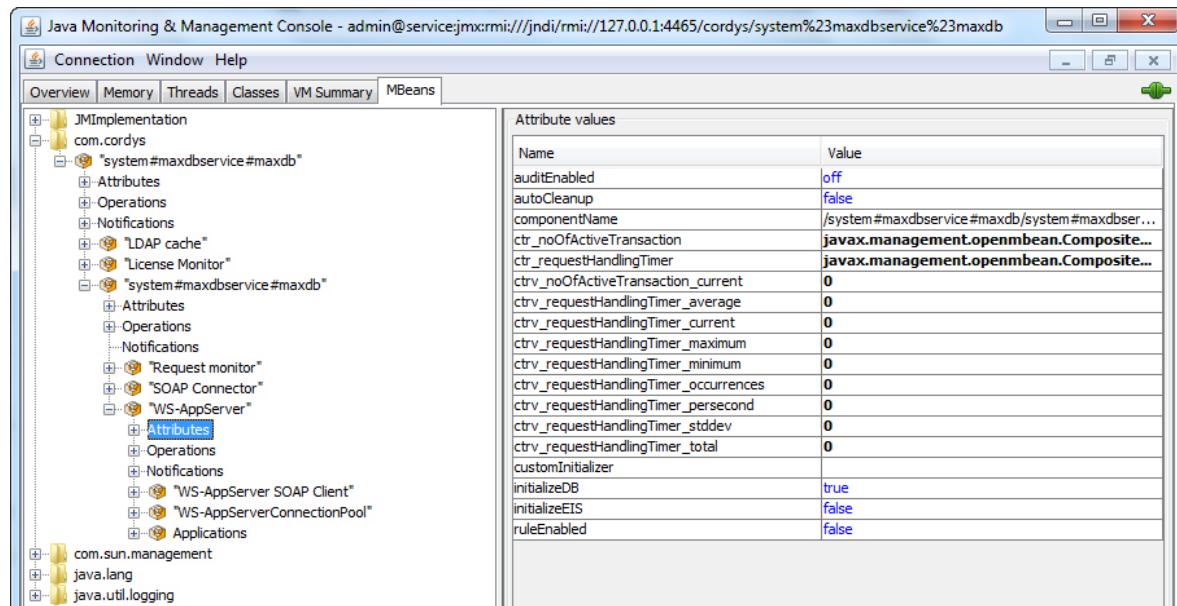
## MBeans

On the MBeans tab, expansion of the `com.cordys` node displays information about the managed components of AppWorks Platform. For each managed component, attributes, operations, notifications and subcomponents can be monitored and managed.

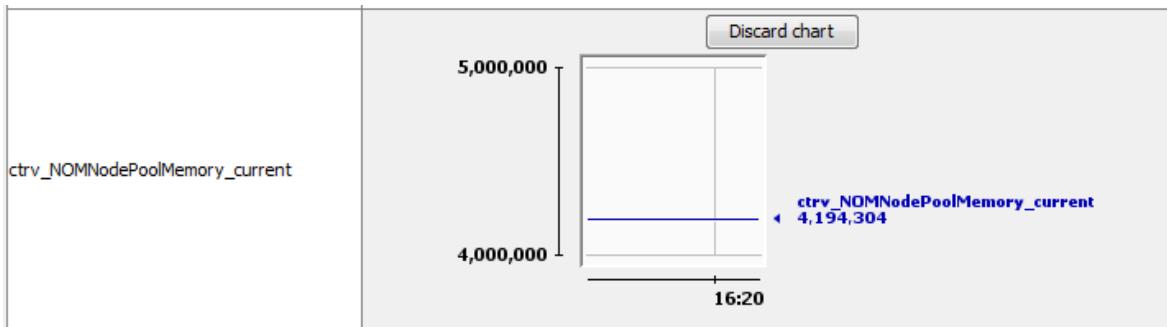
### Attributes

In the context of AppWorks Platform, the settings and counters are exposed as attributes of the managed component.

Counters expose performance data that administrators use to monitor and manage applications. A counter name starts with `ctr`, for example, `ctr_NOMNodePoolMemory`. For information about counters, see [Management interface](#).

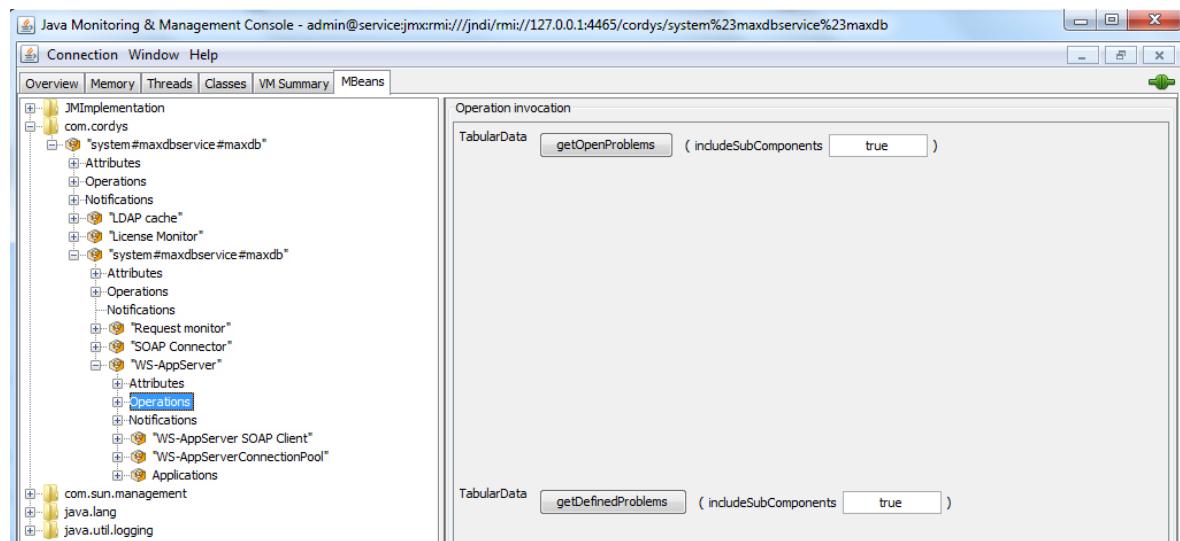


You can double-click any attribute to view the chart that displays the attribute value at a specific time period. For example, see the graphical view of the counter `ctrv_NOMNodePoolMemory_current`.



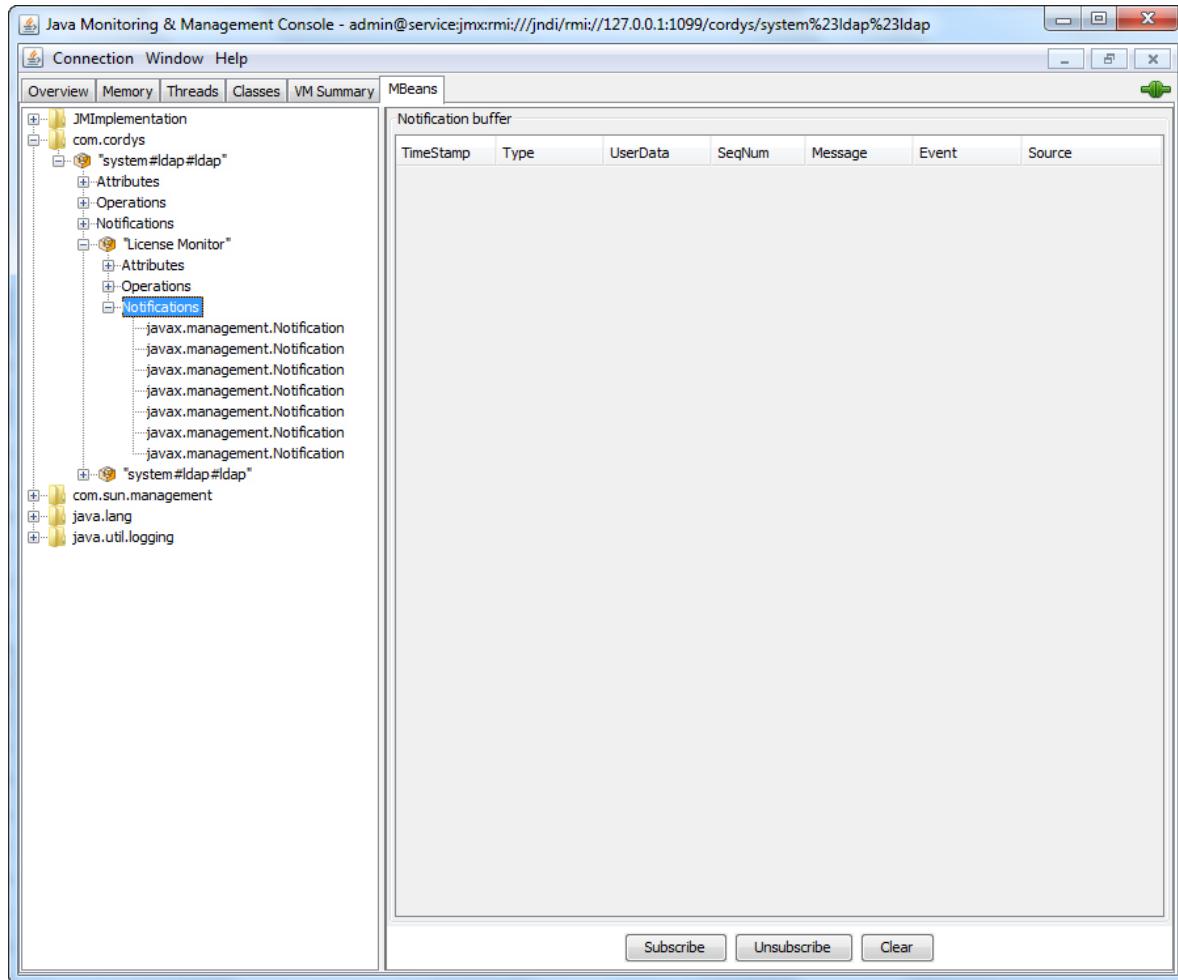
## Operations

The Operations node displays the list of operations to perform in MBeans. The operations display as buttons in MBeans that are invoked when clicked.



## Notifications

Alerts in AppWorks Platform are exposed as JMX notifications in MBeans. The administrator must click **Subscribe** to view notifications or **Unsubscribe** to hide notifications.



# Chapter 29

## Managed components

A managed component is exposed as a JMX Managed Bean (MBean), which represents a management-instrumented resource.

Managed components can have subcomponents, settings, operations, performance counters, problems, and alerts. Through the management subsystem, application developers can perform the following:

- Create managed components.
- Define performance counters.
- Define settings.
- Define and issue alerts.
- Define and raise problems.
- Define operations on the managed components.

In AppWorks Platform, every Operating System (OS) process is associated with a managed component that contains subcomponents for the resources such as LDAP cache and service container. As an application developer, you can create subcomponents in existing components and define performance counters, operations, settings, alerts, and problems on a managed component.

**Note:** The managed component name must not contain the reserved character '/' (forward slash). The managed component type must be a legal Java identifier. According to the Java naming convention, managed component types must begin with an uppercase letter.

See the basic initialization code to define a managed process.

```
ManagedProcess.setProcessName ("process_name");
IManagedComponent m_managedComponent = ManagedProcess.createManagedComponent
("OSProcess", Messages.ROOT_COMPONENT_DESC);
```

After initializing the complete tree of subcomponents, register the tree of managed components.

```
//registration of the component tree
m_managedComponent.registerComponentTree ();
```

For information about defining managed components, see:

- [Defining a managed component in an application connector](#)
- [Defining a managed component in a Web application](#)
- [Creating subcomponents](#)
- [Creating WS-AppServer application components](#)

## Managed component types

MBeans exposes the managed components of AppWorks Platform that an administrator must monitor and manage. The managed components of AppWorks Platform are described below in a logical manner. For each managed component, the corresponding settings, counters, operations, alerts, problems, and subcomponents are explained.

### Before you begin:

- Ensure that a connection with JMX is established through the JConsole. See [Connecting to JMX through JConsole](#).
- Identify the task to perform, for example, changing the settings, invoking the operations, and monitoring the counters.

Managed component	Subcomponent
Monitor	<ul style="list-style-type: none"><li>■ <a href="#">ISVP connector (Managed components)</a></li><li>■ <a href="#">Request monitor (Managed components)</a></li><li>■ <a href="#">SOAP connector (Managed components)</a></li><li>■ <a href="#">Middleware wrapper (Managed components)</a></li></ul>
Gateway	<ul style="list-style-type: none"><li>■ <a href="#">Bus gateway (Managed components)</a></li><li>■ <a href="#">Web application (Managed components)</a><ul style="list-style-type: none"><li>• <a href="#">MessageGateway</a></li><li>• <a href="#">LogMeIn</a></li><li>• <a href="#">PreLogInfo</a></li><li>• <a href="#">XGateway</a></li></ul></li><li>■ <a href="#">WSDLGateway Web application (Managed components)</a></li><li>■ <a href="#">Download gateway (Managed components)</a></li><li>■ <a href="#">Upload gateway (Managed components)</a></li><li>■ <a href="#">Gateway ThreadPool (Managed components)</a></li></ul>

<b>Managed component</b>	<b>Subcomponent</b>
OS Process	<ul style="list-style-type: none"> <li>■ <a href="#">LDAP cache (Managed components)</a></li> <li>■ <a href="#">License monitor (Managed components)</a></li> </ul>
Business Activity Monitor	<ul style="list-style-type: none"> <li>■ <a href="#">Business Activity Monitoring service (Managed components)</a></li> <li>■ <a href="#">Monitoring engine (Managed components)</a></li> <li>■ Monitoring Engine Connection Pool</li> <li>■ <a href="#">Scheduler engine (Managed components)</a></li> <li>■ <a href="#">Rule engine (Managed components)</a></li> </ul>
Business Process Management	<ul style="list-style-type: none"> <li>■ <a href="#">Business Process Engine (Managed components)</a></li> <li>■ <a href="#">Case Management Engine (Managed components)</a></li> <li>■ <a href="#">Embedded data mapper</a></li> <li>■ <a href="#">DBConnectionPool (Managed components)</a> <ul style="list-style-type: none"> <li>• <a href="#">Process Engine Database Connection Pool</a></li> </ul> </li> <li>■ <a href="#">Scheduler DBConnectionPool</a></li> <li>■ <a href="#">PIM query connector connection pool</a></li> </ul>
CAP Connector	<ul style="list-style-type: none"> <li>■ <a href="#">Deployment (Managed components)</a></li> <li>■ <a href="#">WebServiceOperations (Managed components)</a></li> <li>■ <a href="#">XDSRepository (Managed components)</a></li> </ul>
CoBOC	<ul style="list-style-type: none"> <li>■ <a href="#">Business object cache</a></li> <li>■ <a href="#">DBConnectionPool (Managed components)</a></li> <li>■ <a href="#">Rule engine (Managed components)</a></li> <li>■ Rule Repository ConnectionPool</li> <li>■ Scheduler DBConnectionPool</li> </ul>
Collaborative Workspace	<ul style="list-style-type: none"> <li>■ <a href="#">XDSRepository (Managed components)</a></li> </ul>
Data transformation	<ul style="list-style-type: none"> <li>■ <a href="#">Business object cache</a></li> <li>■ CoBOC DBConnectionPool</li> </ul>

Managed component	Subcomponent
Document store connector	<ul style="list-style-type: none"> <li>▪ Subcomponent for External DMS</li> </ul>
E-Mail	No subcomponents
Event handling	No subcomponents
FTP	No subcomponents
HTTP connector	No subcomponents
LDAP	<ul style="list-style-type: none"> <li>▪ <a href="#">LDAP cache (Managed components)</a></li> <li>▪ <a href="#">LDAPConnection (Managed components)</a></li> </ul>
Logging	No subcomponents
MDM hub publisher	<ul style="list-style-type: none"> <li>▪ Application</li> <li>▪ Repository</li> </ul>
Notification	<ul style="list-style-type: none"> <li>▪ Notification Service Connection Pool</li> <li>▪ <a href="#">Notification dispatcher</a></li> <li>▪ <a href="#">TaskIdentifierIndexer</a></li> </ul>
Rule management	<ul style="list-style-type: none"> <li>▪ <a href="#">Rule engine (Managed components)</a></li> <li>▪ Rule Repository ConnectionPool</li> </ul>
Scheduling	<ul style="list-style-type: none"> <li>▪ Scheduler DBConnectionPool</li> <li>▪ <a href="#">Scheduler engine (Managed components)</a></li> </ul>
Security administration	No subcomponents
Single sign-On	<ul style="list-style-type: none"> <li>▪ <a href="#">AppWorks Platform SAML protocol handler (Managed components)</a></li> </ul>
UDDI service	No subcomponents
Ws-AppServer	<ul style="list-style-type: none"> <li>▪ <a href="#">WS-AppServer SOAP Client</a></li> <li>▪ WS-AppServerConnectionPool</li> </ul>
XForms	No subcomponents

## Creating subcomponents

You can create subcomponents to the existing managed components. For example, the WS-AppServer application connector internally uses a `DBConnectionPool` object that handles database connection pooling. In this scenario, you can add a subcomponent to the managed component of the WS-AppServer application connector. The `DBConnectionPool` object can define settings and operations on this subcomponent. The component associated with the

`DBConnectionPool` object is called a subcomponent of the application connector (the parent component).

**Note:** Dynamic creation of subcomponents is also possible. When creating a subcomponent after registering the component tree through `registerComponentTree()`, the subcomponent is not automatically visible. In such cases, you must call `registerComponentTree()` explicitly after dynamically adding the subcomponent.

See the sample code.

```
// Dynamically create a subcomponent of s_managedComponent
IManagedComponent webAppManagedComponent = webApplication.createManagedComponent(s_
managedComponent);

// Explicitly register the newly created component (tree)
webAppManagedComponent.registerComponentTree();

s_webApplications.put(className, webApplication);
```

## Creating WS-AppServer application components

WS-AppServer facilitates creation of managed components under its default subcomponent, Applications. The JMX functionality in WS-AppServer helps system administrators remotely monitor the WS-AppServer SOAP configuration and various managed components of different WS-AppServer-based applications.

The following Web service operation creates the managed component in the `BsfConnector` class.

```
/**
 * Creates a nested managed application subcomponent inside
 * 'Applications' subcomponent
 * @param type The type of subcomponent, for example,
 * dbConnectionPool, soapConnector, and ldapCache
 * @param description The description of the subcomponent
 * @param componentImpl The object actually implementing the
 * subcomponent
 *
 * @return The created managed application subcomponent object
 */

public IManagedComponent createManagedApplicationComponent(String type,
ILocalizableString description, Object componentImpl)
```

The subcomponents are registered internally, which are handled by WS-AppServer. See the sample code to register the WS-AppServer application subcomponent. In this example:

- `sales` is a type of component and represents the application.
- `localizedMessage` is an object implementing the `ILocalizableString` interface.

- object is the object implementing the subcomponent.

```

import com.cordys.cpc.bsf.connector.BsfConnector;
import com.cordys.cpc.bsf.connector.IOnBsfConnector;

public class Initializer implements IOnBsfConnector
{
    public void onOpenConnector(BsfConnector connector)
    {
        //Application specific code
        //To create a managed component for the application
        connector.createManagedApplicationComponent
        ("sales",Messages.APPLICATION_DESC, object)
    }
}

```

## Defining a managed component in an application connector

A managed component is associated with a generic application connector. Application connector developers can use this managed component to define counters, settings, and subcomponents.

To add counters and settings to a specific application connector, overwrite the Web service operation `createManagedComponent()`.

```

/**
 * Create a managed component for this application connector
 * using the processor as the parent component.
 */
protected IManagedComponent createManagedComponent()
{
    IManagedComponent mc = super.createManagedComponent();
    timeoutCounter = (ITimerEventValueCounter) mc.createPerformanceCounter
    ("timeouts",Messages.REQUEST_TIMEOUT_DESC,CounterFactory.TIMER_EVENT_VALUE_COUNTER);
    return mc;
}

```

The service initially calls the `createManagedComponent()` Web service operation, and then calls the `open()` Web service operation of the application connector.

You can define counters and settings in the following Web service operations:

- `createManagedComponent()` after creating the managed component using `super.createManagedComponent()`
- `open()`

When an application connector adds such elements to the managed component defined in the base class, it must also overwrite the `getManagedComponentType()` Web service operation and return a specific type. This is necessary because, by convention, a managed component type has a fixed set of counters, settings, and subcomponents. It is also useful to overwrite the `getManagementDescription()` and `getManagementName()` Web service operations to provide useful descriptions for this specific application connector.

To retrieve the managed component type, and name and description of the application connector, overwrite the Web service operations `getManagedComponentType()`, `getManagementDescription()`, and `getManagementName()`.

```
/**
 * Returns the type of this managed component.
 *
 * @return the type of this application connector.
 */
protected String getManagedComponentType()
{
    return "RequestMonitor";
}

/**
 * Returns the name of this application connector,
 * suitable for usage in a management tool.
 *
 * @return the name of this application connector,
 * suitable for usage in a management tool.
 */
protected String getManagementName()
{
    return "Request monitor";
}

/**
 * Returns the description of this application connector,
 * suitable for usage in a management tool.
 *
 * @return the name of this application connector,
 * suitable for usage in a management tool.
 */
protected String getManagementDescription()
{
    return Messages.REQUEST_MONITOR_CONNECTOR_DESCRIPTION;
}
```

## Defining a managed component in a Web application

A managed component is associated with a generic Web application. Developers writing the Web applications can use this managed component to define counters, operations, and sub-components.

To add performance counters in a specific Web application, overwrite the Web service operation `setupManagedComponent()`.

```
public class TestWebApplication extends WebApplication
{
    protected void setupManagedComponent (IManagedComponent managedComponent)
    {
        super. setupManagedComponent (managedComponent);
        managedComponent.createPropertyBasedValueCounter
        ("numOfActiveSessions",Messages.ACTIVE_SESSIONS_
DESC,"numberOfActiveSessions",Session.class);
    }
}
```

**Note:** To create performance counters of the base classes, call the `setupManagedComponent` super class.

## AppConnector (Managed components)

The AppConnector managed component type represents the base type for all application connectors. Specialized application connectors inherit all settings, counters and subcomponents.

### Settings

None

### Counters

#### requestHandlingTimer

<b>Counter type</b>	Timer event value counter
<b>Event</b>	Frequency of the request handling of this application connector
<b>Value</b>	The duration of the request handling
<b>Hints</b>	None

### Problems

None

### Subcomponents

None

# Auditing application connector (Managed components)

## Settings

None

## Counters

### **requestHandlingTimer**

<b>Counter type</b>	Event value counter
<b>Event</b>	Frequency of the request handling of this application connector
<b>Hints</b>	None

## Alerts

None

## Subcomponents

None

# BAM business object dispatcher (Managed components)

## Settings

### **maxConcurrentWorkers**

<b>Setting type</b>	Hot
<b>Data type</b>	Integer
<b>Description</b>	Maximum number of concurrent threads in the thread pool

### ***monitorPeriod***

<b>Setting type</b>	Hot
<b>Data type</b>	Long
<b>Description</b>	Time interval at which the dispatcher checks the number of threads that are in the idle state

### ***minConcurrentWorkers***

<b>Setting type</b>	Hot
<b>Data type</b>	Integer
<b>Description</b>	Minimum number of concurrent threads in the thread pool

## **Counters**

### ***currentNumWorkers***

<b>Counter type</b>	Event counter
<b>Event</b>	Total number of worker threads
<b>Hints</b>	None

### ***numIdleWorkers***

<b>Counter type</b>	Event counter
<b>Event</b>	Number of idle worker threads
<b>Hints</b>	None

### ***numActiveWorkers***

<b>Counter type</b>	Event counter
<b>Event</b>	Number of active worker threads
<b>Hints</b>	None

## **Alerts**

None

# BAM configuration (Managed components)

## Settings

### *allowQueryLanguageFallback*

<b>Setting type</b>	Hot
<b>Readable</b>	Yes
<b>Type</b>	Boolean
<b>Description</b>	Value for allowing query language fallback

### *connectionPoolRefreshInterval*

<b>Setting type</b>	Hot
<b>Readable</b>	Yes
<b>Type</b>	Integer
<b>Description</b>	Value for connection pool refresh interval

### *conversionConfig*

<b>Setting type</b>	Hot
<b>Readable</b>	Yes
<b>Type</b>	String
<b>Description</b>	Configuration details

### *cursorCacheRefreshInterval*

<b>Setting type</b>	Hot
<b>Readable</b>	Yes
<b>Type</b>	Integer
<b>Description</b>	Cursor cache refresh interval

### *cursorCacheSize*

<b>Setting type</b>	Hot
---------------------	-----

<b>Readable</b>	Yes
<b>Type</b>	Integer
<b>Description</b>	Cursor cache size

### ***datasource***

<b>Setting type</b>	Hot
<b>Readable</b>	Yes
<b>Type</b>	String
<b>Description</b>	Data source configuration

### ***defaultCursorSize***

<b>Setting type</b>	Hot
<b>Readable</b>	Yes
<b>Type</b>	Long
<b>Description</b>	Default cursor size

### ***maximumReadConnections***

<b>Setting type</b>	Hot
<b>Readable</b>	Yes
<b>Type</b>	Integer
<b>Description</b>	Value for maximum read connections

### ***maximumWriteConnections***

<b>Setting type</b>	Hot
<b>Readable</b>	Yes
<b>Type</b>	Integer
<b>Description</b>	Value for maximum write connections

### ***minimumReadConnections***

<b>Setting type</b>	Hot
---------------------	-----

<b>Readable</b>	Yes
<b>Type</b>	Integer
<b>Description</b>	Value for minimum read connections

### ***minimumWriteConnections***

<b>Setting type</b>	Hot
<b>Readable</b>	Yes
<b>Type</b>	Integer
<b>Description</b>	Value for minimum write connections

### ***multibyte***

<b>Setting type</b>	Hot
<b>Readable</b>	Yes
<b>Type</b>	Boolean
<b>Description</b>	Whether multibyte support is required

### ***multithreaded***

<b>Setting type</b>	Hot
<b>Readable</b>	Yes
<b>Type</b>	Boolean
<b>Description</b>	Whether the value is multithreaded

### ***optimiseQueryForCursor***

<b>Setting type</b>	Hot
<b>Readable</b>	Yes
<b>Type</b>	Boolean
<b>Description</b>	Value for optimize query for cursor

### ***precedence***

<b>Setting type</b>	Hot
---------------------	-----

<b>Readable</b>	Yes
<b>Type</b>	Boolean
<b>Description</b>	Value for precedence

### ***queryCacheRefreshInterval***

<b>Setting type</b>	Hot
<b>Readable</b>	Yes
<b>Type</b>	Integer
<b>Description</b>	Value for query cache refresh interval

### ***queryCacheSize***

<b>Setting type</b>	Hot
<b>Readable</b>	Yes
<b>Type</b>	Integer
<b>Description</b>	Value for query cache size

### ***reconnectionAttempts***

<b>Setting type</b>	Hot
<b>Readable</b>	Yes
<b>Type</b>	Integer
<b>Description</b>	Value for reconnection attempts

### ***reconnectionInterval***

<b>Setting type</b>	Hot
<b>Readable</b>	Yes
<b>Type</b>	Integer
<b>Description</b>	Value for interval reconnection

### ***requestThreshold***

<b>Setting type</b>	Hot
---------------------	-----

<b>Readable</b>	Yes
<b>Type</b>	Integer
<b>Description</b>	Value for threshold

### *xmlEncoding*

<b>Setting type</b>	Hot
<b>Readable</b>	Yes
<b>Type</b>	Boolean
<b>Description</b>	Whether XML encoding is required

### *xsiNilForNullData*

<b>Setting type</b>	Hot
<b>Readable</b>	Yes
<b>Type</b>	Boolean
<b>Description</b>	Value for xsiNilForNullData

## Counters

### *numOfCachedReadConnections*

<b>Counter type</b>	Value counter
<b>Event</b>	Number of currently cached read connections
<b>Hints</b>	None

### *numOfCachedWriteConnections*

<b>Counter type</b>	Value counter
<b>Event</b>	Number of currently cached write connections
<b>Hints</b>	None

### *numberOfCachedQueries*

<b>Counter type</b>	Value counter
<b>Event</b>	Number of currently cached queries in all connections
<b>Hints</b>	None

***numberOfOpenCursors***

<b>Counter type</b>	Value counter
<b>Event</b>	Number of currently open cursors in all connections
<b>Hints</b>	None

***readConnectionUsageDuration***

<b>Counter type</b>	Timer event value counter
<b>Event</b>	Frequency and duration of usage of a read connection
<b>Hints</b>	None

***readConnectionWait***

<b>Counter type</b>	Timer event value counter
<b>Event</b>	Frequency and duration of wait actions for a read connection
<b>Hints</b>	None

***writeConnectionUsageDuration***

<b>Counter type</b>	Timer event value counter
<b>Event</b>	Frequency and duration of usage of a write connection
<b>Hints</b>	None

***writeConnectionWait***

<b>Counter type</b>	Timer event value counter
<b>Event</b>	Frequency and duration of wait actions for a write connection
<b>Hints</b>	None

**Alerts*****databaseConnectionFailure***

<b>Severity level</b>	ERROR
<b>Message</b>	This alert is issued when connection to database server is lost.

<b>Description</b>	This alert is issued when connection to database server is lost.
--------------------	------------------------------------------------------------------

### ***databaseConnectionEstablished***

<b>Severity level</b>	INFO
<b>Message</b>	This alert is issued when connection to database server is successful.
<b>Description</b>	This alert is issued when connection to database server is successful.

## **Problems**

None

## **Subcomponents**

None

# **BAM event dispatcher (Managed components)**

## **Settings**

### ***maxConcurrentWorkers***

<b>Setting type</b>	Hot
<b>Data type</b>	Integer
<b>Description</b>	Maximum number of concurrent threads in the thread pool

### ***monitorPeriod***

<b>Setting type</b>	Hot
<b>Data type</b>	Long
<b>Description</b>	Time interval at which the Dispatcher checks how many threads are in the idle state

### ***minConcurrentWorkers***

<b>Setting type</b>	Hot
<b>Data type</b>	Integer
<b>Description</b>	Minimum number of concurrent threads in the thread pool

## **Counters**

### ***currentNumWorkers***

<b>Counter type</b>	Event counter
<b>Event</b>	Total number of worker threads
<b>Hints</b>	None

### ***numIdleWorkers***

<b>Counter type</b>	Event counter
<b>Event</b>	Number of worker threads in the idle state
<b>Hints</b>	None

### ***numActiveWorkers***

<b>Counter type</b>	Event counter
<b>Event</b>	Number of worker threads that are active
<b>Hints</b>	None

## **Alerts**

None

## **Branch cache**

### **Settings**

#### **cacheSize**

<b>Setting type</b>	Hot
---------------------	-----

<b>Type</b>	Long
<b>Description</b>	Branch cache size

## Counters

### noOfCacheHits

<b>Counter type</b>	Event Occurrence counter
<b>Value</b>	Number of branch cache hits
<b>Hints</b>	None

### noOfCacheMisses

<b>Counter type</b>	Event Occurrence counter
<b>Event</b>	Number of branch cache misses
<b>Hints</b>	None

### noOfObjectsInCache

<b>Counter type</b>	Value counter
<b>Event</b>	Number of objects in the branch cache
<b>Hints</b>	None

## Alerts

None

## Operations

### resetCache

<b>Description</b>	Clears the XDS branch cache
<b>Parameters</b>	None
<b>Return value</b>	String

## Subcomponents

None

# Bus gateway (Managed components)

## Settings

None

## Counters

### *numBusErrors*

<b>Counter type</b>	Event occurrence counter
<b>Event</b>	Frequency of number of errors
<b>Hints</b>	None
<b>Value</b>	Tracks frequency of errors

### *numInvalidMessages*

<b>Counter type</b>	Event occurrence counter
<b>Event</b>	Frequency of invalid requests
<b>Hints</b>	None
<b>Value</b>	Tracks frequency of invalid requests

### *numLDAPErrors*

<b>Counter type</b>	Event occurrence counter
<b>Event</b>	Frequency of number of LDAP errors
<b>Hints</b>	None
<b>Value</b>	Tracks frequency of LDAP errors

### *numMessagesReceived*

<b>Counter type</b>	Event occurrence counter
<b>Event</b>	Frequency of number of replies from the Integrator
<b>Hints</b>	None
<b>Value</b>	Tracks frequency of replies from AppWorks Platform

### ***numTimeoutErrors***

<b>Counter type</b>	Event occurrence counter
<b>Event</b>	Frequency of number of timeout errors
<b>Hints</b>	None
<b>Value</b>	Tracks frequency of timeout errors

### ***numUnknownAuthenticatedUsers***

<b>Counter type</b>	Event occurrence counter
<b>Event</b>	Frequency of requests with unknown authentication users
<b>Hints</b>	None
<b>Value</b>	Tracks frequency of requests with unknown authentication users.

### ***numUnknownOrganizationalUsers***

<b>Counter type</b>	Event occurrence counter
<b>Event</b>	Frequency of requests with unknown organizational users
<b>Hints</b>	None
<b>Value</b>	Tracks frequency of requests with unknown organizational users

### ***processingTime***

<b>Counter type</b>	Value counter
<b>Event</b>	Frequency and duration of request handling processing of the Bus Gateway
<b>Hints</b>	None
<b>Value</b>	Tracks frequency and duration of request handling processing of the Bus Gateway

### ***waitTime***

<b>Counter type</b>	Value counter
<b>Event</b>	Duration of request handling wait time of the Bus Gateway

<b>Hints</b>	None
<b>Value</b>	Tracks duration of request handling wait time of the Bus Gateway

## Alerts

None

## Problems

None

## Subcomponents

The Bus Gateway has one sub-component type: [Middleware wrapper \(Managed components\)](#).

# Business Activity Monitoring service (Managed components)

## Alerts

### stateSyncUpFailure

<b>Severity level</b>	Error.
<b>Message</b>	The State SyncUp (SSU) initialization has failed in the BAM service. SSU is required for maintaining the cache coherence across the BAM service containers.
<b>Description</b>	This alert is issued when the application syncup fails.

### ruleEngineInitializationFailure

<b>Severity level</b>	Error.
<b>Message</b>	The embedded monitoring rule engine in the BAM service is not initialized.
<b>Description</b>	This alert is issued when the embedded monitoring rule engine initialization fails.

**scheduleEngineInitializationFailure**

<b>Severity level</b>	Error.
<b>Message</b>	The embedded monitoring schedule engine in the BAM service is not initialized.
<b>Description</b>	This alert is issued when the embedded monitoring schedule engine initialization fails.

**monitoringEngineStopped**

<b>Severity level</b>	Info.
<b>Message</b>	BAM Monitoring engine has stopped.
<b>Description</b>	This alert is issued when the running BAM Monitoring engine stops.

## Problems

**databaseConnectionFailure**

<b>problemRaiseDescription</b>	The database server is unavailable. The database server may not be functioning, or the network may be encountering server connectivity problems. Ensure that the database server is functioning, and the network is communicating properly with the server.
<b>problemClearDescription</b>	Connection to the database server has been established.

# Business object cache

## Settings

**hashbinsCapacity**

<b>Setting type</b>	Hot
<b>Type</b>	Integer
<b>Description</b>	Capacity of hash bin entries

**numCoBOCHashbins**

<b>Setting type</b>	Hot
<b>Type</b>	Integer
<b>Description</b>	Number of CoBOC hash bin entries

**Counters****noOfObjectsInCache**

<b>Counter type</b>	Event value counter
<b>Event</b>	Number of objects in the cache
<b>Hints</b>	None

**Operations****ClearCobocCache**

<b>Description</b>	Clears the CoBOC cache
<b>Parameters</b>	None
<b>Return value</b>	None

**Alerts**

None

**Problems**

None

**Subcomponents**

None

# Business Process Engine (Managed components)

## Settings

### **activateDebugPoints**

<b>Setting type</b>	Hot
<b>Type</b>	Boolean
<b>Description</b>	Activate debug points

### **adminEnabled**

<b>Setting type</b>	Hot
<b>Type</b>	Boolean
<b>Description</b>	Admin-enabled

### **agingTime**

<b>Setting type</b>	Hot
<b>Type</b>	Boolean
<b>Description</b>	Aging time

### **binaryParserEnabled**

<b>Setting type</b>	Hot
<b>Type</b>	Integer
<b>Description</b>	Binary parser-enabled

### **clientConnectionPoint**

<b>Setting type</b>	Hot
<b>Type</b>	String
<b>Description</b>	Client connection point

### **crashRecovery**

<b>Setting type</b>	Hot
---------------------	-----

<b>Type</b>	Boolean
<b>Description</b>	Crash recovery

**dataMapperEnabled**

<b>Setting type</b>	Hot
<b>Type</b>	Boolean
<b>Description</b>	Data mapper-enabled

**enableAging**

<b>Setting type</b>	Hot
<b>Type</b>	Boolean
<b>Description</b>	Enable aging

**notificationService**

<b>Setting type</b>	Hot
<b>Readable</b>	Yes
<b>Type</b>	String
<b>Description</b>	Notification service

**priorityAlgorithm**

<b>Setting type</b>	Hot
<b>Type</b>	String
<b>Description</b>	Priority algorithm

**priorityEnabled**

<b>Setting type</b>	Hot
<b>Type</b>	Boolean
<b>Description</b>	Priority-enabled

**ruleEngineEnabled**

<b>Setting type</b>	Hot
---------------------	-----

<b>Type</b>	Boolean
<b>Description</b>	Rule engine-enabled

**threadPoolSize**

<b>Setting type</b>	Hot
<b>Type</b>	Integer
<b>Description</b>	Thread pool size

**webServiceExecutionDurationWarningThreshold**

<b>Setting type</b>	Hot
<b>Type</b>	Integer
<b>Description</b>	Web service execution duration warning threshold

## Operations

**resetProcessStatusCounters**

<b>Description</b>	This operation resets each of the process status counters: numAbortedProcesses, numCompletedProcesses, and numTerminatedProcesses.
<b>Parameters</b>	None
<b>Return value</b>	None

## Counters

**numAbortedProcesses**

<b>Counter type</b>	Value counter
<b>Value</b>	Number of business processes aborted because of unhandled errors since the last reset or restart of the processor
<b>Hints</b>	None

**numCompletedProcesses**

<b>Counter type</b>	Value counter
<b>Value</b>	Number of business processes completed since the last reset or restart of the processor

<b>Hints</b>	None
--------------	------

**numTerminatedProcesses**

<b>Counter type</b>	Value counter
<b>Value</b>	Number of business processes terminated by the administrator since the last reset or restart of the processor
<b>Hints</b>	None

Only macro processes use the thread pool of the process engine. Therefore, the following counters are thread pool counters that are applicable only to macro processors: numActiveThreads, numFreeThreads, and threadPoolSize.

**numActiveThreads**

<b>Counter type</b>	Value counter
<b>Value</b>	Number of threads in the thread pool of the process engine that are engaged in executing the business process
<b>Hints</b>	None

**numFreeThreads**

<b>Counter type</b>	Value counter
<b>Value</b>	Number of threads in the thread pool of the process engine that are available for accepting new business processes for execution
<b>Hints</b>	None

**threadPoolSize**

<b>Counter type</b>	Value counter
<b>Value</b>	Total number of threads available in the thread pool of the process engine
<b>Hints</b>	None

**queueSize**

Only macro processes use the queue of the process engine. Therefore, this process engine counter is applicable only to macro processors.

<b>Counter type</b>	Value counter
---------------------	---------------

<b>Value</b>	Process engine queue size
<b>Hints</b>	A request to execute a macro process is queued in the process engine when no thread in the process engine thread pool is available to execute it.

**numCacheSize**

<b>Counter type</b>	Value counter
<b>Value</b>	Cache size
<b>Hints</b>	None

**numCacheMisses**

<b>Counter type</b>	Value counter
<b>Value</b>	Number of cache misses
<b>Hints</b>	None

**waitingRequestsQueueSize**

<b>Counter type</b>	Value counter
<b>Value</b>	Queue size of the waiting requests
<b>Hints</b>	None

**numCacheHits**

<b>Counter type</b>	Value counter
<b>Value</b>	Number of cache hits
<b>Hints</b>	None

**Alerts**

None

**Problems**

None

**Subcomponents**

None

# Business Process Management processor

## Settings

None

## Counters

### requestHandlingTimer

<b>Counter type</b>	Event value counter
<b>Event</b>	Frequency of request handling of this application connector
<b>Hints</b>	None

## Alerts

### dbConnectionFailure

<b>Severity level</b>	Error.
<b>Message</b>	The database server may not be functioning, or the network may be encountering server connectivity problems.
<b>Description</b>	This alert is issued when processor cannot be started because of database connectivity problem.

### unableToResolveDso

<b>Severity level</b>	Error.
<b>Message</b>	The process engine was not able to resolve the database configuration from the specified information in the business process model (transactional process: {0} for instance id: {1}.)
<b>Description</b>	This alert is issued because of issues in resolving database configuration for transactional process.

### stateSyncUpFailure

<b>Severity level</b>	Error.
<b>Message</b>	State sync up (SSU) is not configured. State sync up is required to synchronization among multiple BPM Engines

	configured for load balancing.
<b>Description</b>	This alert is issued because of State Synch up configuration problems.

**notificationSOAPProcessorFailure**

<b>Severity level</b>	Error.
<b>Message</b>	The system was not able to deliver the notification from the process engine. The Notification SOAP processor is not functioning properly, or has not yet been started.
<b>Description</b>	This alert is issued because of Notification Soap Processor Failure.

**ruleEngineInitializationFailure**

<b>Severity level</b>	Error.
<b>Message</b>	Embedded Rule Engine could not be initialized. Check the configuration details in the SOAP Processor Configuration.
<b>Description</b>	This alert is issued because Embedded Rule Engine could not be initialized.

**reachedMaxIterationCountDesc**

<b>Severity level</b>	Error.
<b>Message</b>	The execution of business process-instance with instance-id {0} of business process model {1} running in organization {2} and started by user {3} is suspended by the system as the current iteration count {4} for the loop activity {5} with activity-type {6} reached the threshold limit {7} for execution of maximum number of iterations in a single run set by the administrator. To continue execution, the business process instance is required to be resumed manually by the administrator.
<b>Description</b>	The execution of business process-instance with instance-id {0} of business process model {1} running in organization {2} and started by user {3} is suspended by the system as the current iteration count {4} for the loop activity {5} with activity-type {6} reached the threshold limit {7} for execution of maximum number of iterations in a single run set by the administrator. To continue execution, the business process instance is required to be resumed manually by the administrator.

**archiveFileZippingFailure**

<b>Severity level</b>	Error.
<b>Message</b>	The archive file could not be created successfully. Please create a zip of the folder at {0} with the same folder name.
<b>Description</b>	This alert is issued because the archive file could not be created successfully due to zipping failure.

**reachedMaxAllowedInstanceCountDesc**

<b>Severity level</b>	Error.
<b>Message</b>	The current rate of creation of new process-instances {0} for the business process model {1} (deployed to model-space {2}) exceeded the threshold limit {3} process-instances per {4} second(s) set by the administrator. The request is sent from the organization {5} by user {6}. Try after some time or contact administrator.
<b>Description</b>	The current rate of creation of new process-instances {0} for the business process model {1} (deployed to model-space {2}) exceeded the threshold limit {3} process-instances per {4} second(s) set by the administrator. The request is sent from the organization {5} by user {6}. Try after some time or contact administrator.

## Problems

**databaseConnectionFailure**

<b>problemRaiseDescription</b>	The database server is not available. The database server may not be functioning, or the network may be encountering server connectivity problems. Ensure that the database server is functioning, and the network is communicating properly with the server.
<b>problemClearDescription</b>	Connection to the database server has been established.

## Subcomponents

The Business Process Management Processor has five subcomponents: Business Process Engine, Case Management Engine, Embedded Data Mapper, Process Engine Database Connection Pool, and Scheduler DBConnectionPool.

# CAP connector (Managed components)

## Settings

None

## Counters

### **requestHandlingTimer**

<b>Counter type</b>	Event value counter
<b>Event</b>	A request that was received by this connector has been handled
<b>Value</b>	Frequency and duration of request handling of this application connector
<b>Hints</b>	None

## Operations

### **getPacakagesBeingProcessed**

<b>Description</b>	Retrieves list of application packages (CAPs) processed (deploy, upgrade, undeploy, and rollback) at a specific time
<b>Parameters</b>	None
<b>Return value</b>	Test message containing the list of application packages processed by the application package service

## Alerts

None

## Problems

None

## Subcomponents

CAP Connector has three subcomponents:

- Deployment
- WebServiceOperations

- XDSRepository

## Deployment (Managed components)

This subcomponent contains the list of counters, which help track the number of occurrences of deployment actions that are invoked. The deployment actions include:

- Deployment
- Revert
- Rollback

The counters also track the number of invocations that are called per second.

### **Settings**

None

### **Counters**

#### **DeploymentOperations**

<b>Counter type</b>	Event value counter
<b>Event</b>	Handling of a request of type deployment that was received by this connector
<b>Value</b>	Frequency and count of the deployment request handling
<b>Hints</b>	None

#### **RevertOperations**

<b>Counter type</b>	Event value counter
<b>Event</b>	Handling of a request of type deployment to revert an application package that was received by this connector
<b>Value</b>	Frequency and count of the deployment with revert request handling
<b>Hints</b>	None

#### **RollbackOperations**

<b>Counter type</b>	Event value counter
<b>Event</b>	Handling of a request of type deployment to perform a rollback action to revert an application package to the

	previous revision
<b>Value</b>	Frequency and count of the deployment with rollback to the previous version request handling
<b>Hints</b>	None

## Alerts

None

## Problems

None

## Subcomponents

The application package connector type has one subcomponent: [ArtifactDeployment \(Managed components\)](#).

### **ArtifactDeployment (Managed components)**

This subcomponent contains the list of counters that help track the number of occurrences of the artifact actions when deploying application packages.

## Settings

None

## Counters

### **ArtifactActions**

The following counters are available for various artifact actions:

- Create
- Delete
- Move
- Rename
- Update
- Delete from another package

<b>Counter type</b>	Event value counter
<b>Event</b>	Handling of a request for a specific artifact action that was received by this connector
<b>Value</b>	Frequency and duration of the request handling of the specific action
<b>Hints</b>	None

**Alerts**

None

**Problems**

None

## WebServiceOperations (Managed components)

This subcomponent contains the list of counters that track the number of occurrences and frequency of all the Web service operations of the application package connector component.

These counters help analyze the time taken to process a SOAP request for a specific Web service operation.

**Settings**

None

**Counters**

### WbserviceOperations

<b>Counter type</b>	Event value counter
<b>Event</b>	Handling of a request for a specific Web service operation that was received by this connector
<b>Value</b>	Frequency and duration of the request handling of the specific Web service operation
<b>Hints</b>	None

**Alerts**

None

**Problems**

None

## Case Management Engine (Managed components)

**Settings**

None

## Counters

### *numCacheHits*

<b>Counter type</b>	Value counter
<b>Value</b>	numCacheHits
<b>Hints</b>	None

### *numCacheMisses*

<b>Counter type</b>	Value counter
<b>Value</b>	numCacheMisses
<b>Hints</b>	None

### *numCacheRevisionIDHits*

<b>Counter type</b>	Value counter
<b>Value</b>	numCacheRevisionIDHits
<b>Hints</b>	None

### *numCacheRevisionIDMisses*

<b>Counter type</b>	Value counter
<b>Value</b>	numCacheRevisionIDMisses
<b>Hints</b>	None

### *numCacheRevisionIDSize*

<b>Counter type</b>	Value counter
<b>Value</b>	numCacheRevisionIDSize
<b>Hints</b>	None

### *numCacheSize*

<b>Counter type</b>	Value counter
<b>Value</b>	numCacheRevisionIDSize
<b>Hints</b>	None

## Alerts

None

## Problems

None

## Subcomponents

None

# CoBOCAppConnector (Managed components)

## Settings

None

## Counters

### requestHandlingTimer

<b>Counter type</b>	Event value counter
<b>Value</b>	Frequency and duration of the request handling

## Operations

None

## Alerts

### dbConnectionFailure

<b>Severity level</b>	Error.
<b>Message</b>	The database server may not be functioning, or the network may be encountering server connectivity problems.
<b>Description</b>	This alert is issued when processor cannot be started because of a database connectivity issue.

**stateSynchUpFailure**

<b>Severity level</b>	Error.
<b>Message</b>	State synch up is not configured, state synch up is required to enable Orchestrator cache synchronization.
<b>Description</b>	This alert is issued because of state synch up configuration problems.

**MissingClassDefinition**

<b>Severity level</b>	Error.
<b>Message</b>	Class {0} not found.
<b>Description</b>	This alert is issued because the class definition has not been found.

**notificationSOAPProcessorFailure**

<b>Severity level</b>	Error.
<b>Message</b>	The system was not able to deliver the notification from the process engine. The Notification SOAP processor is not functioning properly, or has not yet been started.
<b>Description</b>	This alert is issued because of the Notification SOAP processor failure.

**unableToResolveDso**

<b>Severity level</b>	Error.
<b>Message</b>	The process engine was not able to resolve the database configuration from the specified information in the business process model (transactional process: {0} for instance id: {1}.)
<b>Description</b>	This alert is issued because of issues in resolving the database configuration for the transactional process.

## Problems

**databaseConnectionFailure**

<b>problemRaiseDescription</b>	The database server is not available. The database server may not be functioning, or the network may be encountering server connectivity problems. Ensure that the database
--------------------------------	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------

	server is functioning, and the network is communicating properly with the server.
<b>problemClearDescription</b>	Connection to the database server has been established.

## Subcomponents

The CoBOCApplConnector type has five subcomponents:

- Business Object Cache
- CoBOC DBConnectionPool
- Rule Engine
- Rule Repository ConnectionPool
- Scheduler DBConnectionPool

## CompressionSettings

### Settings

#### *contentSize*

<b>Setting type</b>	Hot
<b>Type</b>	Integer
<b>Description</b>	Facilitates conditional compression wherein only the data that exceeds a specified value (in KB) is compressed. If the value is set to zero, the entire content is compressed. Accepted values are 0 or a positive unsigned integer.

#### *enableCompression*

<b>Setting type</b>	Hot
<b>Type</b>	String
<b>Description</b>	Setting to true enables data compression using XQY

### Counters

None

### Alerts

None

## Subcomponents

None

## CWS server (Managed components)

### Settings

None

### Counters

#### **requestHandlingTimer**

<b>Counter type</b>	Event value counter
<b>Event</b>	Frequency and duration of the request handling
<b>Hints</b>	None

### Operations

#### **reset**

<b>Description</b>	Reset processor
<b>Parameters</b>	None
<b>Return value</b>	None

### Alerts

None

### Problems

#### **databaseConnectionFailure**

<b>problemRaiseDescription</b>	The database server is not available. The database server may not be functioning, or the network may be encountering server connectivity problems. Ensure that the database server is functioning, and the network is communicating properly with the server.
<b>problemClearDescription</b>	Connection to the database server has been established.

## Subcomponents

The CWS Server type has two subcomponents: Server Broker and XDSRepository.

## Data transformation service (Managed components)

### Settings

None

### Counters

#### requestHandlingTimer

<b>Counter type</b>	Event value counter
<b>Value</b>	Frequency and duration of the request handling
<b>Hints</b>	None

### Alerts

#### generalConnectorDescription

<b>Severity level</b>	Info
<b>Message</b>	Basic DataMapper Service
<b>Description</b>	Basic DataMapper Service

#### dbConnectionFailure

<b>Severity level</b>	Error.
<b>Message</b>	The database server may not be functioning, or the network may be encountering server connectivity problems.
<b>Description</b>	This alert is issued when the processor cannot be started because of a database connectivity problem.

#### stateSyncUpFailure

<b>Severity level</b>	Error.
-----------------------	--------

<b>Message</b>	State synch up is not configured, state synch up is required to enable orchestrator cache synchronization.
<b>Description</b>	This alert is issued because of state synch up configuration problems.

### MissingClassDefinition

<b>Severity level</b>	Error.
<b>Message</b>	Class definition not found.
<b>Description</b>	This alert is issued because the class definition has not been found.

### notificationSOAPProcessorFailure

<b>Severity level</b>	Error.
<b>Message</b>	The system was not able to deliver the notification from the process engine. The Notification SOAP processor is not functioning properly or has not yet been started.
<b>Description</b>	This alert is issued because of the Notification SOAP processor failure.

### unableToResolveDso

<b>Severity level</b>	Error.
<b>Message</b>	The process engine was not able to resolve the database configuration from the specified information in the business process model (transactional process: {0} for instance id: {1}).
<b>Description</b>	This alert is issued because of issues in resolving database configuration for the transactional process.

## Problems

### databaseConnectionFailure

<b>problemRaiseDescription</b>	The database server is not available. The database server may not be functioning, or the network may be encountering server connectivity problems. Ensure that the database server is functioning, and the network is communicating properly with the server.
<b>problemClearDescription</b>	Connection to the database server has been established.

## Subcomponents

The DataTransformationService type has two subcomponents: Business Object Cache and CoBOC DBConnectionPool.

## DBConnectionPool (Managed components)

### Settings

#### **auditInfo**

<b>Setting type</b>	Hot
<b>Type</b>	String
<b>Description</b>	XML structure containing audit information

#### **conversionConfig**

<b>Setting type</b>	Hot
<b>Type</b>	String
<b>Description</b>	XML structure containing data conversion information

#### **cursorCacheRefreshInterval**

<b>Setting type</b>	Hot
<b>Type</b>	Integer
<b>Description</b>	Refresh interval for query cache

#### **cursorCacheSize**

<b>Setting type</b>	Hot
<b>Type</b>	Integer
<b>Description</b>	Cursor cache size per connection

#### **datasource**

<b>Setting type</b>	Hot
<b>Type</b>	String
<b>Description</b>	XML structure containing datasource information

**defaultCursorSize**

<b>Setting type</b>	Hot
<b>Type</b>	Integer
<b>Description</b>	Default cursor size

**maximumReadConnections**

<b>Setting type</b>	Hot
<b>Type</b>	Integer
<b>Description</b>	Number of maximum read connections

**maximumWriteConnections**

<b>Setting type</b>	Hot
<b>Type</b>	Integer
<b>Description</b>	Number of maximum write connections

**minimumReadConnections**

<b>Setting type</b>	Hot
<b>Type</b>	Integer
<b>Description</b>	Number of minimum write connections

**minimumWriteConnections**

<b>Setting type</b>	Hot
<b>Type</b>	Integer
<b>Description</b>	Number of minimum write connections

**optimiseQueryForCursor**

<b>Setting type</b>	Hot
<b>Type</b>	Boolean
<b>Description</b>	Optimising query for retrieving first set of records using the cursor

**precedence**

<b>Setting type</b>	Hot
<b>Type</b>	String enumeration. Possible values: <ul style="list-style-type: none"><li>■ data</li><li>■ null</li></ul>
<b>Description</b>	Data or null precedence

**reconnectionAttempts**

<b>Setting type</b>	Hot
<b>Type</b>	Integer
<b>Description</b>	Number of attempts to reconnect to the database server

**reconnectionInterval**

<b>Setting type</b>	Hot
<b>Type</b>	Integer
<b>Description</b>	Interval between two attempts to reconnect to the database server

**requestThreshold**

<b>Setting type</b>	Hot
<b>Type</b>	Integer
<b>Description</b>	Threshold value for each request

**xmlEncoding**

<b>Setting type</b>	Hot
<b>Type</b>	Integer
<b>Description</b>	Support for XML special characters

**xsiNilForNullData**

<b>Setting type</b>	Hot
<b>Type</b>	Integer
<b>Description</b>	Setting <code>xsi:nil</code> for absence of data with <code>null=true</code>

## Counters

### **numOfCachedReadConnections**

<b>Counter type</b>	Value counter
<b>Value</b>	Number of read connections in the cache
<b>Hints</b>	None

### **numOfCachedWriteConnections**

<b>Counter type</b>	Value counter
<b>Value</b>	Number of write connections in the cache
<b>Hints</b>	None

### **numberOfOpenCursors**

<b>Counter type</b>	Value counter
<b>Value</b>	Number of currently open cursors in all connections
<b>Hints</b>	None

### **readConnectionUsageDuration**

<b>Counter type</b>	Event value counter
<b>Value</b>	Frequency and duration of usage of a read connection
<b>Hints</b>	None

### **readConnectionWait**

<b>Counter type</b>	Event value counter
<b>Event</b>	Request waiting for a read connection because the pool is empty
<b>Value</b>	Frequency and duration of wait actions for a read connection
<b>Hints</b>	None

### **writeConnectionUsageDuration**

<b>Counter type</b>	Event value counter
<b>Event</b>	Returning a write connection to the connection pool

<b>Value</b>	Frequency and duration of usage of a write connection
<b>Hints</b>	None

**writeConnectionWait**

<b>Counter type</b>	Event value counter
<b>Event</b>	Request waiting for a write connection because the pool is empty
<b>Value</b>	Frequency and duration of wait actions for a write connection
<b>Hints</b>	None

**Alerts****databaseConnectionFailure**

<b>Severity level</b>	Error
<b>Message</b>	The database server is not available. The database server may not be functioning, or the network may be encountering server connectivity problems. Ensure that the database server is functioning, and the network is communicating properly with the server.
<b>Description</b>	This alert is issued when connection to the database server is lost.

**databaseConnectionEstablished**

<b>Severity Level</b>	Info
<b>Message</b>	Connection to the database server has been established.
<b>Description</b>	This alert is issued when connection to database server is successful.

**Problems****DBConnectionFailure**

<b>problemRaiseDescription</b>	This alert is issued when connection to the database server is lost.
<b>problemClearDescription</b>	This alert is issued when connection to the database server is lost.

## Subcomponents

None

# Decision table cache (Managed components)

## Settings

### **cacheSize**

<b>Setting type</b>	Hot
<b>Readable</b>	Yes
<b>Type</b>	Integer
<b>Description</b>	Cache size

## Counters

### **noOfCacheHits**

<b>Counter type</b>	Event occurrence counter
<b>Event</b>	Number of cache hits
<b>Hints</b>	None

### **noOfCacheMisses**

<b>Counter type</b>	Event value counter
<b>Event</b>	Number of cache misses
<b>Hints</b>	None

### **noOfEmptyObjectsInCache**

<b>Counter type</b>	Value counter
<b>Value</b>	Number of empty objects in the cache
<b>Hints</b>	None

**noOfObjectsInCache**

<b>Counter type</b>	Value counter
<b>Value</b>	Number of objects in the cache
<b>Hints</b>	None

**Alerts**

None

**Problems**

None

**Subcomponents**

None

**Document cache****Settings****cacheSize**

<b>Setting type</b>	Hot
<b>Readable</b>	Yes
<b>Type</b>	Long
<b>Description</b>	Document cache size

**Counters****noOfCacheHits**

<b>Counter type</b>	Event occurrence counter
<b>Event</b>	Number of cache hits
<b>Hints</b>	None

**noOfCacheMisses**

<b>Counter type</b>	Event occurrence counter
---------------------	--------------------------

<b>Event</b>	Number of cache misses
<b>Hints</b>	None

### **noOfObjectsInCache**

<b>Counter type</b>	Value counter
<b>Value</b>	Number of objects in the cache
<b>Hints</b>	None

## Alerts

None

## Operations

<b>Description</b>	Clears the XDS document cache
<b>Parameters</b>	None
<b>Return value</b>	String

## Subcomponents

None

# Document store connector

## Settings

None

## Counters

### **requestHandlingTimer**

<b>Counter type</b>	Event value counter
<b>Event</b>	Frequency of the request handling
<b>Hints</b>	None

## Alerts

None

## Subcomponents

The Document Store Connector has one sub-component. The name of the sub-component is the store name from the document store connector configuration.

### Settings

None

### Counters

#### outboundServiceCallDuration

<b>Counter type</b>	Timer event value counter
<b>Event</b>	Making an outbound request to an external DMS and receiving a response
<b>Value</b>	Frequency and duration of the outbound API call to the external DMS
<b>Hints</b>	None

### Operations

#### Reset

<b>Description</b>	Resets the counter identified by the name <code>outboundServiceCallDuration</code> for this subcomponent
<b>Parameters</b>	None
<b>Return value</b>	None

### Alerts

None

## Download gateway (Managed components)

### Settings

None

## Counters

### *DownloadFailures*

<b>Counter type</b>	Event occurrence counter
<b>Event</b>	
<b>Value</b>	Total download failures
<b>Hints</b>	None

### *DownloadRequestProcessingTime*

<b>Counter type</b>	Event value counter
<b>Event</b>	
<b>Value</b>	Total time for processing the download request
<b>Hints</b>	None

### *DownloadSuccess*

<b>Counter type</b>	Event occurrence counter
<b>Event</b>	
<b>Value</b>	Total download successes
<b>Hints</b>	None

### *MaxFileSizeDownloaded*

<b>Counter type</b>	Value counter
<b>Event</b>	
<b>Value</b>	Maximum file size downloaded
<b>Hints</b>	None

### *downloadFileContent*

<b>Counter type</b>	Event value counter
<b>Event</b>	
<b>Value</b>	Number of download requests using downloadFileContentRequest
<b>Hints</b>	None

### ***downloadFileLocation***

<b>Counter type</b>	Value counter
<b>Event</b>	
<b>Value</b>	Number of download requests using downloadFileLocationRequest
<b>Hints</b>	None

### ***downloadSize***

<b>Counter type</b>	Event value counter
<b>Event</b>	Downloading a file
<b>Value</b>	Download size
<b>Hints</b>	None

### ***requestHandlingDuration***

<b>Counter type</b>	Event value counter
<b>Event</b>	
<b>Value</b>	Duration and frequency of request handling
<b>Hints</b>	None

## **Alerts**

### ***gatewayErrorLdapConnectionFailure***

<b>Severity</b>	Error
<b>Message</b>	Gateway failed to process the request due to failure to establish connection with the LDAP service container.
<b>Description</b>	This alert is issued when gateway fails to process request because LDAP server connection fails.

### ***gatewayErrorSocketFailure***

<b>Severity</b>	Error
<b>Message</b>	Gateway failed to process the request due to failure to open the socket connection.

<b>Description</b>	This alert is issued when the gateway fails to process the request because of a socket problem.
--------------------	-------------------------------------------------------------------------------------------------

## Problems

None

## Subcomponents

None

# Email connector (Managed components)

## Settings

### authenticationRequired

<b>Setting type</b>	Hot
<b>Type</b>	Boolean
<b>Description</b>	Authentication for the SMTP server

### contentType

<b>Setting type</b>	Hot
<b>Type</b>	String
<b>Description</b>	Content type of outgoing mails

### maxNoOfSessions

<b>Setting type</b>	Hot
<b>Type</b>	Integer
<b>Description</b>	Maximum number of sessions

### protocol

<b>Setting type</b>	Hot
<b>Type</b>	String
<b>Description</b>	Protocol of mail server: imap or pop3

## Counters

None

## Alerts

### mailServerConnectionFailure

<b>Severity level</b>	Error
<b>Message</b>	The client is not able to connect to the mail server. The mail server may not be available, or the port may be busy or there may be other network problems.
<b>Description</b>	This alert is issued when connection to the mail server fails.

### mailServerConnectionEstablished

<b>Severity level</b>	Info
<b>Message</b>	Connection to the mail server has been established.
<b>Description</b>	This alert is issued when connection to the mail server is successful.

## Problems

### mailServerConnectionFailure

<b>problemRaiseDescription</b>	The client is not able to connect to the mail server. The mail server may not be available, or the port may be busy or there may be other network problems.
<b>problemClearDescription</b>	Connection to the mail server has been established.

## Subcomponents

None

## Embedded data mapper

### Settings

None

## Counters

None

## Alerts

### **dbConnectionFailure**

<b>Severity level</b>	Error
<b>Message</b>	The database server may not be functioning, or the network may be encountering server connectivity problems.
<b>Description</b>	This alert is issued when the processor cannot be started because of a database connectivity problem.

### **stateSynchUpFailure**

<b>Severity level</b>	Error
<b>Message</b>	State synch up (SSU) is not configured. State synchup is required for synchronization among multiple BPM Engines configured for load balancing.
<b>Description</b>	This alert is issued because of State Synchup configuration problems.

### **MissingClassDefinition**

<b>Severity level</b>	Error
<b>Message</b>	Class {0} not found.
<b>Description</b>	This alert is issued because the class definition was not found.

### **notificationSOAPProcessorFailure**

<b>Severity level</b>	Error
<b>Message</b>	The system was not able to deliver the notification from the process engine. The Notification SOAP processor is not functioning properly or has not yet been started.
<b>Description</b>	This alert is issued because of Notification SOAP processor failure.

**unableToResolveDso**

<b>Severity level</b>	Error
<b>Message</b>	The process engine was not able to resolve the database configuration from the specified information in the business process model (transactional process: {0} for instance id: {1}.)
<b>Description</b>	This alert is issued because of issues in resolving database configuration for the transactional process.

## Problems

**databaseConnectionFailure**

<b>problemRaiseDescription</b>	The database server is not available. The database server may not be functioning, or the network may be encountering server connectivity problems. Ensure that the database server is functioning, and the network is communicating properly with the server.
<b>problemClearDescription</b>	Connection to the database server has been established.

## Subcomponents

The Embedded Data Mapper type has two subcomponents: Business Object Cache and CoBOC DBConnectionPool.

# Event service (Managed components)

## Settings

None

## Counters

**requestHandlingTimer**

<b>Counter type</b>	Event value counter
<b>Event</b>	Frequency of the request handling
<b>Hints</b>	None

## Alerts

None

## Subcomponents

None

# FTP connector (Managed components)

## Settings

None

## Counters

### requestHandlingTimer

<b>Counter type</b>	Event value counter
<b>Event</b>	Frequency of the request handling
<b>Hints</b>	None

## Alerts

None

## Problems

None

## Subcomponents

None

# Gateway ThreadPool (Managed components)

## Settings

None

## Counters

### ***currentNumWorkers***

<b>Counter type</b>	Value counter
<b>Event</b>	
<b>Value</b>	Number of current worker threads
<b>Hints</b>	None

### ***numIdleWorkers***

<b>Counter type</b>	Value counter
<b>Event</b>	Frequency of the requests handled
<b>Value</b>	Number of idle worker threads
<b>Hints</b>	None

## Alerts

None

## Problems

None

## Subcomponents

None

# GatewayWebApplication (Managed components)

This managed component type is the supertype for a group of Web applications.

## Settings

None

## Counters

None

## Alerts

### *gatewayErrorLdapConnectionFailure*

<b>Severity level</b>	Error
<b>Message</b>	Gateway failed to process the request due to failure to establish connection with the LDAP SOAP Processor.
<b>Description</b>	This alert is issued when the gateway fails to process the request because the LDAP server connection fails.

### *gatewayErrorSocketFailure*

<b>Severity level</b>	Error
<b>Message</b>	Gateway failed to process the request due to failure to open the socket connection.
<b>Description</b>	This alert is issued when the gateway fails to process the request because of a socket problem.

## Problems

None

## Subcomponents

None

# Gossip (Managed components)

## Settings

### *clientPortRangeEnd*

<b>Setting type</b>	Hot
<b>Type</b>	Integer
<b>Description</b>	End of the port range that this member uses for outgoing connections. Default is 0.

### ***clientPortRangeStart***

<b>Setting type</b>	Hot
<b>Type</b>	Integer
<b>Description</b>	Start of the port range that this member uses for outgoing connections. Default is 0.

### ***connectTimeout***

<b>Setting type</b>	Hot
<b>Type</b>	Integer
<b>Description</b>	Period a member waits for a connection to be established in milliseconds. Default is 1000.

### ***heartbeatInterval***

<b>Setting type</b>	Hot
<b>Type</b>	Integer
<b>Description</b>	Frequency of sending heartbeat (I am alive) messages in milliseconds. Default is 1000.

### ***investigationTimeout***

<b>Setting type</b>	Hot
<b>Type</b>	Integer
<b>Description</b>	Period a member waits for the next investigation message in milliseconds. Default is 1000.

### ***maxFailureTimeout***

<b>Setting type</b>	Hot
<b>Type</b>	Integer
<b>Description</b>	Maximum delay of a heartbeat message under busy circumstances in milliseconds. Default is 7000.

### ***maxHeartbeatDeviation***

<b>Setting type</b>	Hot
---------------------	-----

<b>Type</b>	Integer
<b>Description</b>	Acceptable delay of a heartbeat message in milliseconds. Default is 50.

### ***maxMessageSize***

<b>Setting type</b>	Hot
<b>Type</b>	Integer
<b>Description</b>	Maximum message size in bytes (used for buffer sizes). Default is 16384.

### ***maxP2PMessageAckWaitTimeout***

<b>Setting type</b>	Hot
<b>Type</b>	Integer
<b>Description</b>	Maximum time a member waits for its neighbor to acknowledge a message in milliseconds. Default is 1000.

### ***maxdropFailureTimeout***

<b>Setting type</b>	Hot
<b>Type</b>	Integer
<b>Description</b>	Maximum decrease of timeout per heartbeat message in milliseconds. Default is 100.

### ***minConnections***

<b>Setting type</b>	Hot
<b>Type</b>	Integer
<b>Description</b>	Minimum number of network connections per member. Default is 5.

### ***myClientPortRangeEnd***

<b>Setting type</b>	Hot
<b>Type</b>	Integer
<b>Description</b>	End of the port range that this member uses for outgoing connections. Default is 0.

### ***myClientPortRangeStart***

<b>Setting type</b>	Hot
<b>Type</b>	Integer
<b>Description</b>	Start of the port range that this member uses for outgoing connections. Default is 0.

### ***myHost***

<b>Setting type</b>	Hot
<b>Type</b>	String
<b>Description</b>	Host that this member uses. Default is all addresses.

### ***myServerPort***

<b>Setting type</b>	Hot
<b>Type</b>	Integer
<b>Description</b>	Port where this member listens. Default is 10001.

### ***normalFailureTimeout***

<b>Setting type</b>	Hot
<b>Type</b>	Integer
<b>Description</b>	Maximum delay of a heartbeat message under normal circumstances in milliseconds. Default is 3000.

### ***randomConnectInterval***

<b>Setting type</b>	Hot
<b>Type</b>	Integer
<b>Description</b>	Frequency of dropping a connection and connecting to a different member. Default is 120000.

## **Info properties**

### ***numberOfConnections***

<b>Type</b>	Integer
<b>Description</b>	Number of connections (neighbors)

## ***neighbors***

<b>Type</b>	String
<b>Description</b>	Names of current neighbors

## ***NetworkInterfaceCard***

<b>Type</b>	String
<b>Description</b>	Network interface card in use

## **Counters**

### ***applicationReceivedCounter***

<b>Counter type</b>	Value counter
<b>Value</b>	Number of application messages received
<b>Hints</b>	None
<b>Version info</b>	Deleted in Cordys BOP 4.1 CU7

### ***elapsedInvestigationTime***

<b>Counter type</b>	Value counter
<b>Value</b>	Elapsed investigation time
<b>Hints</b>	None

### ***inboundPendingRequestsQueueSize***

<b>Counter type</b>	Value counter
<b>Value</b>	Number of messages that are waiting to be pushed to SynchronousAlgorithm
<b>Hints</b>	None
<b>Version info</b>	Added in Cordys BOP 4.1 CU7

### ***messagesDeliveredToApplication***

<b>Counter type</b>	Event occurrence counter
<b>Value</b>	Number of application messages received

<b>Hints</b>	None
<b>Version info</b>	Added in Cordys BOP 4.1 CU7

### ***numberOfActiveMembers***

<b>Counter type</b>	Value counter
<b>Value</b>	Total number of active members in the current graph
<b>Hints</b>	None

### ***numberOfEffectiveMembershipMessages***

<b>Counter type</b>	Value counter
<b>Value</b>	Total effective number of membership messages received
<b>Hints</b>	None

### ***numberOfFailedMembers***

<b>Counter type</b>	Value counter
<b>Value</b>	Total number of failed members in the current graph
<b>Hints</b>	None

### ***numberOfMembershipMessages***

<b>Counter type</b>	Value counter
<b>Value</b>	Total number of membership messages received
<b>Hints</b>	None

### ***numberOfReceivedMessages***

<b>Counter type</b>	Event Occurrence counter
<b>Value</b>	Number of messages received
<b>Hints</b>	None

### ***numberOfSentMessages***

<b>Counter type</b>	Value counter
---------------------	---------------

<b>Value</b>	Total number of messages sent
<b>Hints</b>	None
<b>Version info</b>	Replaced by <code>sentMessagesSize_occurrences</code> in Cordys BOP 4.1 CU7

### ***outboundPendingRequestsQueueSize***

<b>Counter type</b>	Value counter
<b>Value</b>	Number of messages that are waiting to be pushed to <code>Gossipip</code>
<b>Hints</b>	None
<b>Version info</b>	Added in Cordys BOP 4.1 CU7

### ***retransmittedMessages***

<b>Counter type</b>	Event occurrence counter
<b>Value</b>	Number of application messages retransmitted
<b>Hints</b>	None
<b>Version info</b>	Added in Cordys BOP 4.1 CU7

### ***sendTimer***

<b>Counter type</b>	Timer event value counter
<b>Value</b>	Timer for sending the messages over the <code>Gossipip</code> cluster
<b>Hints</b>	None

### ***sentMessagesSize***

<b>Counter type</b>	Event value counter
<b>Value</b>	Size of sent messages
<b>Hints</b>	None
<b>Version info</b>	Added in Cordys BOP 4.1 CU7

### ***toBeProcessedRequestsFromOutboundQueueSize***

<b>Counter type</b>	Value counter
<b>Value</b>	Number of <code>Gossipip</code> messages that are waiting for processing

	by the GD algorithm
<b>Hints</b>	None
<b>Version info</b>	Added in Cordys BOP 4.1 CU7

## Alerts

None

## Subcomponents

None

# HTTP connector (Managed components)

## Settings

None

## Counters

### requestTransformation

<b>Counter type</b>	Timer event value counter
<b>Event</b>	Receiving a SOAP request that must be transformed into the desired structure as defined by the XSL provided in the HTTP Connector configuration
<b>Value</b>	Time taken for applying XSL transformation to the HTTP request message
<b>Hints</b>	None

### httpProcessing

<b>Counter type</b>	Timer event value counter
<b>Event</b>	Making an outbound API call and receiving the response
<b>Value</b>	Time taken by the external system to process the outbound request invoked by the HTTP Connector
<b>Hints</b>	None

**responseTransformation**

<b>Counter type</b>	Timer event value counter
<b>Event</b>	Receiving and submitting an HTTP response (of external HTTP API call) for XSL transformation as defined by the XSL provided in the HTTP Connector configuration
<b>Value</b>	Time taken for applying XSL transformation to the HTTP response message
<b>Hints</b>	None

**jsonToXmlTransformation**

<b>Counter type</b>	Timer event value counter
<b>Event</b>	Receiving an HTTP response of the JSON content type
<b>Value</b>	Time taken to convert JSON to XML
<b>Hints</b>	None

**xmlToJsonTransformation**

<b>Counter type</b>	Timer event value counter
<b>Event</b>	Receiving a SOAP request as XML that must be transformed into JSON to make an outbound JSON call
<b>Value</b>	Time taken to convert XML to JSON
<b>Hints</b>	None

## Operations

**reset**

<b>Description</b>	Resets all the counters identified by the HTTP Connector component
<b>Parameters</b>	None
<b>Return value</b>	None

## Alerts

None

## Problems

None

## Subcomponents

None

# ISVP connector (Managed components)

## Settings

### username

<b>Setting type</b>	Hot
<b>Type</b>	String
<b>Description</b>	User name to sign in to the connector

### password

<b>Setting type</b>	Hot
<b>Type</b>	String
<b>Description</b>	Password to sign in to the connector

## Counters

### requestHandlingTimer

<b>Counter type</b>	Event value counter
<b>Event</b>	Handling a received request
<b>Value</b>	Frequency and duration of the request handling
<b>Hints</b>	None

## Alerts

None

## Problems

### databaseConnectionFailure

<b>Problem Raise Description</b>	The database server is not available. The database server may not be functioning, or the network may be encountering server connectivity problems. Ensure that the database server is functioning, and the network is communicating properly with the server.
<b>problem Clear Description</b>	Connection to the database server has been established.

## Subcomponents

ISVP Connector type has one sub-component: [XDSRepository \(Managed components\)](#).

## LDAP cache (Managed components)

This type represents the cache of LDAP entries.

## Settings

### maxCacheSize

<b>Setting type</b>	Hot
<b>Readable</b>	Yes
<b>Type</b>	Long
<b>Description</b>	Maximum LDAP cache size

## Counters

### accessdenied

<b>Counter type</b>	Event occurrence counter
<b>Event</b>	Looking up and finding an entry in the LDAP cache
<b>Value</b>	Frequency in which items are denied access
<b>Hints</b>	None

**cacheHits**

<b>Counter type</b>	Event occurrence counter
<b>Event</b>	Looking up and finding an entry in the LDAP cache
<b>Value</b>	Frequency of cache hits
<b>Hints</b>	None

**cacheMisses**

<b>Counter type</b>	Event occurrence counter
<b>Event</b>	Looking up and finding an entry in the LDAP cache implying that the entry had to be retrieved from the LDAP SOAP processor
<b>Value</b>	Frequency of cache misses
<b>Hints</b>	When this counter reaches significant values, consider expanding the cache size ( <code>bus.ldap.cache.maxSize</code> in the <code>wcp.properties</code> file)

**cacheSize**

<b>Counter type</b>	Value counter
<b>Event</b>	
<b>Value</b>	Number of items in the LDAP cache
<b>Hints</b>	None

**lookups**

<b>Counter type</b>	Event occurrence counter
<b>Event</b>	Looking up an entry in the LDAP cache
<b>Value</b>	Frequency of cache lookups
<b>Hints</b>	If this counter reaches significant values on the LDAP processor, verify the <code>cacheMisses</code> counter on the other SOAP processors to see whether their caches are configured large enough

**removes**

<b>Counter type</b>	Event occurrence counter
---------------------	--------------------------

<b>Event</b>	Removing an entry from the LDAP cache
<b>Value</b>	Frequency in which items are removed from the cache
<b>Hints</b>	None

## Alerts

None

## Problems

None

## Subcomponents

LDAP Cache has one sub-component type: [SOAP connector \(Managed components\)](#).

## LDAP connector (Managed components)

### Settings

None

### Counters

#### requestHandlingTimer

<b>Counter type</b>	Event value counter
<b>Event</b>	
<b>Value</b>	Frequency and duration of the request handling
<b>Hints</b>	None

## Alerts

#### ldapConnectionFailure

<b>Severity level</b>	Error
<b>Message</b>	The LDAP Server is not available. There was a problem connecting to the server. Ensure that the server is running.
<b>Description</b>	This alert is issued when connection to the LDAP server fails.

## Problems

### directoryConnectionFailure

<b>Problem Raise Description</b>	The directory server is not available. The directory server may not be functioning, or the network may be encountering server connectivity problems. Ensure that the directory server is functioning, and the network is communicating properly with the server.
<b>Problem Clear Description</b>	Connection to the directory server has been established.

## Subcomponents

The LDAP Connector type has two sub-components: [LDAP cache \(Managed components\)](#) and [LDAPConnection \(Managed components\)](#).

## LDAPConnection (Managed components)

### Settings

None

### Counters

#### accessdenied

<b>Counter type</b>	Event occurrence counter
<b>Event</b>	Frequency at which requests are denied
<b>Hints</b>	None

#### addDurationCounter

<b>Counter type</b>	Event value counter
<b>Event</b>	Frequency at which an object is added to LDAP
<b>Value</b>	Time taken for each addition
<b>Hints</b>	None

**deleteDurationCounter**

<b>Counter type</b>	Event value counter
<b>Event</b>	Frequency at which an object is deleted from LDAP
<b>Value</b>	Time taken for each deletion
<b>Hints</b>	None

**modifyDurationCounter**

<b>Counter type</b>	Event value counter
<b>Event</b>	Frequency at which an object is modified in LDAP
<b>Value</b>	Time taken for each modification
<b>Hints</b>	None

**searchDuration**

<b>Counter type</b>	Event value counter
<b>Event</b>	Frequency of the LDAP search
<b>Value</b>	Time taken for the search
<b>Hints</b>	None

## Alerts

**directoryConnectionFailure**

<b>Severity level</b>	Error
<b>Message</b>	The directory server is not available. The directory server may not be functioning, or the network may be encountering server connectivity problems. Ensure that the directory server is functioning, and the network is communicating properly with the server.
<b>Description</b>	This alert is issued when connection to the directory server is lost.

**directoryConnectionEstablished**

<b>Severity level</b>	Error
<b>Message</b>	Connection to the directory server has been established.
<b>Description</b>	This alert is issued when connection to the directory server is reestablished.

**IdapReconnectionSuccessful**

<b>Severity level</b>	Error
<b>Message</b>	
<b>Description</b>	This alert is issued when connection to the LDAP server failed but immediate reconnection succeeded.

## Problems

**directoryConnectionFailure**

<b>Problem Raise description</b>	The directory server is not available. The directory server may not be functioning, or the network may be encountering server connectivity problems. Ensure that the directory server is functioning, and the network is communicating properly with the server.
<b>Problem Clear description</b>	Connection to the directory server has been established.

## Subcomponents

None

# License monitor (Managed components)

## Settings

None

## Counters

None

## Alerts

**reportSentToMaster**

<b>Severity level</b>	Information
<b>Message</b>	The usage report has been sent to the master gateway {0} and the response is {1}.
<b>Description</b>	This alert is issued when the usage report is sent to the master gateway and a response is received.

### ***reportSentToLicense***

<b>Severity level</b>	Information
<b>Message</b>	The usage report has been sent to the license service {0} and the response is {1}.
<b>Description</b>	This alert is issued when the usage report is sent to the license service and a response is received.

### ***consolidatedReportNotSendLicenseServiceFailure***

<b>Severity level</b>	Error
<b>Message</b>	The consolidated report could not be sent to the License Service {0}.
<b>Description</b>	This alert is issued when the consolidated report cannot be sent to the license service.

### ***licenseReportRecieveFailure***

<b>Severity level</b>	Warning
<b>Message</b>	The report could not be obtained from the {0}. Please check the status of the Web Server and Java Call Service on this machine.
<b>Description</b>	This alert is issued when the license service fails to receive the report from a computer.

### ***licenseKeyLoadFailure***

<b>Severity level</b>	Warning
<b>Message</b>	The key could not be loaded on the {0}. Please check the status of the Web Server and Java Call Service on this machine.
<b>Description</b>	This alert is issued when the license key cannot be loaded on a computer.

### ***usageReportNotSendGatewayError***

<b>Severity level</b>	Error
<b>Message</b>	The usage report could not be sent to the master gateway. {0}.

<b>Description</b>	This alert is issued when the usage report cannot be sent to the master gateway.
--------------------	----------------------------------------------------------------------------------

### ***usageReportNotSendLicenseServiceFailure***

<b>Severity level</b>	Error
<b>Message</b>	The usage report could not be sent to the License Service {0}.
<b>Description</b>	This alert is issued when the usage report cannot be sent to the license service.

## **Problems**

None

## **Subcomponents**

None

# **MDM hub publisher**

## **Settings**

None

## **Counters**

### **load**

<b>Counter type</b>	Value counter
<b>Event</b>	Creating a JMX attribute load
<b>Hints</b>	None

### **requestHandlingTimer**

<b>Counter type</b>	Event value counter
<b>Event</b>	Frequency and duration of the request handling
<b>Hints</b>	None

## Alerts

None

## Operations

### **getWorks**

<b>Description</b>	Lists all work performed by all the SOAP processors under the current processor's node if it is a coordinator
<b>Parameters</b>	None
<b>Return value</b>	String

### **changeThreshold**

<b>Description</b>	Changes threshold of all the processors under the current processor node
<b>Parameters</b>	threshold
<b>Return value</b>	String

### **getThreshold**

<b>Description</b>	Retrieves threshold of the current processor
<b>Parameters</b>	None
<b>Return value</b>	Double

### **reallocateWork**

<b>Description</b>	Reallocates work amongst all the processors
<b>Parameters</b>	None
<b>Return value</b>	String

### **isCoordinator**

<b>Description</b>	Checks if the current processor is the coordinator
<b>Parameters</b>	None
<b>Return value</b>	Boolean

**getLoad**

<b>Description</b>	Retrieves load of the current processor
<b>Parameters</b>	None
<b>Return value</b>	Double

## Problems

**databaseConnectionFailure**

<b>problemRaiseDescription</b>	The database server is not available. The database server may not be functioning, or the network may be encountering server connectivity problems. Ensure that the database server is functioning, and the network is communicating properly with the server.
<b>problemClearDescription</b>	Connection to the database server has been established.

## Subcomponents

The MDM Hub Publisher type has one subcomponents: [DBConnectionPool \(Managed components\)](#).

## Middleware wrapper (Managed components)

This managed component type represents the wrapper that is derived from the middleware used for the message transport.

## Settings

None

## Counters

**failedSend**

<b>Counter type</b>	Event occurrence counter
<b>Event</b>	Attempting to send a message fails
<b>Value</b>	Frequency of sending a message
<b>Hints</b>	None

## **sendTimer**

<b>Counter type</b>	Event value counter
<b>Event</b>	Sending a message through this middleware wrapper
<b>Value</b>	Frequency and duration of sending the message
<b>Hints</b>	None

## **Alerts**

None

## **Problems**

None

## **Subcomponents**

Middleware wrapper has one sub-component type: Dispatcher.

# **Monitor connector (Managed components)**

## **Settings**

### **spBatchSize**

<b>Setting type</b>	Hot
<b>Type</b>	Integer
<b>Description</b>	Maximum number of service containers that can be started simultaneously

### **spStartUpTime**

<b>Setting type</b>	Hot
<b>Type</b>	Long
<b>Description</b>	Maximum timeout allowed for a service container to start before accepting another SOAP processor in the batch

## Counters

### requestHandlingTimer

<b>Counter type</b>	Event value counter
<b>Value</b>	Frequency and duration of the request handling
<b>Hints</b>	None

## Alerts

### monitorStartErrorLdapFailure

<b>Severity level</b>	Error
<b>Message</b>	The AppWorks Platform monitor could not be started due to failure to access the LDAP Server. Please ensure that the LDAP server is running. If so, the <machine> could be slow. Add the 'bus.ldaprequest.timeout=<timeout>' property in the wcp.properties file and specify the duration for request timeout.
<b>Description</b>	This alert is issued when the monitor cannot be started because of an LDAP connectivity problem.

### processLaunchError

<b>Severity level</b>	Error
<b>Message</b>	The process {0} could not be launched.
<b>Description</b>	This alert is issued when launching of the process fails.

### soapProcessorFailure

<b>Severity level</b>	Error
<b>Message</b>	SOAP processor {0} has crashed.
<b>Description</b>	This alert is issued when the SOAP processor becomes unresponsive.

### soapProcessorStopped

<b>Severity level</b>	Error
<b>Message</b>	The SOAP processor {0} has been stopped.
<b>Description</b>	This alert is issued when the SOAP processor stops.

**soapProcessorStarted**

<b>Severity level</b>	Info
<b>Message</b>	The SOAP processor {0} has been started.
<b>Description</b>	This alert is issued when the SOAP processor started successfully.

**processLaunch**

<b>Severity level</b>	Info
<b>Message</b>	Launching process {0}
<b>Description</b>	This alert is issued when a process is in the starting mode.

**processStop**

<b>Severity level</b>	Info
<b>Message</b>	Stopping process {0}
<b>Description</b>	This alert is issued when a process is in the stopping mode.

## Problems

None

## Subcomponents

None

# Monitoring engine (Managed components)

## Settings

**businessObjectCacheSize**

<b>Setting type</b>	Hot
<b>Data Type</b>	Integer
<b>Description</b>	Maximum number of business objects that can be cached

## Operations

### resetCache

<b>Description</b>	Clears the cache data, removes all the cached objects, and resets the cache counters
<b>Parameters</b>	None
<b>Return value</b>	None

## Counters

### missedEventProcessCount

<b>Counter type</b>	Event counter
<b>Value</b>	Number of events not processed in the first iteration
<b>Hints</b>	None

### noOfCacheHits

<b>Counter type</b>	Event counter
<b>Value</b>	Number of cache hits for business objects
<b>Hints</b>	None

### eventProcessCount

<b>Counter type</b>	Event counter
<b>Value</b>	Number of processed events
<b>Hints</b>	None

### noOfCacheMisses

<b>Counter type</b>	Event counter
<b>Value</b>	Number of cache misses for business objects
<b>Hints</b>	None

### noOfObjectsInCache

<b>Counter type</b>	Event counter
---------------------	---------------

<b>Value</b>	Number of business objects currently in the cache
<b>Hints</b>	None

## Alerts

None

## Problems

None

## Subcomponents

The MonitoringEngine type has two subcomponents: [BAM business object dispatcher \(Managed components\)](#) and [BAM event dispatcher \(Managed components\)](#).

# Notification connector (Managed components)

## Settings

### HideSubcase

<b>Setting type</b>	Hot
<b>Type</b>	Boolean
<b>Description</b>	Hides the subcase

### ShowAllFolders

<b>Setting type</b>	Hot
<b>Type</b>	Boolean
<b>Description</b>	Displays all the folders

### ShowCount

<b>Setting type</b>	Hot
<b>Type</b>	Integer
<b>Description</b>	Displays the count

**ShowNotifications**

<b>Setting type</b>	Hot
<b>Type</b>	Boolean
<b>Description</b>	Displays notifications

**autoRefresh**

<b>Setting type</b>	Hot
<b>Type</b>	Boolean
<b>Description</b>	If true, refreshes the Inbox automatically

**autoclaim**

<b>Setting type</b>	Hot
<b>Type</b>	Boolean
<b>Description</b>	If true, claims automatically

**defaultDelivery**

<b>Setting type</b>	Hot
<b>Type</b>	String
<b>Description</b>	Default delivery to Mail or CPCINBOX if no target is specified in the request

**defaultDispatchTimeOut**

<b>Setting type</b>	Hot
<b>Type</b>	Integer
<b>Description</b>	Time after which incomplete tasks are reassigned to the next available recipient

**eventService**

<b>Setting type</b>	Hot
<b>Type</b>	String
<b>Description</b>	Distinguished name of the event service processor

**mailId**

<b>Setting type</b>	Hot
<b>Type</b>	String
<b>Description</b>	Default email ID that is used if no email ID is specified in the request

**mailService**

<b>Setting type</b>	Hot
<b>Type</b>	String
<b>Description</b>	Distinguished name of the email service processor

**mailText**

<b>Setting type</b>	Hot
<b>Type</b>	String
<b>Description</b>	Default text for the email if the target is specified as <code>mail</code> in the request

**noOfThreads**

<b>Setting type</b>	Hot
<b>Type</b>	Integer
<b>Description</b>	Number of threads internally used by the Notification Service Engine to dispatch messages to the Inbox or mailbox.

**refreshRate**

<b>Setting type</b>	Hot
<b>Type</b>	Integer
<b>Description</b>	Time interval (in seconds) between two successive autoRefresh operations

**urlPrefix**

<b>Setting type</b>	Hot
<b>Type</b>	String
<b>Description</b>	URL prefix for the task pages in the <code>http://&lt;server name&gt;</code> format

**usecustomstatemodel**

<b>Setting type</b>	Hot
<b>Type</b>	Boolean
<b>Description</b>	To use a custom state model

## Counters

**noOfMessagesInQueue**

<b>Counter type</b>	Value counter
<b>Value</b>	Number of messages in the queue at a specific time
<b>Hints</b>	If there are a large number of messages in the queue, OpenText recommends increasing the number of threads

**noOfMessagesDispatchedToInBox**

<b>Counter type</b>	Value counter
<b>Value</b>	Number of messages already dispatched to the Inbox since the last reset or restart of the processor
<b>Hints</b>	None

**noOfMessagesDispatchedToMailBox**

<b>Counter type</b>	Value counter
<b>Value</b>	Number of messages already dispatched to the user's mailbox since the last reset or restart of the processor
<b>Hints</b>	

**requestHandlingTimer**

<b>Counter type</b>	Event value counter
<b>Value</b>	Request handling duration
<b>Hints</b>	None

## Alerts

### **emailSOAPProcessorFailure**

<b>Severity level</b>	Error
<b>Message</b>	The Email SOAP processor is not functioning properly or has not yet been started.
<b>Description</b>	The SOAP processor {} is down or has not been started.

### **eventSOAPProcessorFailure**

<b>Severity level</b>	Error
<b>Message</b>	The Event SOAP processor is not functioning properly or has not been started.
<b>Description</b>	The SOAP processor {} is down or has not been started.

## Problems

### **databaseConnectionFailure**

<b>problemRaiseDescription</b>	The database server is not available. The database server may not be functioning, or the network may be encountering server connectivity problems. Ensure that the database server is functioning, and the network is communicating properly with the server.
<b>problemClearDescription</b>	Connection to the database server has been established.

## Subcomponents

The Notification Connector type has three sub-components: Notification Service Connection Pool, NotificationDispatcher, and TaskIdentifierIndexer.

## Notification dispatcher

### Settings

#### **maxConcurrentWorkers**

<b>Setting type</b>	Hot
---------------------	-----

<b>Type</b>	Integer
<b>Description</b>	Maximum concurrent threads in the thread pool

### ***maxPendingWorks***

<b>Setting type</b>	Hot
<b>Type</b>	Integer
<b>Description</b>	Maximum number of works that the dispatcher can queue up

### ***minConcurrentWorkers***

<b>Setting type</b>	Hot
<b>Type</b>	Integer
<b>Description</b>	Minimum concurrent worker threads in the thread pool

### ***monitorPeriod***

<b>Setting type</b>	Hot
<b>Type</b>	Long
<b>Description</b>	Interval period at which the dispatcher checks how many idle workers are running

## **Counters**

### ***currentNumWorkers***

<b>Counter type</b>	Value counter
<b>Value</b>	Number of current worker threads
<b>Hints</b>	None

### ***numActiveWorkers***

<b>Counter type</b>	Value counter
<b>Value</b>	Number of current active worker threads
<b>Hints</b>	None

### ***numIdleWorkers***

<b>Counter type</b>	Value counter
<b>Value</b>	Number of idle worker threads
<b>Hints</b>	None

### **Alerts**

None

### **Subcomponents**

None

## **OS Process (Managed components)**

This managed component type is the supertype for all operating system processes (currently SOAPPProcessorOSProcess, representing any process hosting SOAP processors, including the monitor and application server).

### **Settings**

None

### **Counters**

#### ***numOfNOMDocuments***

<b>Counter type</b>	Value counter
<b>Value</b>	Number of NOM documents that exist in this process.
<b>Hints</b>	An increasing value hints at a memory leak. Use the <code>setNOMLeakInfoBaseline</code> and <code>getNOMLeakInfo</code> operations for analysis.

#### ***numOfNOMNodes***

<b>Counter type</b>	Value counter
<b>Value</b>	Number of NOM nodes that exist in this process.
<b>Hints</b>	An increasing value hints at a memory leak. Use the <code>setNOMLeakInfoBaseline</code> and <code>getNOMLeakInfo</code> operations for analysis.

***totalCPUTime***

<b>Counter type</b>	Value counter
<b>Value</b>	Number of milliseconds the process runs per CPU
<b>Hints</b>	None

***residentMemoryUsage***

<b>Counter type</b>	Value counter
<b>Value</b>	Number of bytes the process has allocated in resident memory
<b>Hints</b>	None

***virtualMemoryUsage***

<b>Counter type</b>	Value counter
<b>Value</b>	Number of bytes the process has allocated in virtual memory
<b>Hints</b>	None

***totalNOMMemory***

<b>Counter type</b>	Value counter
<b>Value</b>	Number of bytes the process has allocated for NOM
<b>Hints</b>	None

***totalNOMNodesMemory***

<b>Counter type</b>	Value counter
<b>Value</b>	Total memory used by all XML nodes across all NOM document instances in this process
<b>Hints</b>	None

***NOMDocumentCreation***

<b>Counter type</b>	Event occurrence counter
<b>Value</b>	Frequency of NOM document creation
<b>Hints</b>	None

## Operations

### ***clearNOMLeakInfoBaseline***

<b>Description</b>	This operation clears the baseline on NOM objects that was set through the <code>setNOMLeakInfoBaseline</code> operation and takes away the performance penalty.
<b>Parameters</b>	None
<b>Return value</b>	None

### ***getNOMLeakInfo***

<b>Description</b>	After setting a baseline on the NOM objects through the <code>setNOMLeakInfoBaseline</code> operation, use the <code>getNOMLeakInfo</code> operation to obtain an overview of the NOM objects created since the baseline was set.
<b>Parameters</b>	
<b>newTreesThreshold</b>	A document with lesser new trees than this threshold is not reported.
<b>Return value</b>	XML string containing the NOM memory leak information.

### ***resetCounter***

<b>Description</b>	Resets the specified counter on the specified managed component; use for a component or counter.
<b>Parameters</b>	
<b>componentName</b>	Name of the component; use for all components.
<b>counterName</b>	Name of the counter; use for all counters.
<b>Return value</b>	None

### ***setCounterMAWLength***

<b>Description</b>	Sets the moving average window length (in milliseconds) of the given counters of the given components.
<b>Parameters</b>	
<b>componentName</b>	Name of the component. Use * for all components.
<b>counterName</b>	Name of the counter. Use * for all counters.

<b>windowLength</b>	Length of the moving average window in milliseconds.
<b>Return value</b>	None

### ***setCounterMAWSlots***

<b>Description</b>	Sets the number of slots into which the moving average window of the given counters of the given components must be subdivided.
<b>Parameters</b>	
<b>componentName</b>	Name of the component. Use * for all components.
<b>counterName</b>	Name of the counter. Use * for all counters.
<b>numOfSlots</b>	Number of slots in the moving average window.
<b>Return value</b>	None

### ***setNOMLeakInfoBaseline***

<b>Description</b>	This operation sets a baseline on the NOM documents and trees that exist at the moment of invocation. Use <code>getNOMLeakInfo</code> to retrieve the NOM objects that are created since the baseline creation. You must call <code>clearNOMLeakInfoBaseline</code> after completing the analysis because NOM baselining causes a performance penalty.
<b>Parameters</b>	None
<b>Return value</b>	None

## Alerts

### ***monitorStartErrorSyncupFailure***

<b>Severity level</b>	Error
<b>Message</b>	The AppWorks Platform (<instance name>) could not be started because of a failure to initialize State SyncUp. Ensure that Multicast is enabled and both the Port and IP Address specified are correct.
<b>Description</b>	This alert is issued when the monitor cannot be started because State SyncUp initialization fails.

### ***monitorStartErrorBootstrapFailure***

<b>Severity level</b>	Error
<b>Message</b>	The AppWorks Platform (<instance name>) could not be started because of an exception that occurred while loading the bootstrap information.
<b>Description</b>	This alert is issued when the monitor cannot be started because of an error in loading bootstrap information.

### ***monitorStartErrorLdapFailure***

<b>Severity level</b>	Error
<b>Message</b>	The AppWorks Platform (<instance name>) could not be started because of a failure to access the LDAP server. Ensure that the LDAP server is running. If yes, the <computer> could be slow. Add the bus.ldaprequest.timeout=<timeout> property in the wcp.properties file and specify the duration for the request timeout.
<b>Description</b>	This alert is issued when the monitor cannot be started because of an LDAP connectivity problem.

### ***monitorStartErrorEntryNotInLdap***

<b>Severity level</b>	Error
<b>Message</b>	The AppWorks Platform (<instance name>) {0} has not been registered on the LDAP server {1}, and therefore cannot be started.
<b>Description</b>	This alert is issued when the monitor entry is not present in LDAP.

### ***monitorStartErrorLdapProcessorFailure***

<b>Severity level</b>	Error
<b>Message</b>	The AppWorks Platform (<instance name>) could not be started because of a failure to contact the LDAP processor. Ensure that the LDAP SOAP processor is running.
<b>Description</b>	This alert is issued when the monitor cannot be started because the LDAP SOAP processor is not running.

### ***licenseExpired***

<b>Severity level</b>	Warning
<b>Message</b>	License expired {0} day(s) ago. Use the License manager to send the usage report and obtain a new license key.
<b>Description</b>	This alert is issued when the license validity expires.

### ***monitorStartErrorLicensekeyReadLdapFailure***

<b>Severity level</b>	Error
<b>Message</b>	The AppWorks Platform (<instance name>) could not be started because of a failure to read the license key from LDAP.
<b>Description</b>	This alert is issued when the monitor cannot be started because of a failure in reading the license key from LDAP.

### ***monitorStartErrorLicenseInvalid***

<b>Severity level</b>	Error
<b>Message</b>	The AppWorks Platform (<instance name>) could not be started due to an invalid license key. Enter the correct key.
<b>Description</b>	This alert is issued when the monitor cannot be started due to an invalid license.

### ***monitorStartErrorLicensekeyReadFailure***

<b>Severity level</b>	Error
<b>Message</b>	The AppWorks Platform (<instance name>) could not be started since the license key could not be read.
<b>Description</b>	This alert is issued when the monitor cannot be started because the license key cannot be read.

### ***monitorStartErrorExceptionOccured***

<b>Severity level</b>	Error
<b>Message</b>	The AppWorks Platform (<instance name>) could not be started owing to a catastrophic exception that occurred while starting the same.
<b>Description</b>	This alert is issued when the monitor cannot be started because of an exception.

### ***monitorStarted***

<b>Severity level</b>	Information
<b>Message</b>	AppWorks Platform (<instance name>) has been started. {0}
<b>Description</b>	This alert is issued when the OpenText AppWorks Platform (<instance name>) starts successfully.

### ***license\_Valid***

<b>Severity level</b>	Information
<b>Message</b>	The validity of the license key is {0} day(s).
<b>Description</b>	This alert is issued to inform of the license validity.

## **Problems**

None

## **Subcomponents**

A generic OS process has two subcomponents: LDAP Cache and License Monitor.

## **PIM query connector connection pool**

### **Settings**

#### ***optimiseQueryForCursor***

<b>Hot setting</b>	Yes
<b>Type</b>	Boolean
<b>Description</b>	Optimizing query to retrieve the first set of records using the cursor

#### ***queryCacheSize***

<b>Hot setting</b>	Yes
<b>Type</b>	Integer

<b>Description</b>	Size of the query cache
--------------------	-------------------------

**precedence**

<b>Hot setting</b>	Yes
<b>Type</b>	String
<b>Description</b>	Data or null precedence

**requestThreshold**

<b>Hot setting</b>	Yes
<b>Type</b>	Long
<b>Description</b>	Threshold value for each request

**minimumWriteConnections**

<b>Hot setting</b>	Yes
<b>Type</b>	Integer
<b>Description</b>	Number of minimum write connections

**cursorCacheRefreshInterval**

<b>Hot setting</b>	Yes
<b>Type</b>	Integer
<b>Description</b>	Refresh interval for the cursor cache

**reconnectionInterval**

<b>Hot setting</b>	Yes
<b>Type</b>	Long
<b>Description</b>	Interval between two attempts to reconnect to the database server

**reconnectionAttempts**

<b>Hot setting</b>	Yes
<b>Type</b>	Integer
<b>Description</b>	Number of attempts to reconnect to the database server

**cursorCacheSize**

<b>Hot setting</b>	Yes
<b>Type</b>	Integer
<b>Description</b>	Size of the cursor cache

**maximumWriteConnections**

<b>Hot setting</b>	Yes
<b>Type</b>	Integer
<b>Description</b>	Maximum number of write connections

**connectionPoolRefreshInterval**

<b>Hot setting</b>	Yes
<b>Type</b>	Integer
<b>Description</b>	Refresh interval for the connection pool

**queryCacheRefreshInterval**

<b>Hot setting</b>	Yes
<b>Type</b>	Integer
<b>Description</b>	Refresh interval for the query cache

**allowQueryLanguageFallback**

<b>Hot setting</b>	Yes
<b>Type</b>	String
<b>Description</b>	Switching query language from DBSQL to DBVIEW automatically

**conversionConfig**

<b>Hot setting</b>	Yes
<b>Type</b>	String
<b>Description</b>	XML structure containing data conversion information

**minimumReadConnections**

<b>Hot setting</b>	Yes
<b>Type</b>	Integer
<b>Description</b>	Number of minimum read connections

**multibyte**

<b>Hot setting</b>	Yes
<b>Type</b>	String
<b>Description</b>	Multibyte character support

**xsiNilForNullData**

<b>Hot setting</b>	Yes
<b>Type</b>	String
<b>Description</b>	Setting <code>xsi:nil</code> for absence of data along with <code>null=true</code>

**xmlEncoding**

<b>Hot setting</b>	Yes
<b>Type</b>	String
<b>Description</b>	Support for XML special characters

**maximumReadConnections**

<b>Hot setting</b>	Yes
<b>Type</b>	Integer
<b>Description</b>	Number of maximum read connections

**defaultCursorSize**

<b>Hot setting</b>	Yes
<b>Type</b>	Integer
<b>Description</b>	Default size of the cursor

**auditInfo**

<b>Hot setting</b>	Yes
<b>Type</b>	String
<b>Description</b>	XML structure containing audit information

**multithreaded**

<b>Hot setting</b>	Yes
<b>Type</b>	String
<b>Description</b>	Multithread provider support

**datasource**

<b>Hot setting</b>	Yes
<b>Type</b>	String
<b>Description</b>	XML structure containing datasource information

## Counters

**numOfCachedReadConnections**

<b>Counter type</b>	Value counter
<b>Value</b>	Number of business processes aborted because of unhandled errors since the last reset or restart of the processor
<b>Hints</b>	None

**numOfCachedWriteConnections**

<b>Counter type</b>	Value counter
<b>Value</b>	Number of business processes aborted because of unhandled errors since the last reset or restart of the processor
<b>Hints</b>	None

**numberOfCachedQueries**

<b>Counter type</b>	Value counter
<b>Value</b>	Number of business processes aborted because of unhandled errors since the last reset or restart of the processor
<b>Hints</b>	None

**numberOfOpenCursors**

<b>Counter type</b>	Value counter
<b>Value</b>	Number of business processes aborted because of unhandled errors since the last reset or restart of the processor
<b>Hints</b>	None

**readConnectionUsageDuration**

<b>Counter type</b>	Event value counter
<b>Value</b>	Number of business processes aborted because of unhandled errors since the last reset or restart of the processor
<b>Hints</b>	None

**readConnectionWait**

<b>Counter type</b>	Event value counter
<b>Value</b>	Number of business processes aborted because of unhandled errors since the last reset or restart of the processor
<b>Hints</b>	None

**writeConnectionUsageDuration**

<b>Counter type</b>	Event value counter
<b>Value</b>	Number of business processes aborted because of unhandled errors since the last reset or restart of the processor
<b>Hints</b>	None

**writeConnectionWait**

<b>Counter type</b>	Event Value counter
<b>Value</b>	Number of business processes aborted because of unhandled errors since the last reset or restart of the processor
<b>Hints</b>	None

## Alerts

### databaseConnectionFailure

<b>Severity level</b>	Error
<b>Message</b>	The database server may not be functioning, or the network may be encountering server connectivity problems.
<b>Description</b>	This alert is issued when connection to the database server is lost.

### databaseConnectionEstablished

<b>Severity level</b>	Info
<b>Message</b>	Connection to the database server has been established.
<b>Description</b>	This alert is issued when connection to the database server is successful.

## Problems

### DBConnectionFailure

<b>problemRaiseDescription</b>	The database server is not available. The database server may not be functioning, or the network may be encountering server connectivity problems. Ensure that the database server is functioning, and the network is communicating properly with the server.
<b>problemClearDescription</b>	Connection to the database server has been established.

## Subcomponents

None

# AppWorks Platform SAML protocol handler (Managed components)

## Settings

### webSoapValidator

<b>Setting type</b>	Hot
<b>Type</b>	String
<b>Description</b>	Web validator implementation class to add to SSOCordysValidator

## Counters

### accessDenied

<b>Counter type</b>	Event occurrence counter
<b>Event</b>	
<b>Value</b>	Frequency of invalid credentials occurrences
<b>Hints</b>	None

### authenticateDuration

<b>Counter type</b>	Event value counter
<b>Event</b>	
<b>Value</b>	Frequency and duration of authentication
<b>Hints</b>	None

## Alerts

None

## Problems

None

## Subcomponents

None

# Request monitor (Managed components)

## Settings

None

## Counters

### *requestHandlingTimer*

<b>Counter type</b>	Timer event value counter
<b>Event</b>	Frequency and duration of the request handling
<b>Hints</b>	None

### *timeouts*

<b>Counter type</b>	Timer event value counter
<b>Event</b>	Frequency and duration of the timed out requests
<b>Hints</b>	None

## Alerts

None

## Problems

None

## Subcomponents

None

# Rule engine (Managed components)

## Settings

### maxDispatcherQueueSize

<b>Setting type</b>	Hot
<b>Type</b>	Long
<b>Description</b>	Maximum number of actions that can be in the dispatcher queue.

### monitorRequestAndResponse

<b>Setting type</b>	Hot
<b>Type</b>	Boolean
<b>Description</b>	When set to true, persists the request and response of external rule actions such as Web service invocations for notification and other Web services to the rule monitoring database. Otherwise, request and response information is not persisted.  This property is effective only when the <code>ruleAudit</code> property is set to true.

### publishAudit

<b>Setting type</b>	Hot
<b>Type</b>	Boolean
<b>Description</b>	Publishes the audit data to the rule engine host

### ruleAudit

<b>Setting type</b>	Hot
<b>Type</b>	Boolean
<b>Description</b>	When set to true, enables the rule engine to store monitoring information in the Rule Monitor database.

**ruleMonitorListener**

<b>Setting type</b>	Hot
<b>Type</b>	String
<b>Description</b>	Hooks on to events raised during the rule execution lifecycle. Sets the fully qualified class name of a concrete class that implements the <code>com.cordys.bre.rim.RuleMonitor</code> interface.

**threadPoolSize**

<b>Setting type</b>	Hot
<b>Type</b>	Integer
<b>Description</b>	Number of rule action threads to be present

**Counters****dispatcherQueueSize**

<b>Counter type</b>	Value counter
<b>Event</b>	Dispatcher queue size
<b>Hints</b>	Specifies number of actions in the dispatcher

**ruleActionExecutionRate**

<b>Counter type</b>	Event value counter
<b>Event</b>	A timer event value counter that returns the rate at which external actions of AppWorks Platform rules are executed
<b>Hints</b>	None

**ruleExecutionRate**

<b>Counter type</b>	Event value counter
<b>Value</b>	A timer event value counter that returns the rate at which AppWorks Platform rules are executed
<b>Hints</b>	None

**Alerts**

None

## Operations

### resetCounter

<b>Description</b>	Resets the Rule Engine counter identified by a specific name. Use * to reset all rule engine counters
<b>Parameters</b>	counterName
<b>Return value</b>	String

## Subcomponents

The Rule Engine type has two subcomponents: Decision table cache and UriBinding cache.

## Rule management

### Settings

None

### Counters

#### requestHandlingTimer

<b>Counter type</b>	Event value counter
<b>Event</b>	Frequency of the request handling
<b>Hints</b>	None

### Alerts

None

### Problems

#### databaseConnectionFailure

<b>problemRaiseDescription</b>	The database server is not available. The database server may not be functioning, or the network may be encountering server connectivity problems. Ensure that the database server is functioning, and the network is communicating properly with the server.
<b>problemClearDescription</b>	Connection to the database server has been established.

## Subcomponents

The Rule Management has two subcomponents: Rule Engine and Rule Repository ConnectionPool.

## Scheduler engine (Managed components)

### Settings

#### TriggerMissedSchedule

<b>Setting type</b>	Hot
<b>Type</b>	String
<b>Description</b>	Setting the Trigger Missed Schedule value. The values are: all, none, and one.

### Counters

None

### Alerts

None

### Problems

None

## Subcomponents

None

## Scheduling

### Settings

None

## Counters

<b>Counter type</b>	Event Value counter
<b>Event</b>	Frequency of the request handling of this application connector
<b>Hints</b>	None

## Alerts

None

## Problems

<b>problemRaiseDescription</b>	The database server is not available. The database server may not be functioning, or the network may be encountering server connectivity problems. Ensure that the database server is functioning, and the network is communicating properly with the server.
<b>problemClearDescription</b>	Connection to the database server has been established.

## Subcomponents

Scheduling has two subcomponents: Scheduler DBConnectionPool and Scheduler Engine.

# SearchConnector (Managed components)

## Settings

None

## Counters

### *numOfDocsIndexed*

<b>Counter type</b>	Event Value counter
<b>Event</b>	A document is indexed
<b>Value</b>	Number of documents indexed; incremental indexing of a document is taken as indexing of a new document
<b>Hints</b>	None

### ***numOfSearches***

<b>Counter type</b>	Event Value counter
<b>Event</b>	A search is done that is not yet cached
<b>Value</b>	Number of searches made

### **Alerts**

None

### **Problems**

None

### **Subcomponents**

None

## **Security administration (Managed components)**

### **Settings**

None

### **Counters**

#### ***requestHandlingTimer***

<b>Counter type</b>	Event Value counter
<b>Event</b>	Frequency of the request handling of this application connector
<b>Hints</b>	None

### **Alerts**

None

## Problems

### databaseConnectionFailure

<b>problemRaiseDescription</b>	The database server is not available. The database server may not be functioning, or the network may be encountering server connectivity problems. Ensure that the database server is functioning, and the network is communicating properly with the server.
<b>problemClearDescription</b>	Connection to the database server has been established.

## Subcomponents

Security Administration has one subcomponent: XDSRepository.

## Service container (Managed components)

### Processor - Settings

#### abortTime

<b>Setting type</b>	Hot
<b>Type</b>	Integer
<b>Description</b>	Time in seconds to abort SOAP transactions; a maximum of 5 seconds is considered

#### aclCacheSize

<b>Setting type</b>	Hot
<b>Type</b>	Integer
<b>Description</b>	Maximum number of user's ACL entries in the cache

#### cancelReplyInterval

<b>Setting type</b>	Hot
<b>Type</b>	Long
<b>Description</b>	Interval after which a cancel response is sent to the requestor; can use to restart the processor (to get rid of deadlocks that are currently not implemented by any client)

**gracefulCompleteTime**

<b>Setting type</b>	Hot
<b>Readable</b>	Yes
<b>Type</b>	Integer
<b>Description</b>	Time in seconds to complete SOAP transactions gracefully; a maximum of 15 seconds is considered

**Info properties****status**

<b>Type</b>	String
<b>Description</b>	Status of the service container

**Counters****SOAPRequests**

<b>Counter type</b>	Event value counter
<b>Event</b>	A SOAP request handled by the service container
<b>Value</b>	Duration and frequency of the SOAP requests handled by the SOAP processor
<b>Hints</b>	None

**identityValidationDuration**

<b>Counter type</b>	Event value counter
<b>Event</b>	
<b>Value</b>	Frequency and duration of identity validation requests.
<b>Hints</b>	None

**invalidIdentity**

<b>Counter type</b>	Event Occurrence counters
<b>Event</b>	
<b>Value</b>	Total number of invalid identity validation occurrences
<b>Hints</b>	None

## Alerts

None

## Problems

None

## Subcomponents

A service container has the following subcomponents: [Request monitor \(Managed components\)](#) and [SOAP connector \(Managed components\)](#).

# Service container OS process (Managed components)

This managed component type is a subtype of OSProcess that represents the Operating System process of a process hosting SOAP processors. This type is used for the AppWorks Platform Monitor and processes hosting regular SOAP processors.

## Settings

The SOAPProcessorOSProcess currently does not add settings to the ones defined on the OSProcess supertype.

## Counters

The SOAPProcessorOSProcess currently does not add counters to the ones defined on the OSProcess supertype.

## Alerts

None

## Problems

None

## Subcomponents

The SOAPProcessorOSProcess managed component type adds one subcomponent type to the set defined by the OSProcess supertype: Processor.

Other subcomponents are components of type License, LDAPCache, and State SyncUp.

# SOAP connector (Managed components)

## Settings

### **defaultTimeout**

<b>Setting type</b>	Hot
<b>Readable</b>	Yes
<b>Type</b>	Long
<b>Description</b>	Default timeout used when the call to sendWait does not specify a timeout

## Counters

### **requestHandlingTimer**

<b>Counter type</b>	Event Value counter
<b>Event</b>	A request that was received by this connector has been handled
<b>Value</b>	Frequency and duration of the request handling
<b>Hints</b>	None

### **sendAndWaitTimer**

<b>Counter type</b>	Event Value counter
<b>Event</b>	A request that was received by this connector has been handled
<b>Value</b>	Frequency and duration of outgoing requests
<b>Hints</b>	None

## Alerts

None

## Problems

None

## Subcomponents

[Middleware wrapper \(Managed components\)](#)

## SQLApp connector (Managed components)

This managed component type represents an application connector to connect to any SQL database. This type is used for three different application connector implementations:

- JDBC application connector
- OLDB application connector
- Baan application connector

### Settings

None

### Counters

None

### Alerts

None

### Problems

None

### Subcomponents

The SQLAppConnector managed component type adds one subcomponent type to the set defined by the AppConnector supertype: DBConnectionPool.

## SSO connector (Managed components)

### Settings

#### provider

<b>Setting type</b>	Hot
<b>Readable</b>	Yes
<b>Type</b>	String
<b>Description</b>	SSO Authentication provider implementation class

## Counters

### **requestHandlingTimer**

<b>Counter type</b>	Event Value counter
<b>Event</b>	A request that was received by this connector has been handled
<b>Value</b>	Frequency and duration of the request handling of this application connector
<b>Hints</b>	None

## Alerts

None

## Problems

None

## Subcomponents

A SSO Connector type has one sub-component type: [AppWorks Platform SAML protocol handler \(Managed components\)](#).

## State Syncup (Managed components)

### Settings

None

## Counters

None

## Alerts

### **syncupMemberRemoved**

<b>Severity level</b>	Warning
<b>Message</b>	The member(s) {0} are no longer a part of the ring.
<b>Description</b>	This alert is issued when a member is removed from the syncup ring.

### ***newSyncupMember***

<b>Severity level</b>	Information
<b>Message</b>	A new member {0} has joined the ring.
<b>Description</b>	This alert is issued when a new member joins the ring.

### ***Problems***

None

### ***Subcomponents***

None

## **Tag server (Managed components)**

### ***Settings***

None

### ***Counters***

#### ***requestHandlingTimer***

<b>Counter type</b>	Timer event value counter
<b>Event</b>	Frequency and duration of the request handling of this application connector
<b>Hints</b>	None

### ***Alerts***

None

### ***Problems***

None

### ***Subcomponents***

The Tag Server type has two sub-components: ResourceDBConnectionPool and XDSRepository.

# TaskIdentifierIndexer

## Settings

### *maxConcurrentWorkers*

<b>Setting type</b>	Hot
<b>Type</b>	Integer
<b>Description</b>	Maximum concurrent threads in the thread pool

### *maxPendingWorks*

<b>Setting type</b>	Hot
<b>Type</b>	Integer
<b>Description</b>	Maximum number of works that the dispatcher can queue up

### *minConcurrentWorkers*

<b>Setting type</b>	Hot
<b>Type</b>	Integer
<b>Description</b>	Minimum concurrent threads in the thread pool

### *monitorPeriod*

<b>Setting type</b>	Hot
<b>Type</b>	Long
<b>Description</b>	Interval period at which the dispatcher checks how many idle workers are running

## Counters

### *currentNumWorkers*

<b>Counter type</b>	Value counter
<b>Value</b>	Number of current worker threads
<b>Hints</b>	None

### ***numActiveWorkers***

<b>Counter type</b>	Value counter
<b>Value</b>	Number of current active worker threads
<b>Hints</b>	None

### ***numIdleWorkers***

<b>Counter type</b>	Value counter
<b>Value</b>	Number of idle worker threads
<b>Hints</b>	None

### **Alerts**

None

### **Subcomponents**

None

## **UDDI connector (Managed components)**

### **Settings**

#### ***isProxyEnabled***

<b>Setting type</b>	Hot
<b>Readable</b>	Yes
<b>Type</b>	Boolean
<b>Description</b>	Verifies whether the proxy is enabled

#### ***proxyHost***

<b>Setting type</b>	Hot
<b>Readable</b>	Yes
<b>Type</b>	String
<b>Description</b>	HTTP proxy host

***proxyPort***

<b>Setting type</b>	Hot
<b>Readable</b>	Yes
<b>Type</b>	Integer
<b>Description</b>	HTTP proxy port

***username***

<b>Setting type</b>	Hot
<b>Readable</b>	Yes
<b>Type</b>	String
<b>Description</b>	HTTP proxy username

***password***

<b>Setting type</b>	Hot
<b>Readable</b>	Yes
<b>Type</b>	String
<b>Description</b>	HTTP proxy password

**Counters*****publishRequestTimer***

<b>Counter type</b>	Timer Event Value counter
<b>Event</b>	
<b>Value</b>	Frequency and duration of the publish request processed by the UDDI Connector
<b>Hints</b>	None

***requestHandlingTimer***

<b>Counter type</b>	Timer Event Value counter
<b>Event</b>	
<b>Value</b>	Frequency and duration of the request handling of this application connector

<b>Hints</b>	None
--------------	------

## Alerts

None

## Problems

None

## Subcomponents

None

# Upload gateway (Managed components)

## Settings

None

## Counters

### *MaxFileSizeUploaded*

<b>Counter type</b>	Value counter
<b>Event</b>	
<b>Value</b>	Maximum file size uploaded
<b>Hints</b>	None

### *UploadFailures*

<b>Counter type</b>	Event Occurrence counter
<b>Event</b>	
<b>Value</b>	Total upload failures
<b>Hints</b>	None

### ***UploadRequestProcessingTime***

<b>Counter type</b>	Event Value counter
<b>Event</b>	
<b>Value</b>	Total time taken for processing upload requests
<b>Hints</b>	None

### ***requestHandlingDuration***

<b>Counter type</b>	Event Value counter
<b>Event</b>	
<b>Value</b>	Duration and frequency of request handling
<b>Hints</b>	None

### ***uploadFileContent***

<b>Counter type</b>	Value counter
<b>Event</b>	
<b>Value</b>	Number of upload requests using FilePathRequest
<b>Hints</b>	None

### ***uploadFilePath***

<b>Counter type</b>	Value counter
<b>Event</b>	
<b>Value</b>	Number of upload requests using FilePathRequest
<b>Hints</b>	None

### ***uploadSize***

<b>Counter type</b>	Event Value counter
<b>Event</b>	
<b>Value</b>	Upload size
<b>Hints</b>	None

### ***uploadSuccess***

<b>Counter type</b>	Event Occurrence counter
<b>Event</b>	
<b>Value</b>	Total upload success
<b>Hints</b>	None

## **Alerts**

### ***gatewayErrorLdapConnectionFailure***

<b>Severity</b>	Error
<b>Message</b>	Gateway failed to process the request due to failure to establish connection with the LDAP service container.
<b>Description</b>	This alert is issued when the gateway fails to process the request because the LDAP server connection fails.

### ***gatewayErrorSocketFailure***

<b>Severity</b>	Error
<b>Message</b>	Gateway failed to process the request due to failure to open the socket connection.
<b>Description</b>	This alert is issued when the gateway encounters a socket problem and fails to process the request.

## **Problems**

None

## **Subcomponents**

None

# UriBinding cache (Managed components)

## Settings

### **cacheSize**

<b>Setting type</b>	Hot
<b>Readable</b>	Yes
<b>Type</b>	Integer
<b>Description</b>	Cache size

## Counters

### **noOfCacheHits**

<b>Counter type</b>	Event occurrence counter
<b>Event</b>	Number Of cache hits
<b>Hints</b>	None

### **noOfCacheMisses**

<b>Counter type</b>	Event Value counter
<b>Event</b>	Number Of cache misses
<b>Hints</b>	None

### **noOfEmptyObjectsInCache**

<b>Counter type</b>	Value counter
<b>Value</b>	Number of empty objects in the cache
<b>Hints</b>	None

### **noOfObjectsInCache**

<b>Counter type</b>	Value counter
<b>Value</b>	Number of objects in the cache
<b>Hints</b>	None

## Alerts

None

## Problems

None

## Subcomponents

None

# Web application (Managed components)

This managed type is the super type for all Web applications. A Web application is a plug-in in the Web gateway.

## Settings

None

## Counters

### requestHandlingDuration

<b>Counter type</b>	Event Value counter
<b>Event</b>	Frequency of the requests handled by the Web gateway
<b>Value</b>	Duration and frequency of the request handling
<b>Hints</b>	None

## Alerts

### gatewayErrorLdapConnectionFailure

<b>Severity</b>	Error
<b>Message</b>	Gateway failed to process the request due to failure to establish connection with the LDAP service container.
<b>Description</b>	This alert is issued when the gateway fails to process the request because the LDAP server connection fails.

### **gatewayErrorSocketFailure**

<b>Severity</b>	Error
<b>Message</b>	Gateway failed to process the request due to failure to open the socket connection.
<b>Description</b>	This alert is issued when the gateway encounters a socket problem and fails to process the request.

## Problems

None

## Subcomponents

None

# WebApplication (Managed components)

This managed type is the super type for all Web applications. A Web application is a plug-in in the Web gateway.

## Settings

None

## Counters

### **requestHandlingDuration**

<b>Counter type</b>	Timer event value counter
<b>Event</b>	A request was handled by this web application
<b>Value</b>	The request handling duration in milliseconds
<b>Hints</b>	None

## Alerts

None

## Problems

None

## Subcomponents

None

# WebGateway (Managed components)

## Settings

None

## Counters

### asynchronousRequest

<b>Counter type</b>	Event occurrence counter
<b>Event</b>	An asynchronous request was handled by the Web gateway
<b>Hints</b>	None

### requestHandlingDuration

<b>Counter type</b>	Timer event value counter
<b>Event</b>	Frequency of the requests handled by the Web gateway
<b>Value</b>	Request handling duration
<b>Hints</b>	None

### synchronousRequest

<b>Counter type</b>	Event occurrence counter
<b>Event</b>	A synchronous request was handled by the Web gateway
<b>Hints</b>	None

## Alerts

None

## Problems

None

## Subcomponents

The WebGateway managed component type adds two subcomponent types: WebApplication and BusGateway.

A WebApplication is an abstract super-type with the following concrete subtypes:

- AuthenticateWebApplication
- GatewayWebApplication
- WSDLGatewayWebApplication
- DownloadGatewayWebApplication
- UploadGatewayWebApplication

## WebLogger connector

### Settings

None

### Counters

#### **requestHandlingTimer**

<b>Counter type</b>	Event Value counter
<b>Event</b>	Frequency of the request handling of this application connector
<b>Hints</b>	None

### Alerts

None

## Subcomponents

None

# WS-AppServer (Managed components)

## Settings

### **auditEnabled**

<b>Setting type</b>	Hot
<b>Type</b>	String
<b>Description</b>	

### **autoCleanup**

<b>Setting type</b>	Hot
<b>Type</b>	Boolean
<b>Description</b>	

### **customInitializer**

<b>Setting type</b>	Hot
<b>Type</b>	String
<b>Description</b>	

### **initializeDB**

<b>Setting type</b>	Hot
<b>Type</b>	Boolean
<b>Description</b>	

### **initializeEIS**

<b>Setting type</b>	Hot
<b>Type</b>	Boolean
<b>Description</b>	

### **ruleEnabled**

<b>Setting type</b>	Hot
---------------------	-----

<b>Type</b>	Boolean
<b>Description</b>	

## Counters

### noOfActiveTransactions

<b>Counter type</b>	Value counter
<b>Value</b>	Number of active transactions available with the system at a specific time.
<b>Hints</b>	Higher values indicate a higher load on the system. Clustering can distribute the load among multiple WS-AppServer processors.

### requestHandlingTimer

<b>Counter type</b>	Event Value counter
<b>Value</b>	Frequency and duration of the request handling of this application connector
<b>Hints</b>	None

## Alerts

### dbConnectionFailure

<b>Severity level</b>	Error
<b>Message</b>	The database server may not be functioning, or the network may be encountering server connectivity problems.
<b>Description</b>	This alert is issued when the processor does not start because a database connectivity problem exists.

### stateSynchUpFailure

<b>Severity level</b>	Error
<b>Message</b>	State synch up is not configured, state synch up is required to enable Orchestrator cache synchronization.
<b>Description</b>	This alert is issued because there are problems with the state synchup configuration.

**missingTagAndNamespaceInClass**

<b>Severity level</b>	Error
<b>Message</b>	Ensure that a tag and namespace {0} for the required class is defined in the class registry.
<b>Description</b>	This alert is issued because the missing tag and namespace for the required class is not defined in the class registry.

**MissingClassDefinition**

<b>Severity level</b>	Error
<b>Message</b>	Class {0} not found.
<b>Description</b>	This alert is issued because the class definition is not found.

## Problems

**databaseConnectionFailure**

<b>problemRaiseDescription</b>	The database server is not available. The database server may not be functioning, or the network may be encountering server connectivity problems. Ensure that the database server is functioning, and the network is communicating properly with the server.
<b>problemClearDescription</b>	Connection to the database server has been established.

## Subcomponents

The WS-AppServer type has two subcomponents: WS-AppServer SOAP Client and WS-AppServer Connection Pool.

## WS-AppServer SOAP Client

### Settings

**defaultTimeout**

<b>Setting type</b>	Hot
<b>Type</b>	Long
<b>Description</b>	Default timeout used when the call to sendWait does not specify a timeout

## Counters

### **requestHandlingTimer**

<b>Counter type</b>	Event Value counter
<b>Value</b>	Frequency and duration of the request handling
<b>Hints</b>	None

### **sendAndWaitTimer**

<b>Counter type</b>	Event Value counter
<b>Value</b>	Frequency and duration of outgoing requests
<b>Hints</b>	None

## Alerts

None

## Problems

None

## Subcomponents

The WS-AppServer SOAP Client has one subcomponent: Middleware wrapper.

# WSDLGateway Web application (Managed components)

## Settings

The WSDLGatewayWebApplication currently does not add settings to the ones defined on the GatewayWebApplication supertype.

## Counters

The WSDLGatewayWebApplication adds the following counters to the ones defined on the GatewayWebApplication supertype:

**wsdlSize**

<b>Counter type</b>	Event value counter
<b>Event</b>	A WSDL document was downloaded
<b>Value</b>	Size of the downloaded WSDL document
<b>Hints</b>	None

**Alerts**

None

**Problems**

None

**Subcomponents**

The WSDLGatewayWebApplication managed component currently does not add any sub-components to the set defined by the GatewayWebApplication supertype.

## XDS Synchup (Managed components)

**Settings*****ipAddress***

<b>Setting type</b>	Hot
<b>Readable</b>	Yes
<b>Type</b>	String
<b>Description</b>	ipAddress

***portNo***

<b>Setting Type</b>	Hot
<b>Readable</b>	Yes
<b>Type</b>	String
<b>Description</b>	portNo

## Counters

None

## Alerts

None

## Subcomponents

None

# XDSRepository (Managed components)

## Settings

None

## Counters

### *noOfDBQuery*

<b>Counter type</b>	Event Value counter
<b>Event</b>	Number of database queries
<b>Hints</b>	None

### *noOfDbUpdate*

<b>Counter type</b>	Event Value counter
<b>Event</b>	Number of database updates
<b>Hints</b>	None

## Alerts

None

## Problems

None

## Subcomponents

XDSRepository has five subcomponents: BranchCache, CompressionSettings, DocumentCache, CWSDBConnectionPool, and XDS\_SyncUp.

## XForms application connector (Managed components)

### Settings

#### *expiryType*

<b>Setting type</b>	Hot
<b>Readable</b>	Yes
<b>Type</b>	Long
<b>Description</b>	Set the content expiry type to one of the following values: <ul style="list-style-type: none"><li>■ 1: Content expires immediately</li><li>■ 2: Content expires after the number of days entered in expiryValue.</li><li>■ 3: Content expires on the date mentioned in expiryValue. Enter the date in milliseconds.</li></ul>

#### *expiryValue*

<b>Setting type</b>	Hot
<b>Readable</b>	Yes
<b>Type</b>	Long
<b>Description</b>	Set the content expiry value. If the value in expiryType is: <ul style="list-style-type: none"><li>■ 1: Enter 0 (if the value is non-zero, it is not considered).</li><li>■ 2: Enter the number of days after which the content expires.</li><li>■ 3: Enter the expiry date in milliseconds.</li></ul>

#### *cacheSize*

<b>Setting type</b>	Hot
<b>Readable</b>	Yes
<b>Type</b>	Long
<b>Description</b>	Maximum number of XForms that can be stored in the XForms cache

## Counters

### *currentCacheSize*

<b>Counter type</b>	Value counter
<b>Value</b>	Current cache size of the XForms Engine
<b>Hints</b>	None

### *gettingObjectFromXFormsCache*

<b>Counter type</b>	Event Value counter
<b>Value</b>	Frequency and duration taken to retrieve the object either from the cache or to generate the object
<b>Hints</b>	None

### *requestHandlingTimer*

<b>Counter type</b>	Event Value counter
<b>Value</b>	Frequency and duration of the request handling of this application connector
<b>Hints</b>	None

## Alerts

None

## Problems

None

## Subcomponents

None

# XMLStore connector (Managed components)

## Settings

None

## Counters

### requestHandlingTimer

<b>Counter type</b>	Timer event value counter
<b>Event</b>	Frequency and duration of the request handling of this application connector
<b>Hints</b>	None

## Alerts

None

## Problems

None

## Subcomponents

The XMLStoreAppConnector type has one subcomponent: XDSRepository.

# Chapter 30

# Monitoring managed components

Monitoring the managed components of AppWorks Platform involves the following tasks:

- [Changing managed component settings](#)
- [Invoking managed component operations](#)
- [Monitoring managed component counters](#)
- [Receiving managed component notifications](#)
- [Monitoring heap memory usage of service containers](#)

For information about the values and settings of managed components of AppWorks Platform, see [Managed component types](#).

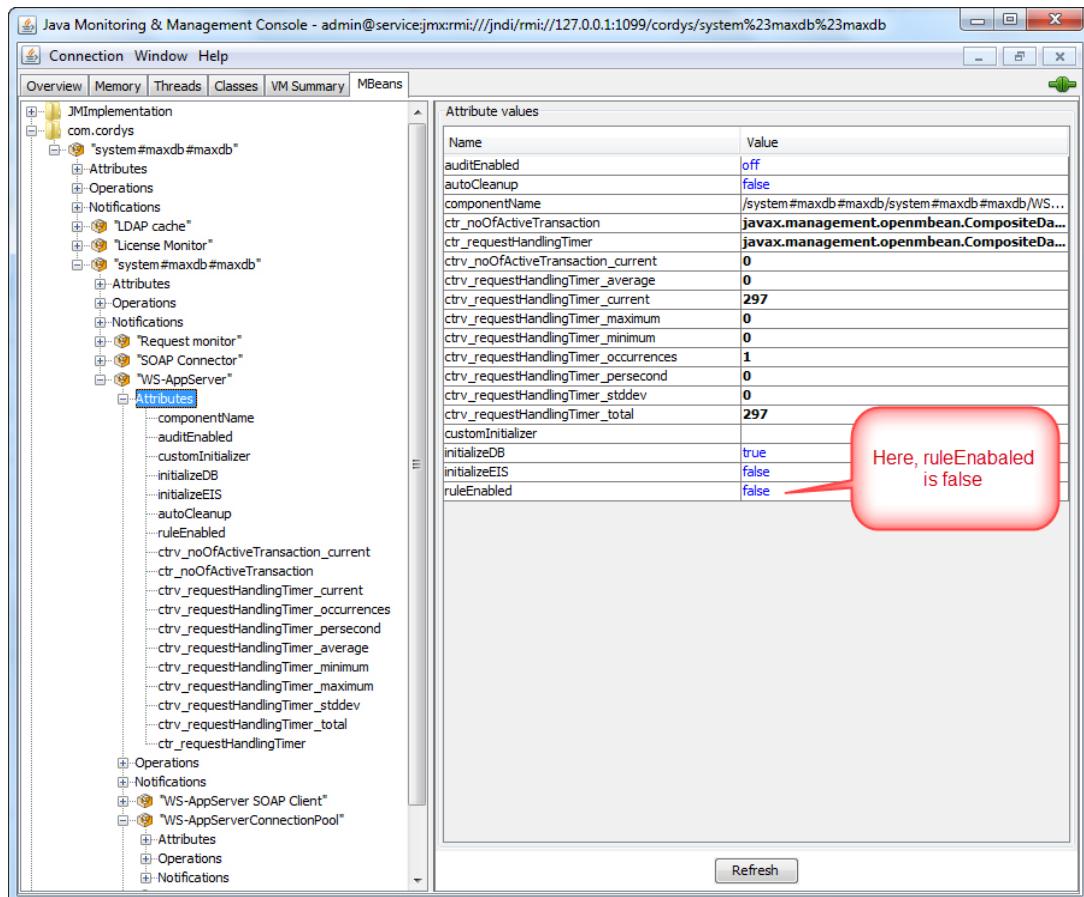
## Changing managed component settings

Consider that an administrator must change the settings of a managed component of AppWorks Platform such as the WS-AppServer service container. By default, the `ruleEnabled` property of the service container is disabled.

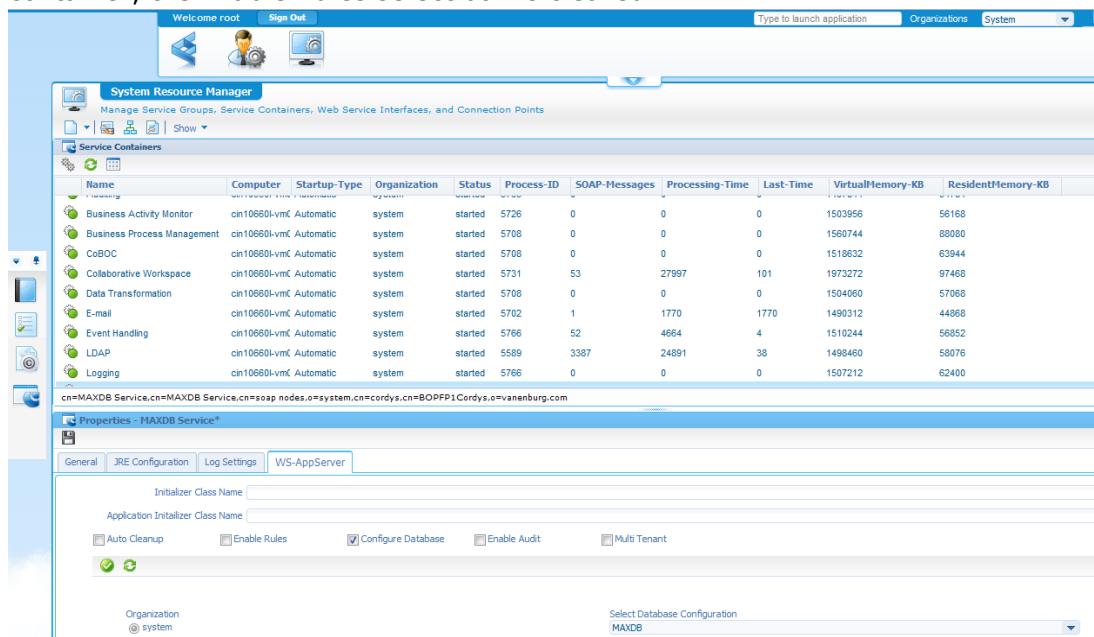
**Note:** Administrators can only change the hot settings that appear in blue. Restarting the service container is not required because the changes are applied immediately.

### To enable the `ruleEnabled` property through JConsole:

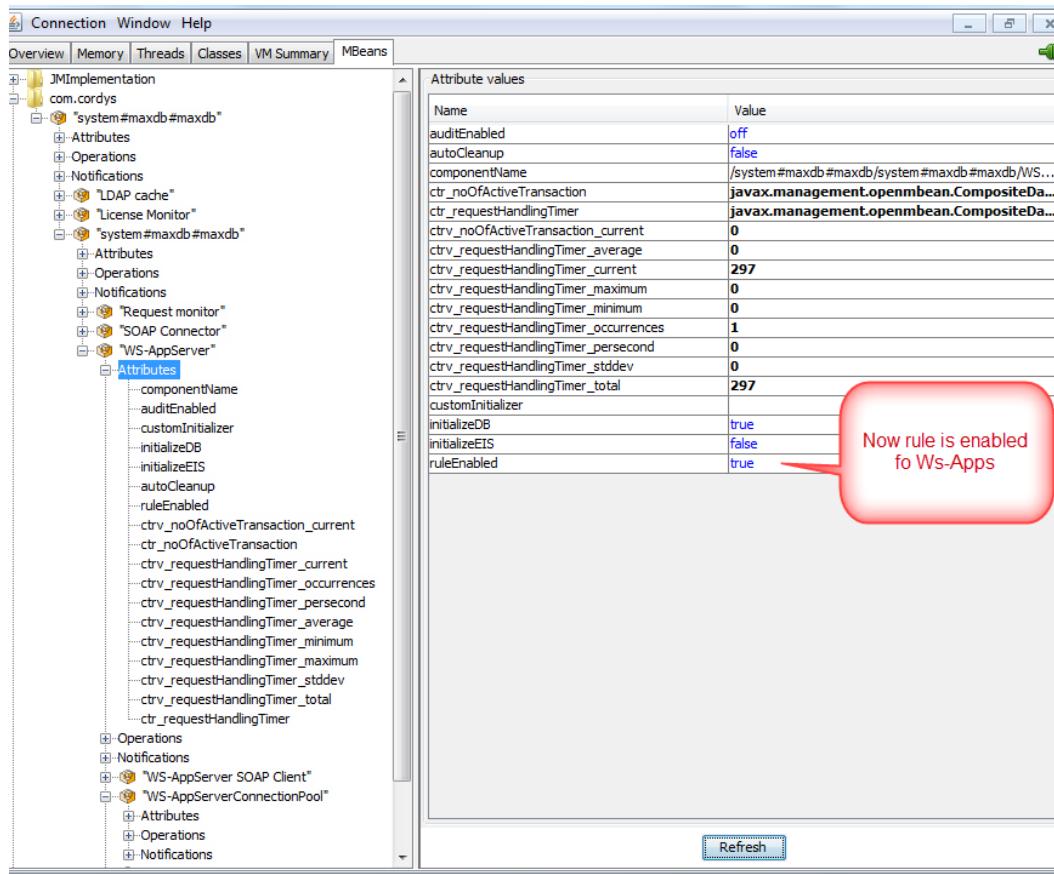
1. Connect to [JConsole](#) using the JMX address URL of the WS-AppServer service container.  
For example:  
`service:jmx:rmi://jndi/rmi://<Host Name>/cordys/system%23maxdb%23maxdb`
2. Click **MBeans**, expand **com.cordys**, and then click **Attributes** under the WS-AppServer service container.  
`ruleEnabled` is one of the hot settings of the WS-AppServer service container and is set to `false` by default.



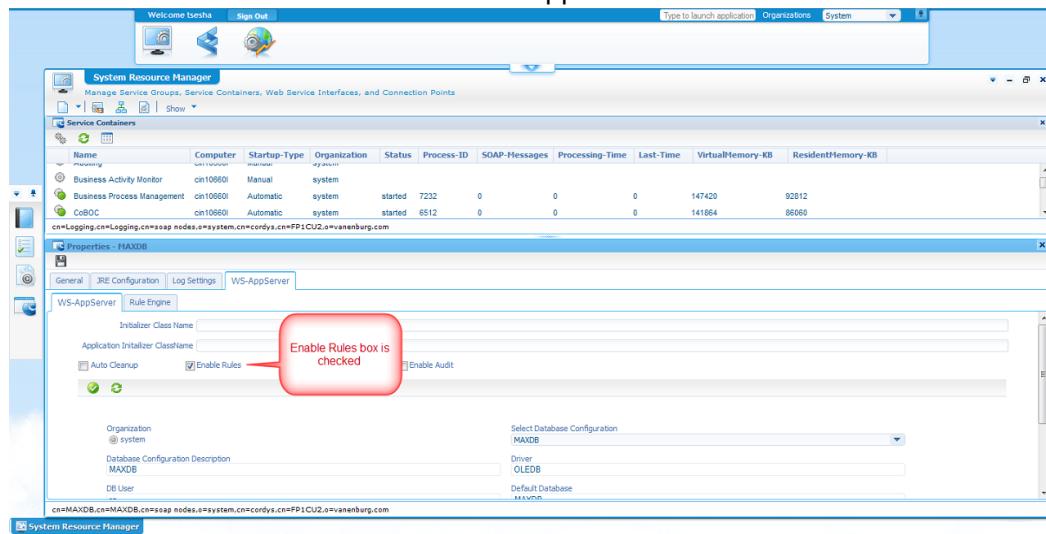
Also, in System Resource Manager, under the properties of WS-AppServer service container, the Enable Rules select box is cleared.



3. Change the value of the ruleEnabled attribute to true.



The Enable Rules check box on the WS-AppServer tab is selected.



## Invoking managed component operations

Consider that an administrator must invoke the operations of a managed component of AppWorks Platform such as the WS-AppServer service container to find a memory leak. The following operations of the service container can identify the memory leaks in Java code:

- `setNOMLeakInfoBaseline`: Sets a baseline for the number of existing NOM nodes.
- `getNOMLeakInfo`: Retrieves the information related to the leaking NOM nodes since the last baseline.

**Note:** The `resetCounter` operation resets the counters with default values.

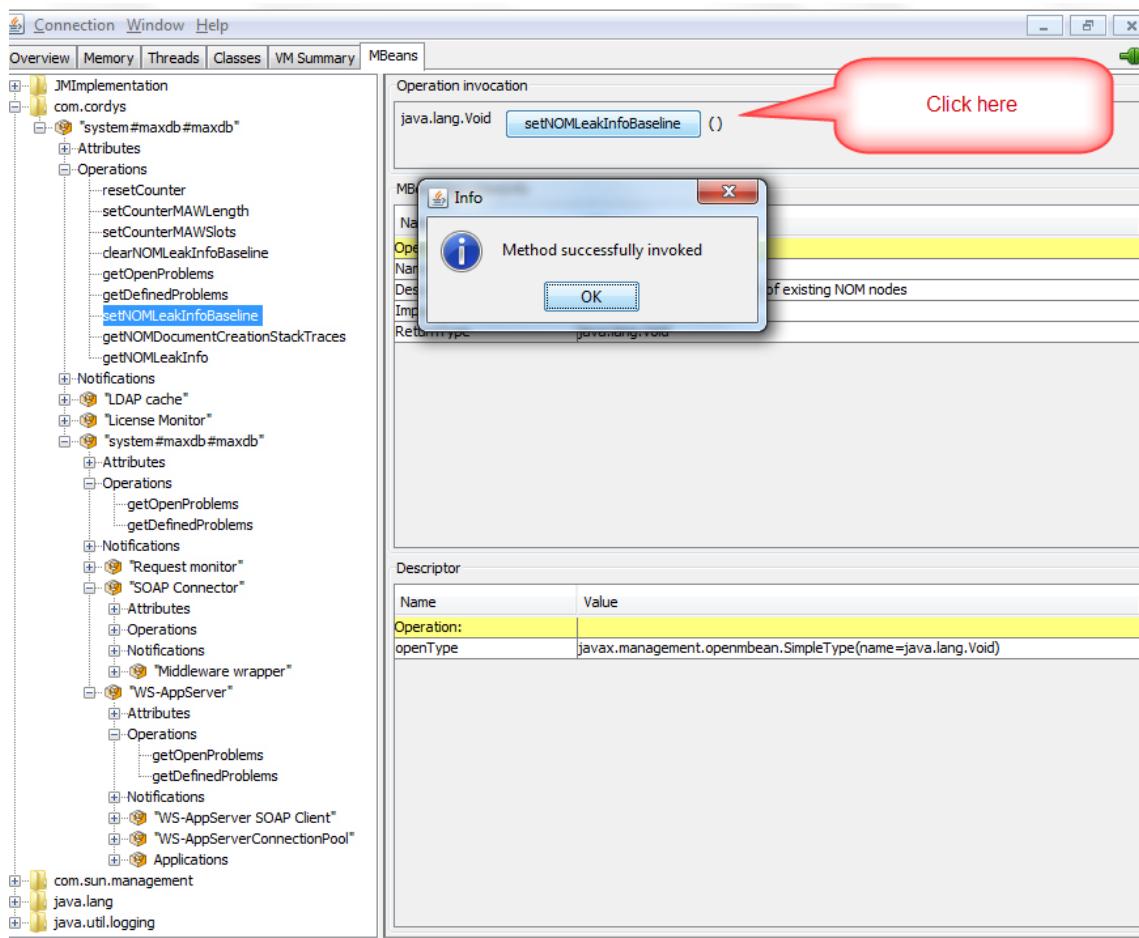
### Before you begin:

1. Create the WS-AppServer service container.
2. Generate Web services on the database.

### To find the memory leaks:

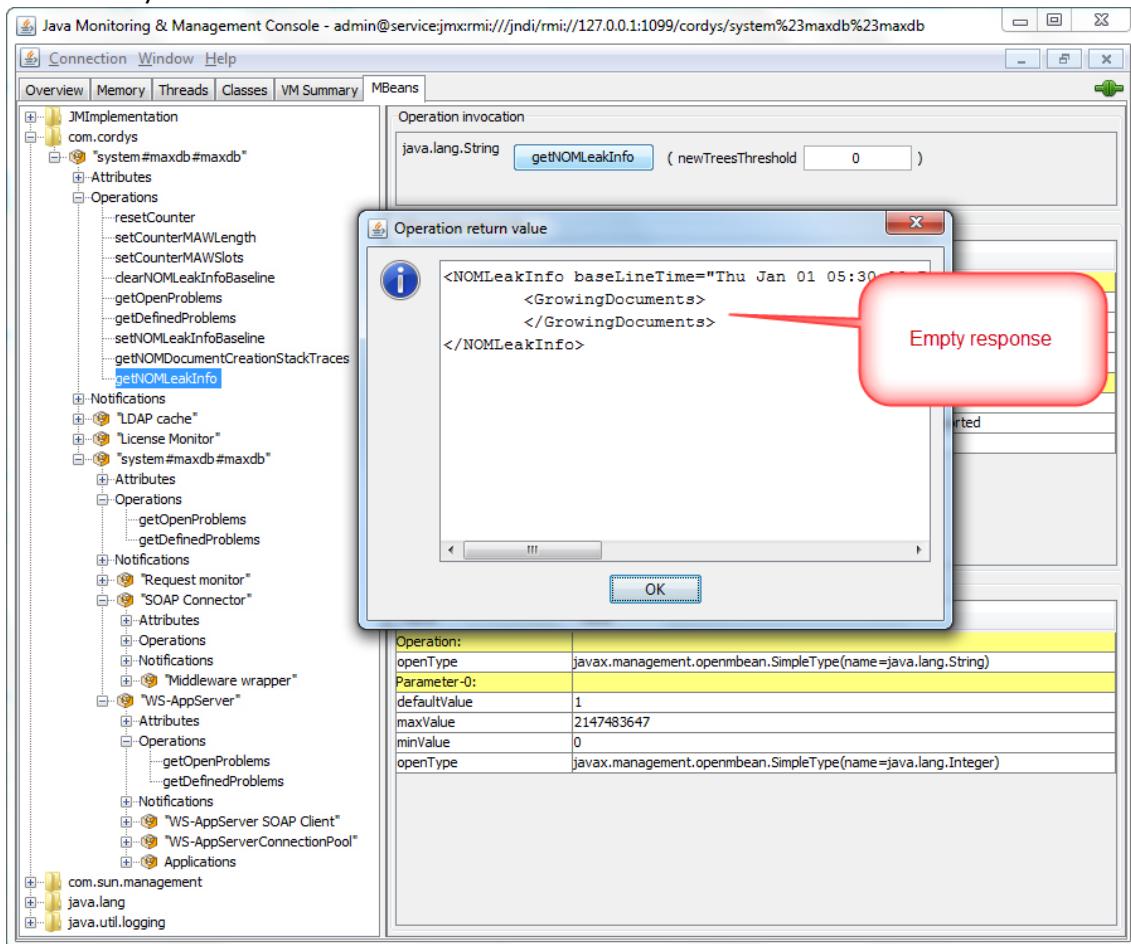
1. Connect to [JConsole](#) using the JMX address URL of the WS-AppServer service container.
2. Click **MBeans**, expand **com.cordys**, and then click **Operations** under the WS-AppServer service container.

3. Select and invoke the `setNOMLeakInfoBaseline` operation.



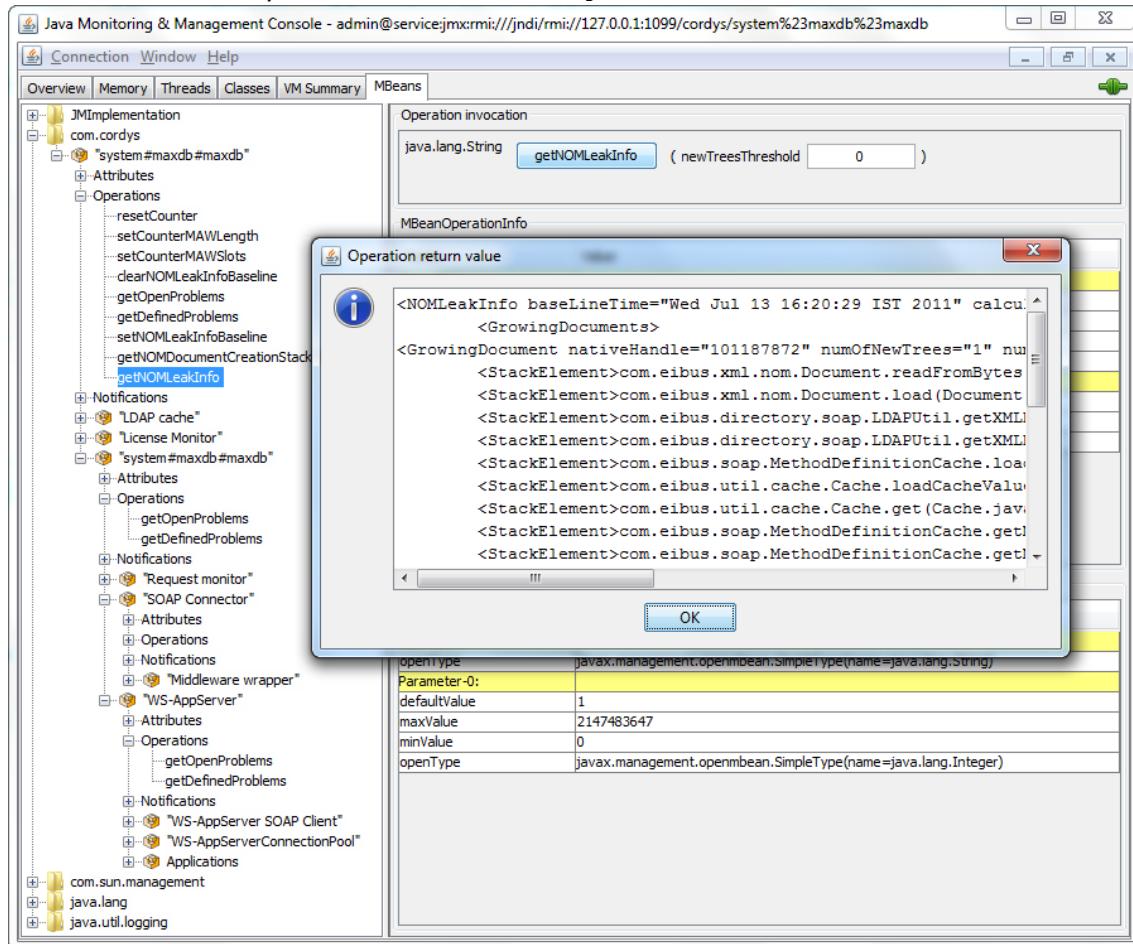
4. Select and invoke the `getNOMLeakInfo` operation without executing any Web Service. In this case, the response is empty under the `GrowingDocuments` node because there is

no memory leak.



- Run the Web services specific to the WS-AppServer service container using Service Test Tool and load or reload the database.

6. Invoke the `getNOMLeakInfo` operation. If there is a memory leak, you can see the stack trace of the memory leak under the `GrowingDocuments` node.



## Monitoring managed component counters

Consider that an administrator must monitor the counter value of a managed component of AppWorks Platform such as WS-AppServer ConnectionPool. The `ctr_numberOfCachedQueries` counter of WS-AppServer ConnectionPool records the number of queries (Web service operations) that are executed. The current value of the `ctr_numberOfCachedQueries` counter is stored in the `ctrv_numberOfCachedQueries_current` attribute.

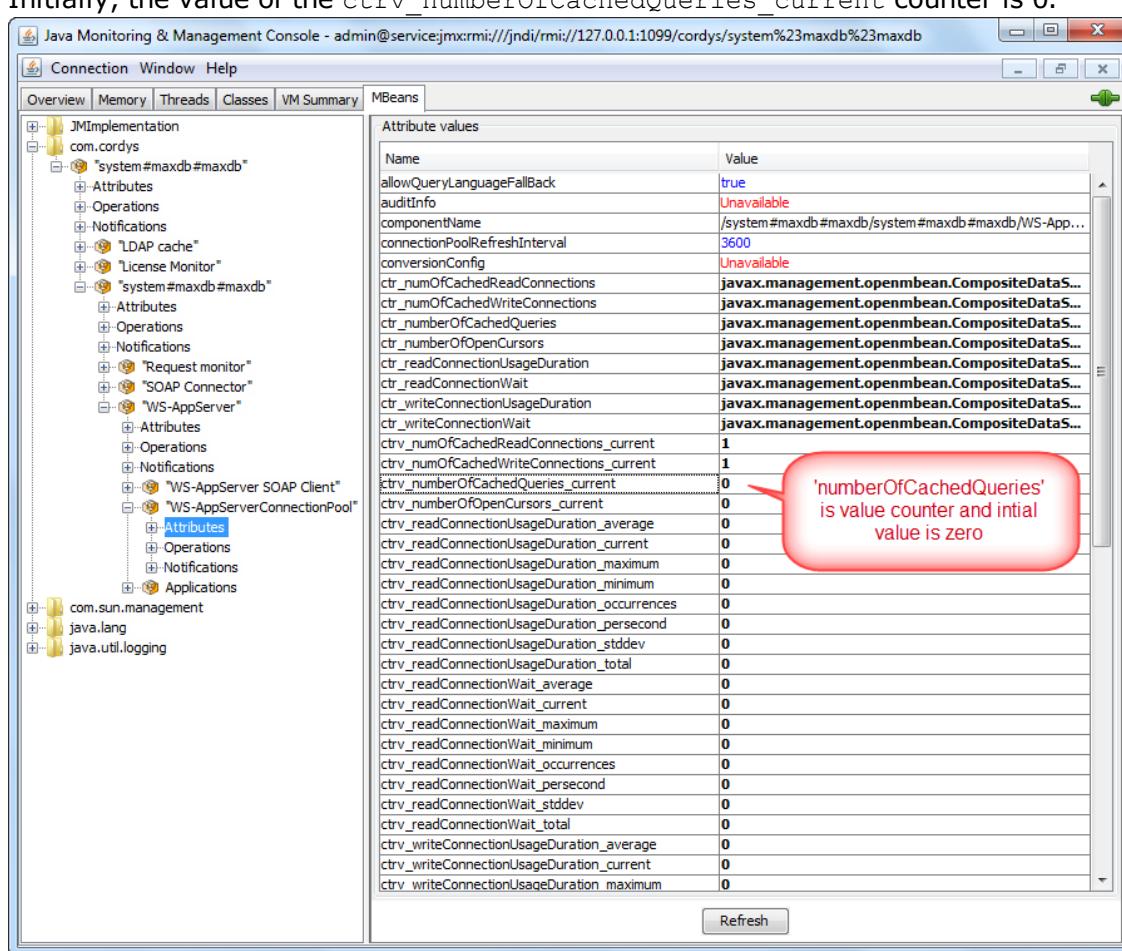
### Before you begin:

1. Create the WS-AppServer service container.
2. Generate Web services on the database.

### To monitor a counter:

1. Connect to [JConsole](#) using the JMX address URL of the WS-AppServer service container.  
For example:  
service:jmx:rmi:///jndi/rmi://<Host Name>/cordys/system%23maxdb%23maxdb
2. Click **MBeans**, expand **com.cordys** and **WS-AppServer** service container, and then click **Attributes** under WS-AppServer ConnectionPool.

Initially, the value of the `ctrv_numberOfCachedQueries_current` counter is 0.



3. Execute the Web services using Service Test Tool.

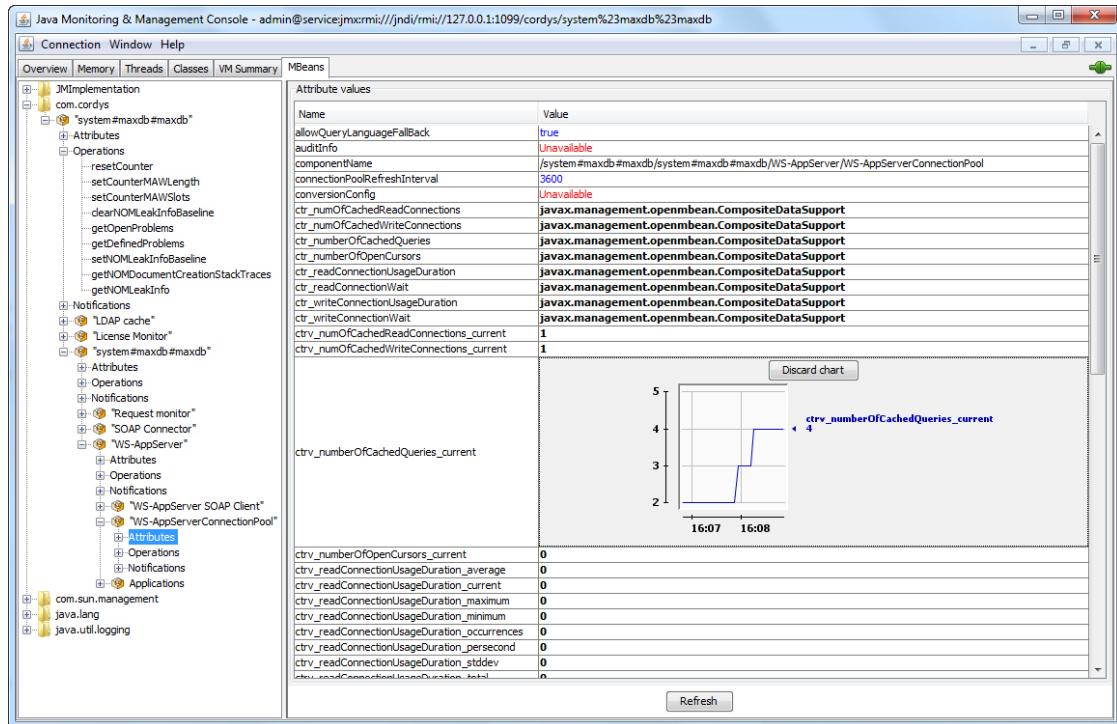
4. Click **Refresh** to refresh the Attributes node.

The value of the `ctrv_numberOfCachedQueries_current` counter increases based on

the number of executed Web services.

Name	Value
allowQueryLanguageFallback	true
auditInfo	Unavailable
componentName	/system#maxdb#maxdb/system#maxdb#maxdb/WS-App...
connectionPoolRefreshInterval	3600
conversionConfig	Unavailable
ctrv_numOfCachedReadConnections	javax.management.openmbean.CompositeDataS...
ctrv_numOfCachedWriteConnections	javax.management.openmbean.CompositeDataS...
ctrv_numberOfCachedQueries	javax.management.openmbean.CompositeDataS...
ctrv_numberOfOpenCursors	javax.management.openmbean.CompositeDataS...
ctrv_readConnectionUsageDuration	javax.management.openmbean.CompositeDataS...
ctrv_readConnectionWait	javax.management.openmbean.CompositeDataS...
ctrv_writeConnectionUsageDuration	javax.management.openmbean.CompositeDataS...
ctrv_writeConnectionWait	javax.management.openmbean.CompositeDataS...
ctrvv_numOfCachedReadConnections_current	1
ctrvv_numOfCachedWriteConnections_current	1
ctrvv_numberOfCachedQueries_current	2
ctrvv_numberOfOpenCursors_current	0
ctrvv_readConnectionUsageDuration_average	0
ctrvv_readConnectionUsageDuration_current	0
ctrvv_readConnectionUsageDuration_maximum	0
ctrvv_readConnectionUsageDuration_minimum	0
ctrvv_readConnectionUsageDuration_occurrences	0
ctrvv_readConnectionUsageDuration_persecond	0
ctrvv_readConnectionUsageDuration_stddev	0
ctrvv_readConnectionUsageDuration_total	0
ctrvv_readConnectionWait_average	0
ctrvv_readConnectionWait_current	0
ctrvv_readConnectionWait_maximum	0
ctrvv_readConnectionWait_minimum	0
ctrvv_readConnectionWait_occurrences	0
ctrvv_readConnectionWait_persecond	0
ctrvv_readConnectionWait_stddev	0
ctrvv_readConnectionWait_total	0
ctrvv_writeConnectionUsageDuration_average	0
ctrvv_writeConnectionUsageDuration_current	0
ctrvv_writeConnectionUsageDuration_maximum	0

5. To view the graph of the counter, double-click the value of the `ctrvv_numberOfCachedQueries_current` counter.



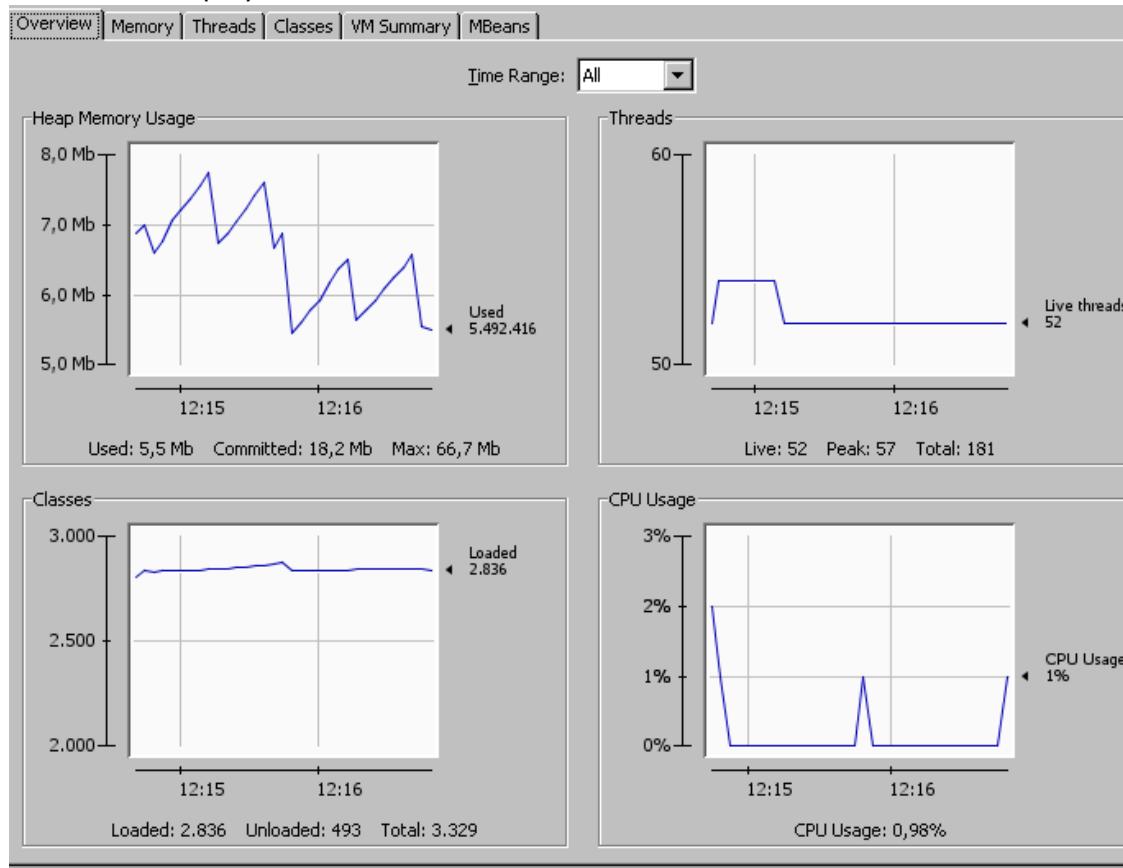
## Monitoring heap memory usage of service containers

Heap memory is an internal memory pool that is created when the service container or Java Virtual Machine (JVM) is started. In heap memory, the memory is allocated dynamically as needed.

### To monitor the heap memory usage of the WS-AppServer service container:

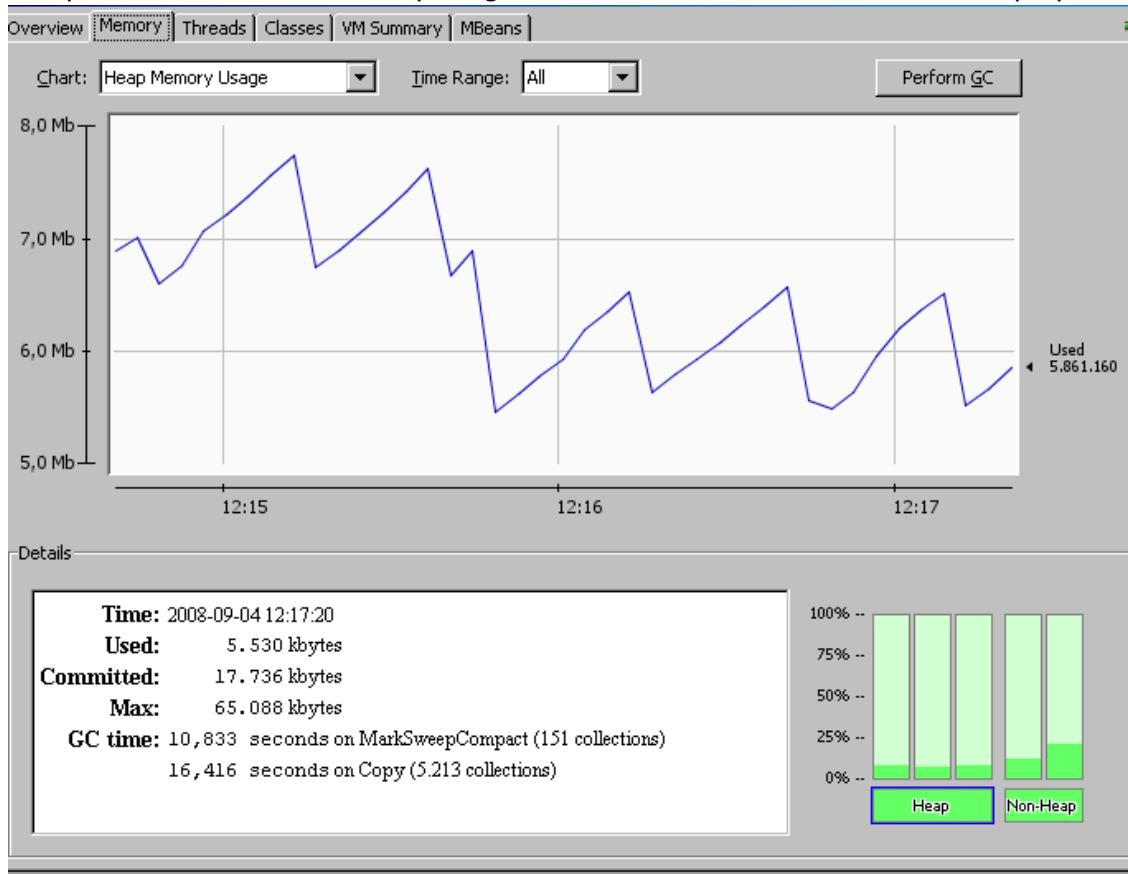
1. Connect to [JConsole](#) using the JMX address URL of the WS-AppServer service container.
2. Click the **Overview** tab in the interface.  
The heap memory usage of the service container, and the created threads and loaded

classes are displayed.



3. Click the **Memory** tab, and then click **Perform GC** to perform garbage collection forcefully.
4. Run the Web Services specific to the WS-AppServer service container multiple times and observe the memory graph.

An upward trend of the memory usage for each run of the Web service is displayed.



## Receiving managed component notifications

Consider that an administrator must view or receive notifications of a managed component such as Monitor when a license report is sent. The license report must be sent to the license manager when the license is updated and the `reportSentToLicense` alert is raised. An administrator can view this alert as a notification in JConsole.

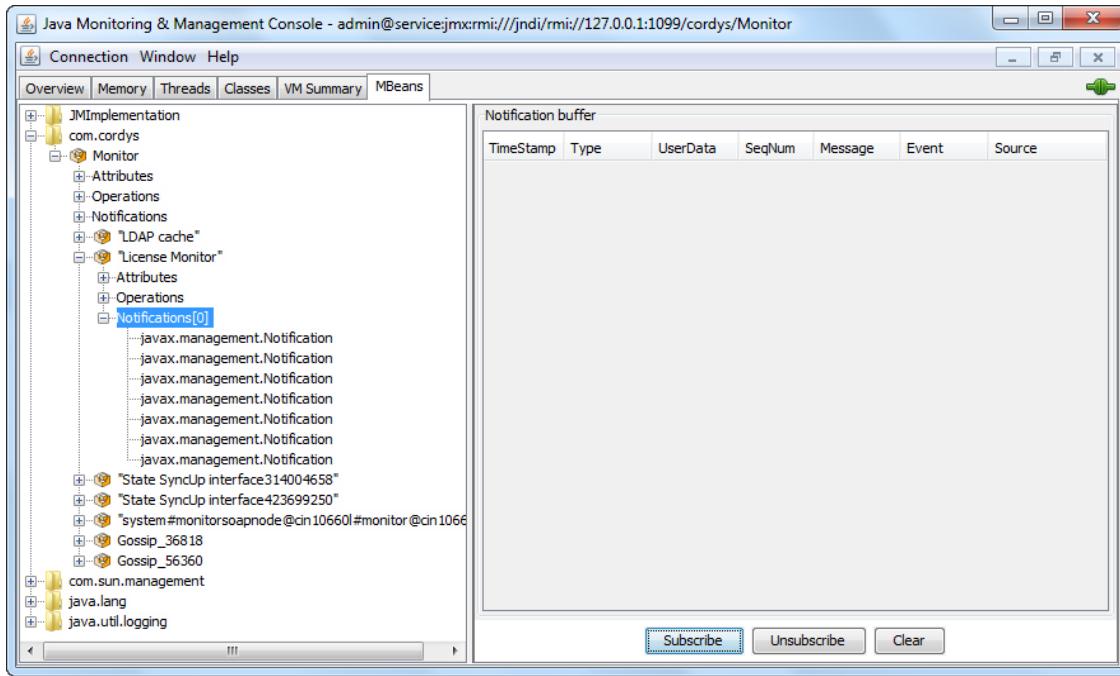
### To view a notification in JConsole:

1. Connect to [JConsole](#) using the JMX address URL of the Monitor. For example:  
`service:jmx:rmi:///jndi/rmi://<Host Name>/<instance>/Monitor`
2. Click **MBeans**, expand **com.cordys**, **Monitor**, and **License Monitor**, and then click **Notifications**.

The Notification buffer opens.

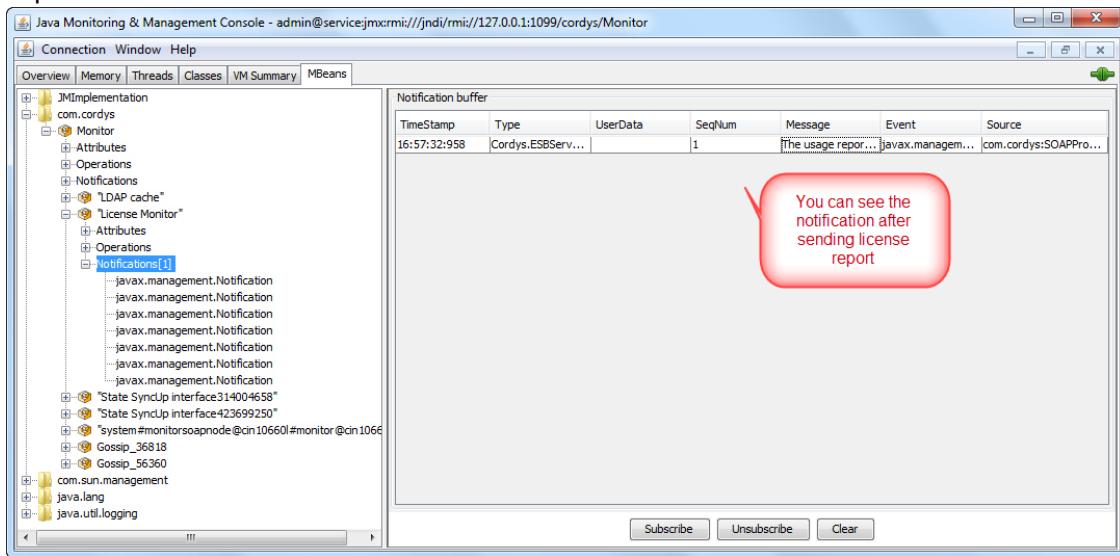
### 3. Click **Subscribe**.

The number of notifications is 0.



### 4. Update the license through (License Manager).

The license report is updated. The notification is available after sending the license report.



# Chapter 31

## Monitoring and maintenance

Use JMX for monitoring. You can also set multiple properties related to the authentication process and configuration of the single sign-on in the `wcp.properties` file.

This section describes:

- **JMX attributes in a single sign-on:** JMX attributes related to authentication and Single Sign-on.
- **JMX attributes for authentication:** JMX attributes related to authentication within service containers.

### JMX attributes for authentication

Use the JMX attributes to monitor the authentication process in every service container. The available attributes are:

JMX attribute name	Description
componentName	Name of the component.
ctrv_identityValidationDuration_average	Time needed to validate a WS-Security SAML token or an AppWorks Platform identity.
ctrv_identityValidationDuration_current	Current or last value of theidentityValidationDurationcounter
ctrv_identityValidationDuration_maximum	Maximum value of theidentityValidationDurationcounter
ctrv_identityValidationDuration_minimum	Minimum value of theidentityValidationDurationcounter
ctrv_identityValidationDuration_occurrences	Number of occurrences since reset of theidentityValidationDurationcounter
ctrv_identityValidationDuration_persecond	Events per second of anidentityValidationDurationoccurrence
ctrv_identityValidationDuration_stddev	Standard deviation of theidentityValidationDurationvalue

JMX attribute name	Description
ctrv_identityValidationDuration_total	Total value of the duration of all identityValidationDuration occurrences
ctrv_invalidIdentity_occurrences	Number of invalid identities encountered during identity validation.
ctrv_invalidIdentity_persecond	Number of invalid identities encountered per second during identity validation. Use this attribute to detect possible Denial of Service (DoS) attacks in an early stage.

## JMX attributes in a single sign-on

Use the JMX attributes to monitor the authentication process in the single sign-on service container. The attributes in the tree for single sign-on are located at `com.eibus > SOAProcessorOSProcess > Processor > SSOAppConnector > SSOCordysProtocolHandler > Attributes`.

The following table contains the available attributes:

JMX Attribute name	Description
componentName	Name of the component.
ctr_accessDenied	Frequency of invalid credentials occurrences. Use this attribute to detect possible Denial of Service (DoS) attacks in an early stage.
ctr_authenticateDuration	Amount of time to authenticate a user with the configured back-end plugin. Use this timer to detect possible bottlenecks in the authenticationWeb service operation.
ctrv_accessDenied_occurrences	Number of occurrences since the reset of accessDeniedcounter
ctrv_accessDenied_persecond	Events per second of accessDeniedcounter
ctr_authenticateDuration_average	Average duration of the authentication process.
ctr_authenticateDuration_current	Current or last value of theauthenticateDurationcounter
ctr_authenticateDuration_maximum	Maximum value of theauthenticateDurationcounter

JMX Attribute name	Description
ctr_authenticateDuration_minimum	Minimum value of theauthenticateDurationcounter
ctr_authenticateDuration_occurrences	Number of occurrences since the reset of authenticateDurationcounter
ctr_authenticateDuration_persecond	Events per second of anauthenticateDurationoccurrence
ctr_authenticateDuration_stddev	Standard deviation of theauthenticateDurationvalue
ctr_authenticateDuration_total	Total value of the duration of all authentication occurrences.

## JMS Connector Configuration interface

The JMS Connector Configuration interface contains the following tabs:

### General

Field	Description
JMS Polling Interval	Polling interval to poll messages from the destination. If not specified, the default value is 10 seconds. <b>Note:</b> This value is not used when the poller type is Compatibility. Only the Interval Poller uses this configuration. Enter a value in milliseconds.
Charset	Character set to use. When not specified, the default value is UTF-8.
Timeout	Timeout to use when sending SOAP messages to the bus.
Is BTC Enabled	Whether BTC (Binary Transformation Configuration) is enabled. The default value is false.
Run With Config Error	Whether the connector must start if there are configuration errors. The default value is false.
Check Connection On Request	Whether the connections must be checked for every request. The default value is false.

Field	Description
Disable Message Selectors	Whether the message selectors must be disabled even if the JMS type supports them. The default value is false. When set to true, the connector internally maintains the status of the message processing.

## Triggers

### Trigger

Field	Description
Name	Required. Logical name for the trigger to enter for the InboundMessageTrigger in the destination configurations.
Operation	Required. Name of the operation to call when the trigger is fired. <code>ExecuteProcess</code> is the operation name when triggering a business process.
Namespace	Required. Namespace of the operation to use when the trigger is fired. For example: <code>ExecuteProcess</code> in BPM, <code>http://schemas.cordys.com/bpm/execution/1.0</code>
User DN	Required. DN of the user using which the messages are sent. All the messages in the queue are sent using this account. For example: <code>cn=userId, cn=organizational users, o=system, cn=cordys, cn=CORDYS_JMS, o=opentext.net</code>
Organization DN	Required. DN of the organization in which the message is sent. For example: <code>o=system, cn=cordys, cn=CORDYS_JMS, o=opentext.net</code>
Charset	Charset to use for the trigger. The default value is UTF8.
Request Timeout	Timeout to use when sending the message to the bus. If the message is sent in a non-reliable mode, the system uses a socket to connect to the bus.
Parameters	Required. Parameter XML for the trigger to specify nodes and attributes that contain variables, which are replaced with information received from the message. The following variables are available that apply to attributes and nodes too if no limitation is provided: <ul style="list-style-type: none"> <li>■ <code>inputmessage</code>: Message body appears exactly as it is in the message. It is only available for nodes.</li> </ul> <p><b>Note:</b> This can produce errors if this is binary. Use <code>inputmessagebase64</code> instead.</p>

Field	Description
	<ul style="list-style-type: none"> <li>■ <b>inputmessagebase64:</b> This is similar to <code>inputmessage</code> with the difference that the message body is base64 encoded. It is only available for nodes.</li> <li>■ <b>xmlmessage:</b> Message body converted to XML using the Binary Transformation Configuration (if not configured, this will be empty). It is only available for nodes.</li> <li>■ <b>messageprotocol:</b> Binary transformation protocol to convert the message.</li> <li>■ <b>messageid:</b> Message ID from the message.</li> <li>■ <b>correlationid:</b> Correlation ID from the message.</li> <li>■ <b>fromdestination:</b> Name of the destination from where the message came.</li> <li>■ <b>reply2destination:</b> Name of the destination to reply to</li> <li>■ <b>jmstype:</b> JMS type from the message.</li> <li>■ <b>properties:</b> Properties found in the message. They are only available for nodes. Each property has a child node (named <code>property</code>) with a <code>name</code> attribute containing the name of the property and a <code>type</code> attribute containing the type of the property value. The value of the node is the property value, for example, <code>&lt;property name="Prop" type="String"&gt;value&lt;/property&gt;</code>.</li> </ul>

The following are the sample parameters for ExecuteProcess:

```

<type>definition</type>
    <receiver>com-cordys-
coe/jmsconnector/bpm/DemoTrigger</receiver>
    <source>Triggered via JMS</source>
    <message>
        <DemoTrigger_IP xmlns="http://schemas.cordys.com/default">
            <inputmessageinxml>
{$inputmessageinxml}</inputmessageinxml>
            <inputmessage>{$inputmessage}</inputmessage>
            <fromdestination>{$fromdestination}</fromdestination>
            <correlationid>{$correlationid}</correlationid>
            <inputmessagebase64>
{$inputmessagebase64}</inputmessagebase64>
            <xmlmessage>{$xmlmessage}</xmlmessage>
            <reply2destination>
{$reply2destination}</reply2destination>
            <jmstype>{$jmstype}</jmstype>
            <messageprotocol>{$messageprotocol}</messageprotocol>
            <messageid>{$messageid}</messageid>
            <properties>{$properties}</properties>
        </DemoTrigger_IP>
    </message>

```

## Custom Trigger Processor

Field	Description
Class Name	Fully qualified class name of the trigger. For example: com.test.custom.SampleCustomTrigger
Input Parameters	Input parameter passed to the custom trigger processor class.

## Destination Managers

### Destination Manager

Field	Description
Name	Required. Logical name of the destination manager. This is the name used in the SOAP API to identify the manager. Therefore, to send a message to this destination, you must use <code>destinationmanager.destination</code> .
JMS Vendor	Required. Class name of the <code>InitialContextFactory</code> to use. For ActiveMQ, you must use: <code>org.apache.activemq.jndi.ActiveMQInitialContextFactory</code>
Provider URL	Required. Value to use for the <code>java.naming.provider.url</code> parameter in the JMS context properties. Broker URL, for example, <code>tcp://localhost:61616</code> . The default value is set to <code>tcp://localhost:61616</code> . This must change as per the environment.
Username	Username for connecting to the specific destination manager. If security is set to broker, the username is mandatory.
Password	Password for connecting to the specific destination manager. If security is set to broker, the password is mandatory.
Connection Factory Name	Required. Logical name of the connection factory referred by the destination.
Connection User	Username to use if the JNDI connection needs a username. If security is set for the broker connection factory, the username is mandatory.
Connection Password	Password to use if the JNDI connection needs a username. If security is set for the broker connection factory, the password is mandatory.
Authentication Type	Authentication type to use. Set it to <b>none</b> when no authentication is required. Set it to <b>Basic</b> when a

Field	Description
	username/password is required to connect to the destination manager.
Charset	Character set to use for the destination manager. The default value is UTF8.
Timeout	Timeout to use when sending and receiving messages for the destination manager. The default destination manager timeout is 20 seconds.
Shutdown Destination	Specific queue to stop listening to the messages where the system creates an inactive listener.

## Destinations

Field	Description
Name	Required. Logical name for the destination. To identify a queue in the SOAP API, use the logical names. For example, if the destination manager is called <code>activemq</code> and the destination is called <code>CordysToOutside</code> , use <code>activemq.CordysToOutside</code> to identify the queue.
Access	Required. Access type of the queue through which you can ensure only messages are read from the destination. The possible values are: <ul style="list-style-type: none"><li>■ Read</li><li>■ Write</li><li>■ Both (Default)</li></ul>
Username	Queue-specific username to use.
Password	Queue-specific password to use.
Charset	Charset to use. When not specified, the charset of the destination manager is used. The default value is UTF8.
Inbound Message Trigger	Name of the trigger to execute when a message is picked up.
Authentication Type	Authentication type to use. When set to none, no authentication is required. Set it to basic if the destination requires a different username and password.
Destination Physical Name	Required. Physical details of the destination. Actual name of the queue or topic defined on the broker.
Type	Required. Destination type. The possible values are:

Field	Description
	<ul style="list-style-type: none"> <li>■ Queue</li> <li>■ Topic</li> </ul>
Message Selector	<p>Custom selector to use for filtering the messages that are monitored.</p> <p>The sample values are:</p> <ul style="list-style-type: none"> <li>■ JMSMessageID='&lt;JMS MessageId&gt;'</li> <li>■ JMSCorrelationID='&lt;JMS Message CorrelationId&gt;'</li> </ul>
Dynamic Destination Parameters	Parameters in case this is a dynamic destination.
Error Destination	Name of the destination to which messages must be copied when processing fails. This must be the actual error queue name, for example, <code>destinationManager.Queue</code> .
Number of Trigger Listeners	Number of threads to use for handling messages that are received from the queue. If <code>MaintainSequence</code> is set to <code>true</code> , this must always be set to 1. This is the number of listeners created on the destination.
Is Dynamic	Whether this destination is dynamic. The default value is false.
Is Durable Subscriber	Whether this destination is a durable subscriber. The default value is false.
Is Default Error Destination	Whether this destination is used as the default error queue if processing fails. The default value is false.

## Poller

Field	Description
Type	<p>Required. Type of poller to use.</p> <p>The possible values are:</p> <ul style="list-style-type: none"> <li>■ Compatibility (Default) <ul style="list-style-type: none"> <li>• Ignore Redelivery Count: The poller processes the message irrespective of the message redelivery count, if set to true. The default value must be set to true.</li> </ul> </li> <li>■ Interval_Polling <ul style="list-style-type: none"> <li>• Problem Check Interval: Interval to use to solve the problem by trying to process the message again in seconds. The default value is 300, which means 5 minutes. The expected value is in seconds.</li> <li>• Max Retry Count: Number of times the problem solver tries to resend the message. If after this time, the message is not sent, the</li> </ul> </li> </ul>

Field	Description
	<p>queue is blocked until an administrator unblocks the queue. The default value is 3.</p> <ul style="list-style-type: none"> <li>• Polling Interval: Polling interval to use for polling. If not specified, the generically configured polling interval is used. The default value is 10 seconds. The expected value is in minutes.</li> <li>• Maintain Sequence: Whether the sequence of the messages must be maintained. When set to true, if the current message fails to process, this poller is blocked until the problem is resolved.</li> <li>• Raise Problem When Blocked: Whether or not the problem must be raised in the problem registry when this poller is blocked.</li> <li>▪ Custom <ul style="list-style-type: none"> <li>• Class Name: Required. Name of the class.</li> <li>• Input Parameters: Parameters for the custom class.</li> </ul> </li> <li>▪ Request_Reply_Compatibility <ul style="list-style-type: none"> <li>• Ignore Redelivery Count: If set to true, the poller processes the message irrespective of the message redelivery count. The default value must be set to true.</li> </ul> </li> </ul>

## JMX counter statistics

This topic lists the statistics captured for the different counter types. The attributes are listed for each of the counter type (xxxx is the placeholder for the counter name).

### Value counters

Counter	Description
ctr_xxxx	Counter is a complex object.
ctrv_xxxx_current	Counter value.

### Event Occurrence counters

Counter	Description
ctr_xxxx	Counter is a complex object.
ctrv_xxxx_occurrences	Number of events since the counter reset.
ctrv_xxxx_persecond	Number of events per second in the moving average window.

## Event Value counters

Counter	Description
ctr_xxxx	Counter is a complex object.
ctrv_xxxx_average	Average event value in the moving average window.
ctrv_xxxx_current	Last event value.
ctrv_xxxx_maximum	Maximum event value in the moving average window.
ctrv_xxxx_minimum	Minimum event value in the moving average window.
ctrv_xxxx_occurrences	Number of events since the counter reset.
ctrv_xxxx_persecond	Number of events per second in the moving average window.
ctrv_xxxx_stddev	Standard deviation of the event values in the moving average window.
ctrv_xxxx_total	Total event value since the counter reset.



## Part IV

# Utilities

# Chapter 32

## Utilities

Utilities are the specially designed applications or tools that are used for specific tasks. They serve as additional tools that assist you while working with the AppWorks Platform.

However, they are not directly involved in the development or modeling environments of AppWorks Platform. Simply put, these are the stand-alone alternative programs for few capabilities of AppWorks Platform.

Depending upon the your role, these utilities enable you to access, manage, configure, and debug system resources. In addition to these, you view and debug SOAP messages, format XML messages, test the performance of the Web Gateway, view and manage LDAP content and XML Store content.

AppWorks Platform offers the following utilities:

- **Management console** - The Management Console enables you to perform various administrative tasks such as monitoring service containers, managing content, Synchronizing users, configuring licenses.
- **LDAP Explorer** - The LDAP Explorer enables you to access LDAP objects and modify their attributes.
- **XMLStore Explorer** - The XMLStore Explorer enables you to view, add, modify or delete the XML objects residing in the XML Store.
- **CoBOC Browser** - The CoBOC Browser allows you to create and maintain business objects. CoBOC Browser is deprecated in this version.
- **Signing Packages** - The Package Signer enables you to sign packages.
- **Web Services Explorer** - The Web Services Explorer provides quick access to Web services. Using this tool, you can access the WSDL for Web service operations.
- **Formatting XML Data** - The XML Formatter tool parses and formats XML strings.
- **Managing AppWorks Platform Using Web Task Manager** - The Web Task Manager enables you to manage the applications and libraries that run in the AppWorks Platform application.
- **Testing Web Service Operations Using Service Test Tool** - The Service Test Tool is used to test the Web service operation and check if the SOAP Request and Response are generated properly.
- **Testing Rules Using Rule Test Tool** - The Rule Test Tool is used to simulate the execution of rules on an object.

- Using Test Web Gateway - The Test Web Gateway enables you to test the performance of Web Gateway in executing multiple requests from the client.
- Using the SOAP Debugger - The SOAP Debugger enables you to test and debug SOAP requests that are exchanged with the backend by applications running in AppWorks Platform.
- Sending and Receiving SOAP Messages using Simple Client - The Simple Client enables you to send and receive SOAP Messages. The tool offers a feature to compose and edit the outgoing SOAP message that is to be sent to a specific back end.
- Healthcheck - The Healthcheck utility enables you to check the health of the Cordys system. The Cordys system includes both the AppWorks Platform (<instance name>) and the AppWorks Platform gateway.

# Chapter 33

## Management console

The management console is used to perform various administrative tasks. To use the management console, log into the LDAP server. For more information, see [Accessing Management Console](#).

**Note:** You cannot use this console to access the properties and network configuration.

Use the Explorer to:

- Manage service containers
- Modify license configuration
- Manage LDAP content

The Explorer depends on the LDAP service container, XML store service container, and Web server. If these do not work properly, you cannot use the Explorer. In case of such an eventuality, the management console facilitates managing AppWorks Platform as it does not depend on these components. The following administrative functions can be performed using the management console:

Utility	Description
Repository Browser <b>Note:</b> See <a href="#">Using repository browser</a> for details.	Simple interface that enables an administrator to view, update, or delete data in the repository.
State Syncup <b>Note:</b> See <a href="#">Using state syncup to synchronize instances of AppWorks Platform</a> for details.	Supports distributed applications. When multiple instances of the same application run on a single or different machines, synchronization among these instances is essential. AppWorks Platform State SyncUp provides synchronized data across instances, similar to it being available locally.
Content Management <b>Note:</b> See <a href="#">Managing LDAP content</a> for details.	Enables viewing, managing, diagnosing, and repairing of LDAP content, which could represent organizational roles, organizational users, and applications.

<b>Utility</b>	<b>Description</b>
Role & ACL Overview <b>Note:</b> See <a href="#">Viewing roles and ACLs</a> for details.	Roles and subroles within the system and the ACLs configured to them.
Monitor Service Containers <b>Note:</b> See <a href="#">Managing service containers using the management console</a> for details.	Manage service containers. You can perform the following activities using this utility: <ul style="list-style-type: none"> <li>■ View all service containers</li> <li>■ Start service containers</li> <li>■ Stop service containers</li> <li>■ Restart service containers</li> <li>■ View error details of service containers</li> </ul>
License Configuration See <a href="#">Modifying AppWorks Platform license information</a> .	Configure license management.
Security Properties <b>Note:</b> See <a href="#">Configuring AppWorks Platform security policy</a> for details.	Modify security properties that are stored in the <code>securityproperties.xml</code> file located in the <code>&lt;AppWorks_Platform_installdir&gt;/config</code> folder.
Network Configuration <b>Note:</b> See <a href="#">Enabling AppWorks Platform to run in standalone or network mode</a> for details.	Modify AppWorks Platform configuration and enable it to work with or without network connectivity. This is useful in cases where the network connectivity might be temporarily unavailable and AppWorks Platform must function as a standalone computer.
Administration Recovery	Restore the administrator account with its permissions and reset its password.
Platform Properties <b>Note:</b> See <a href="#">Global configuration properties for AppWorks Platform</a> for details.	View and modify the <code>wcp.properties</code> file. This file is located in the <code>&lt;AppWorks_Platform_installdir&gt;</code> , and is read for all attributes and properties that need to be set for AppWorks Platform to work.

**Note:** You cannot access the LDAP on another machine for the following utilities. The content of these utilities can be accessed in your machine if the AppWorks Platform server is installed. You can neither view nor manage the content in these utilities in another machine.

- License configuration
- Properties

- Security properties
- Network configuration
- State syncup

The monitor service containers utility connects to another machine. You can view the service containers in another machine but cannot start or stop them. The content management utility connects to another machine and also enables accessing and managing the content.

**Note:** Management console supports Chinese characters in the Windows edition. This feature requires the Arial Unicode MS font for displaying multibyte characters. This font is available in Microsoft Office 2000, FrontPage 2000, Office XP, Windows 2000, and Windows XP.

## Managing LDAP content

The Content Management utility of the Management Console enables viewing, managing, diagnosing, and repairing LDAP content that represents organizational roles, organizational users, and applications in AppWorks Platform.

This section includes the following topics:

- [Managing LDAP entries](#)
- [Adding an entry to LDAP](#)
- [Diagnosing LDAP entries](#)

## Adding an entry to LDAP

### Before you begin

[Accessing Management Console](#) and connect to LDAP with your credentials.

### To add an entry to LDAP:

1. In the Management Console window, click **Content Management**.  
The Content Management page appears.
2. To add the new entry, select a LDAP entry on the LDAP tree and do one of the following:
  - Right-click the selected LDAP entry, and select **New**.  
or
  - Click **Edit** Menu and select **New Entry**.  
or
  - Click  (New) on the toolbar.  
The Add page appears.

3. Enter a **Name** for the LDAP entry and select an **Object Class**.

**Note:** For more information, see Object Classes in the *AppWorks Platform API Guide*. When a Name for the LDAP entry is typed, the Parent in Directory Service value is automatically generated.

4. In the Add page, click **Next**.

The Attribute Viewer page appears with the attributes and corresponding values of the new LDAP entry.

5. Click **OK**.

The new LDAP entry is added and appears in the LDAP tree.

## Diagnosing LDAP entries

This topic describes the procedure to diagnose LDAP entries using the management console.

### Before you begin:

See [Accessing Management Console](#) and connect to LDAP using your credentials.

The content management utility of the Management Console has a diagnostic tool that diagnoses and repairs LDAP entries.

### To diagnose an LDAP entry:

1. In Management Console, click **Content Management**.

The Content Management page opens.

2. Click **Diagnostics** and select <**Diagnostic option**>. For more information refer to the [Diagnostic options for LDAP entries](#).

**Note:** Select the **Check all** option to perform all diagnostic operations sequentially.

3. In the Information dialog box, click **Yes** to continue the diagnostic process.

The <Diagnostic option> window opens with the status and results of the diagnostic process.

**Note:** In the results, a warning may appear for the Repository Service, with Referential Integrity as the source. Dismiss this warning as it does not impact the operation.

## Managing LDAP entries

Use the Content Management page of the Management Console to:

- View an LDAP entry
- Modify an LDAP entry
- Delete an LDAP entry
- Rename an LDAP entry

### To view an LDAP entry:

Click the required entry in the LDAP tree.

The attributes of the selected LDAP entry with the corresponding values are displayed in the right pane.

### To modify the attributes of an LDAP entry:

1. Click the required **LDAP** entry.

The attributes of the selected LDAP entry with the corresponding values are displayed in the right pane.

2. In the **Attribute** list, double click the attribute that you want to modify. Or, right click the attribute and select **Edit**.

**Note:** You cannot modify the **cn** and **objectClass (non-leaf)** attributes.

3. Make the necessary changes in the **Attribute Properties** page, and click **Save**.

### To delete an LDAP entry:

1. Right click the required **LDAP** entry in the LDAP tree, and select **Delete**.

or

Select the **LDAP** entry and click on the toolbar.

Or

Select the required **LDAP** entry, click **Edit** and select **Delete Entry**.

2. In the **Delete confirmation** dialog box, click **Yes**:

**Note:** You cannot rename the **cn** and **objectClass (non-leaf)** attributes.

### To rename an LDAP entry:

1. Select the required **LDAP** entry, click **Edit**, and select **Rename**. Or, right click the **LDAP** entry and select **Rename**.

2. Enter information in **New Name** for the LDAP entry in the **Rename** page and click **OK**.

**Note:** You cannot rename the **cn** and **objectClass (non-leaf)** attributes.

## Accessing Management Console

### To access the Management Console:

1. Depending on your operating system, do one of the following:

Operating system	Procedure
Windows	In a Windows-based computer, click <b>Start &gt; Programs &gt; OpenText AppWorks Platform &lt;Version&gt; &gt; &lt;Instance Name&gt; &gt; Tools &gt; Management Console.</b>
Linux	<ol style="list-style-type: none"> <li>Launch Terminal.</li> <li>CD &lt;AppWorks Platform_installdir&gt;/bin</li> <li>Run ./cmc.sh</li> </ol>

The Management Console window opens.

- On the Management Console, click  (**Connect**).  
The LDAP Logon window opens.
- Enter the required information to connect to the LDAP server.

The following table contains information about the LDAP Logon interface:

Field	Description
Server	Name of the LDAP server.
Port	Port to connect with.
Use as default on this computer	Select to set the user name as default for the next logon.
LDAP Server is running in SSL Mode	Select to connect to LDAP running in SSL mode.
Search Root	LDAP root.
Name	User name for logging on to LDAP.
Password	Password to connect to LDAP.

- Click **OK**.  
You are connected to the LDAP server as LDAP administrator.

**Note:** The  (Connect) option is disabled and  (Disconnect) is enabled. To disconnect LDAP, click  (Disconnect) on the Management Console window.

## Administration recovery

AppWorks Platform has a default administrator account called 'administrator' with a password that is set during installation. It might happen that the account cannot be used anymore to administer the AppWorks Platform deployment, for example, due to

incorrect configuration of authentication or deletion of the account. The Administration Recovery function helps to recover the administrator account.

The Administration Recovery utility does the following:

- Re-creates the 'administrator' user.
- Sets the password of the 'administrator' user as provided.
- Grants the system administrator privileges to the 'administrator' user.

**To recover the administrator account:**

1. In a Windows based computer, click **Start > Programs > AppWorks Platform <Version> > <Instance Name> > Tools > Management Console.**

**For Linux users**

- a. Launch Terminal
- b. CD AppWorks Platform Installation Directory/bin
- c. Execute ./cmc.sh

The Management Console window opens.

2. Click **Administration Recovery** in the Management Console window.  
The Administration Recovery page opens.
3. In **Password**, enter a new password.
4. Click **Set password**.

To force a direct sign in, click the displayed URL.

**Note:** The password override is effective only for signing into AppWorks Platform directly. It does not work when authenticating with, for example, OTDS or SAML 2.

## Enabling AppWorks Platform to run in standalone or network mode

The network configuration utility of the management console enables AppWorks Platform to work either with network connectivity or in standalone mode. This is useful when the network connectivity may be temporarily unavailable, and AppWorks Platform needs to function as a standalone system. Running AppWorks Platform in a standalone mode requires all AppWorks Platform-related components, including OpenText CARS, to run on the local host.

The management console provides a list of NICs and the IP addresses using the network interface. The network card chosen during installation is displayed in the Network Configuration page. You can edit this page.

**To enable AppWorks Platform to run either in standalone or network mode:**

1. Depending on the operating system, do either of the following:

Operating system	Procedure
Windows	In a Windows based computer, click <b>Start &gt; Programs &gt; OpenText AppWorks Platform &lt;Version&gt; &gt; &lt;Instance Name&gt; &gt; Tools &gt; Management Console.</b>
Linux	<ol style="list-style-type: none"> <li>1. Launch Terminal.</li> <li>2. CD &lt;AppWorks Platform_installdir&gt;/bin</li> <li>3. Run <code>./cmc.sh</code></li> </ol>

The Management Console window opens.

2. In the management console window, click **Network Configuration**. The Network Configuration page opens.
3. Do one of the following:

Option	Description
Enable Standalone configuration	Enables AppWorks Platform to work in the standalone mode. You can access AppWorks Platform using <code>http://localhost/home</code>
Disable Standalone configuration	Enables AppWorks Platform to work in the network mode. You can access AppWorks Platform using: <ul style="list-style-type: none"> <li>■ <code>http://machinename/home</code></li> <li>■ <code>http://localhost/home</code></li> <li>■ <code>http://IP address/home</code></li> </ul>

**Note:** By default, there is a network card or NIC plugged in for every machine enabling a network connection. If your machine is configured with multiple network cards, they are listed in the Interface Display Name list. However, if you intend to connect through other network cards, select Disable Standalone Configuration option, and select a Network Interface from the list.

4. Save the changes and restart the OpenText AppWorks Platform using the <instance name>, OpenText CARS, and Webserver for the modifications to be effective.

**Note:** If the standalone mode is enabled for AppWorks Platform, any change in network, for example, from one network to another (for example, change from wireless to wired network) requires you to restart of the OpenText AppWorks Platform (<instance name>), OpenText CARS, and Webserver to make it work.

## Managing AppWorks Platform properties

The AppWorks Platform properties are used to start the system resources that are crucial to perform certain administrative tasks. The corresponding file for these properties is `wcp.properties` and is located at the following path: `<AppWorks Platform_installdir>/config` folder.

You can manage [Platform properties](#) by adding a property, modifying the name and value of a property, and deleting an unnecessary property.

### Before you begin:

- Access Management Console and connect to LDAP as an authenticated user. See [Accessing Management Console](#).
- You must have the role of a `SystemAdmin` to manage Platform Properties.

### To add a property:

1. Access Management Console and click **Platform Properties**. See [Accessing Management Console](#).
  - a. Click  (Add) on the toolbar.  
The Add Property window is displayed.
  - b. Type the name and value for the property, and then click **OK**.  
The new property displays in the list of properties in the Platform Properties page.

### To modify a property:

- Double-click the required property, make the necessary changes to the name and value of the property, and then click **OK**.  
The changes display in the Platform Properties page.

### To delete a property:

- Select the property which you want to delete, and click  (Delete) on the toolbar.  
The selected property is deleted from the list in the Platform Properties page.

**Note:** When you insert, change, and delete a property, it is saved automatically. Also, before any action, a verification is done to see if the property file has changed. If there is a change, the user is prompted to reload the values.

## Managing service containers using the management console

**Required role:** SystemAdmin

### To manage service containers:

1. Navigate to the Management Console and connect to LDAP using your credentials. For more information, see Accessing Management Console.
2. Depending on the operating system of your computer, do one of the following:

Windows based	Linux based
Click <b>Start &gt; Programs &gt; OpenText AppWorks Platform &lt;Version&gt; &gt; &lt;Instance Name&gt; &gt; Tools &gt; Management Console.</b>	<ol style="list-style-type: none"> <li>a. Launch <b>Terminal</b>.</li> <li>b. CD &lt;AppWorks Platform_installdir&gt;/bin</li> <li>c. Run ./cmc.sh</li> </ol>

The Management Console window opens.

3. In the Management Console window, click **Monitor Service Containers**.

The Monitor Service Container window displays a list of service containers.

The following options are available:

Option	Action	Result
To Start a Service Container	Right-click the service container that must be started, and select Start.	The service container is started and the status icon changes to  .
To Stop a Service Container	Right-click the service container that must be stopped, and select Stop.	The service container is stopped and the status icon changes to  .
To Restart a Service Container	Right-click the service container that must be restarted, and select Restart.	The service container is restarted and the status icon changes to  .
To View Error Details of a Service Container	Right-click the service container with a configuration error, and select Show Error Details. The service container with a configuration error is displayed with the  status icon.	The Error Details dialog box displays the error message.

## Modifying AppWorks Platform license information

When AppWorks Platform License key expires, OpenText AppWorks Platform (<instance name>) will not be running and thus none of the AppWorks Platform services will work. You can use Management Console to renew or modify the license, which will enable OpenText

AppWorks Platform (<instance name>) to start temporarily. The OpenText AppWorks Platform (<instance name>) will be enabled for a limited period during which you can renew your license key by sending the usage report and providing the right key.

### **Before you begin**

Access Management Console and connect to LDAP as an authenticated user. See [Accessing Management Console](#).

### **To modify license information:**

1. In a Windows based computer, click **Start > Programs > OpenText AppWorks Platform <Version> > <Instance Name> > Tools > Management Console.**

#### **For Linux users**

- a. Launch Terminal
- b. CD <AppWorks Platform\_installdir>/bin
- c. Execute ./cmc.sh  
The Management Console window opens.

2. Click **License Configuration** in the Management Console window.

The License Configuration page is displayed.

3. In **License Mode**, select the installation type.

**Master Machine** is enabled, if **Multiple** is selected; otherwise it is disabled.

**Note:** AppWorks Platform licensing recognizes two types of installations.

Single - In this installation, there is only one installation of AppWorks Platform in a site.

Multiple - In this installation, the site consists of several installs. In other words, in this installation, the customer site can have more than one installation of AppWorks Platform. In this type of installation, one Install is recognized as the Master, while the others are the Contributors. This facilitates the Usage Report generation and management.

4. Modify the required information in the License Configuration window. For more information, see [License configuration interface](#).

5. Select the **Send Usage Report Automatically** checkbox.

**Note:** If this checkbox is selected, the usage report is sent to the AppWorks Platform Licensing Service every month. The license key thus obtained from the AppWorks Platform Licensing Service is automatically updated in the LDAP. If this checkbox is cleared, the Usage Report needs to be manually generated and sent to the AppWorks Platform Licensing Service for obtaining the license key.

6. Click .

AppWorks Platform licensing information is modified.

### **Post requisite:**

After the license is updated, restart the OpenText CARS, Cordys, and Web server services, as follows:

1. Stop Web Server.  
service httpd stop
2. Stop OpenText AppWorks Platform (<instance name>).  
For example, if BOP is the AppWorks Platform instance name, then the service name will be wcpdBOP.  
service wcpdBOP stop
3. Stop OpenText CARS.  
For example, if BOP is the OpenText CARS instance name, then the service name will be OpenText CARS-slapdBOP.  
service OpenText CARS-slapdBOP stop
4. Start OpenText CARS.  
service OpenText CARS-slapdBOP start
5. Start OpenText AppWorks Platform (<instance name>).  
service wcpdBOP start
6. Start Web Server.  
service httpd start

## Platform properties

<b>AppWorks Platform property</b>	<b>Description</b>	<b>Default value (milliseconds)</b>
bus.blacklist.usersession.time	This property determines the duration of a session for all the users. The messages received within this duration are analyzed against the factors. The value must be specified in milliseconds.	3600000
<user DN>.session.time	This property determines the duration of a session for a particular user. The messages received within this duration are analyzed against the factors. The value must be specified in milliseconds.	3600000
bus.blacklist.daemon	This property determines the interval of the calculations. The messages received within an interval are stored and taken up for calculation along with the requests received during all the	15000

<b>AppWorks Platform property</b>	<b>Description</b>	<b>Default value (millisecond s)</b>
	preceding intervals within the session.	
bus.blacklist.thresholdvalue	The threshold value is determined from this property.	3
security.dos.default.requestLength	The value of this property determines the maximum size of a request that can be received from any user. Requests sent by any user are blocked if they exceed this size. This value must be specified in KB.	4096
security.dos.<user ID>.requestLength	The value of this property determines the maximum size of a request that can be received from a particular authenticated user. This value must be specified in KB. This value is given priority over the value of the security.dos.default.requestLength property.	4096
security.dos.watchInterval	The value of this property determines the duration within which a certain number of requests can be received from any user. The duration must be specified in milliseconds.	1000
security.dos.default.numRequest	The value of this property determines the maximum number of requests that can be received from any user, within the duration specified in the security.dos.watchInterval property.	100
security.dos.blockPeriod	The value of this property determines the maximum number of requests that can be received from a particular	100

<b>AppWorks Platform property</b>	<b>Description</b>	<b>Default value (milliseconds)</b>
	authenticated user, within the duration specified in the security.dos.watchInterval property. This value is given priority over the value of the security.dos.default.numRequest property.	
security.dos.<user ID>.numRequest	The value of this property determines the duration for which a user must be blocked if the requests from the user exceeds the given number within the given duration. The duration must be specified in milliseconds.	360000
bus.blacklist.factor.NUM_REQUESTS	Number of requests sent by the user to the gateway.	0
bus.blacklist.factor.NUM_INVALID_MESSAGES	Number of invalid messages.	2
bus.blacklist.factor.UNKNOWN_USER	Messages from an unknown user.	2
bus.blacklist.factor.NUM_MESSAGE_SENT	Number of messages sent to the service container by the gateway.	0
bus.blacklist.factor.NUM_MESSAGE RECEVIED	Number of responses received by the gateway from the service container.	0
bus.blacklist.factor.NUM_TIME_OUT_ERRORS	Number of timeout errors occurred when the messages are received and sent to a user.	1
bus.blacklist.factor.NUM_SOAP_FAULTS	Number of SOAP faults occurred when the messages are received and sent to a user.	0

## Using repository browser

The repository browser is a simple interface that enables an administrator to view data in the repository. It displays data in a hierarchy for easy viewing. The administrator can also use this as a troubleshooting tool when data in the repository is corrupted.

1. To use the repository browser:

Operating system	Procedure
Windows	Click <b>Start &gt; Programs &gt; OpenText AppWorks Platform &lt;Version&gt; &gt; &lt;Instance Name&gt; &gt; Tools &gt; Management Console.</b>
Linux	a. Launch <b>Terminal</b> . b. CD <AppWorks Platform_installdir>/bin c. Run ./cmc.sh

The Management Console window opens.

2. Click the **Repository Browser** utility. You may be prompted to provide **Logon** information. See [Accessing Management Console](#) for details.  
The Select Data Source pane opens, displaying the Select Data Source list.
3. Select the required data source from Data Source list and click **Connect**.  
The data in the repository is displayed in a hierarchical structure in the repository browser.
4. Navigate to the required folder in the tree structure and click the document.

**Note:**

- Alternatively, you can use the search feature to find the required artifact. Click search on the toolbar, select the branch that contains the artifact from the branch list, provide the document identifier in the Document ID field and click **Search**.
- When you click an artifact in the repository tree, the information related to the artifact is displayed in the pane. The related data is displayed in an XML format.

## Using state syncup to synchronize instances of AppWorks Platform

State SyncUp is a utility that facilitates synchronization of distributed instances of AppWorks Platform applications running on multiple systems. It also facilitates synchronization of information or messages related to the state of the system.

1. To use state syncup:

Operating system	Procedure
Windows	Click <b>Start &gt; Programs &gt; OpenText AppWorks Platform &lt;Version&gt; &gt; &lt;Instance Name&gt; &gt; Tools &gt; Management Console.</b>
Linux	<ol style="list-style-type: none"> <li>a. Launch <b>Terminal</b>.</li> <li>b. CD &lt;AppWorks Platform_installdir&gt;/bin</li> <li>c. Run <code>./cmc.sh</code></li> </ol>

The Management Console window opens.

2. In the Management Console window, click **State SyncUp**.  
The LDAP Logon window opens.
3. To connect to the LDAP server, enter the required information. See [LDAP logon interface](#) for details.
4. Click **OK**.  
The Syncup Administration page opens displaying the state syncup information. For more information on the contents of the State Syncup screen, see Example of State Syncup Contents in the *AppWorks Platform API Reference Guide*.

## Viewing roles and ACLs

### Before you begin:

Access Management Console and connect to LDAP as an authenticated user. See [Accessing Management Console](#).

The Role and ACL Overview utility enables you to view the following:

- Default roles
- Subroles extended from the default roles. For example, in Cordys SystemAdmin, OrganizationalAdmin and Developer are sub-roles of the everyone role.
- ACLs set for the default roles and subroles.

**Note:** For more information on roles and subroles in AppWorks Platform, see [Managing roles](#). For more information, see [ACL types](#).

1. In the Management Console window, click **Role and ACL Overview** utility.  
The Role Overview page appears.

Expand...	To View...
All roles with subroles	All default roles and subroles with the corresponding ACLs.
All roles with one sublevel	Only the default roles with the corresponding ACLs.

## Global configuration properties for AppWorks Platform

AppWorks Platform can be configured through a set of global configuration properties. These properties are stored in the `wcp.properties` file that is available in the `<AppWorks Platform_installdir>/config` folder.

You can override these properties as follows:

- For service containers, add the JRE properties in the JRE configuration tab of these specific service containers. See Service container configuration Interface in the *AppWorks Platform Advanced Development Guide*.
- For TomEE (and the Service Containers running in TomEE), add the properties in the `CATALINA_BASE/conf/catalina.properties` file. See <https://tomcat.apache.org/tomcat-7.0-doc/config/>.

You can edit the `wcp.properties` file by using the Platform Properties task in the Management Console. This file contains properties of the following components:

- [LDAP related properties](#)
- [Web gateway](#)
- [XML parser \(NOM\) properties](#)
- [Management Console properties](#)
- [UDDI connector properties](#)
- [General properties](#)
- [SSL options on OpenText CARS](#)
- [Port ranges within AppWorks Platform](#)
- [Debugging](#)

### LDAP related properties

This table describes LDAP related properties.

**Note:** By default, multicast is not used anymore. When multicast is enabled, it is only used in the bootstrap phase.

<b>Property</b>	<b>Default value</b>	<b>Description</b>
bus.ldap.cache.maxSize	10000	Represents the maximum cache size.
bus.ldap.information.file	true	Setting the value to 'false' enables multicast for the bootstrap phase.
bus.ldap.processor.host		Host on which the LDAP server is found.
bus.ldap.processor.maximumno ofsearchresult	2000	
bus.ldap.processor.password		A password, base64 encoded.
bus.ldap.processor.port	6366	Port that the LDAP service connects to; the SSL option must be enabled.
bus.ldap.processor.user	cn=Directory Manager,cn=mydomain.com	User to sign in to LDAP.
bus.ldaprequest.timeout	10000 milliseconds	
bus.network.udp.ttl	1	Set the default TTL for

<b>Property</b>	<b>Default value</b>	<b>Description</b>
		multicast packets sent out on the multicast socket.
com.eibus.util.multicast.packet.size	2048	Set the size for multicast packets in bytes.
ldap.multicast.ip	multicast IP number	Specify a unique IP number to use for the multicast feature.
ldap.multicast.port	6366	Port to multicast on, which is the same as the port of the specific LDAP server.
ldap.multicast.timeout	30000	Property applicable in load balancing and failover scenarios.
ldap.root	cn=cordys,cn=defaultInst,cn=mydomain.com	DN represents the root DN of the LDAP server.

Property	Default value	Description
ldap.server	port=6366	Represented as <host>:<port> where host is the hostname and port is the port on which the LDAP server is listening.
ldap.soap.processor.dn	cn=LDAP,cn=LDAP Service,cn=soap nodes,o=system,cn=cordys,cn=defaultInst, cn=mydomain.com	LDAP DN to use for the service.

## Management Console properties

This topic describes the Management Console properties.

Property	Default value	Description
bus.administrator.pwd		A password, base64 encoded
bus.administrator.user	cn=Directory Manager,o=vanenburg.com	User to sign in

## Authentication and single sign-on properties

This topic describes the Platform Properties related to authentication, authorization, and single sign-on that an administrator can configure using the Management Console.

Various properties of the authentication process and configuration of the Single Sign-On component are set in the `wcp.properties` file.

This table shows the properties:

<b>Property name</b>	<b>Default value</b>	<b>Description</b>
com.eibus.web.authentication.authuserkey	AUTH_USER	Sets which variable contains the user identity. Third-party Web-based single sign-on solutions sometimes do not use the default AUTH_USER variable. Normally this does not need to be configured.
bus.identity.wsusername.expirytime	1800	Integer indicating the expiry time-frame for username tokens, containing a creation timestamp, in seconds.
bus.truststore.file	Default location of generated trust store:  <AppWorks_Platform_installdir>/certificates/trust_store/CordysDefaultTrustStore.p12	Location on the disk of the default trust store that is used by the monitor.
bus.truststore.password	Random generated	Password that protects the trust store of the monitor, encoded in Base64.
bus.keystore.file	Default location of generated key store:	Location on

<b>Property name</b>	<b>Default value</b>	<b>Description</b>
	<AppWorks Platform_installdir>/certificates/keystore/ <computername>_monitor.p12	disk of the default key store that is used by the monitor.
bus.keystore.password	Random generated	Password that protects the key store of the monitor, encoded in Base64.
bus.keystore.certificate.alias	monitor@<computername>	Certificate alias of the monitor certificate.
bus.keystore.privatekey.password	Random generated	Password that protects the private key in the key store of the monitor, encoded in Base64.
com.eibus.web.authentication.enableanonymousaccess	true	Enable anonymous access for the gateway. Do not disable this if you use Single Sign-On. Anonymous access is limited through ACLs.
com.eibus.security.x509.validation.certificate.default		Referenced from WS-Security. Default is the alias of the

<b>Property name</b>	<b>Default value</b>	<b>Description</b>
		default certificate. This certificate is used when no key information is present in a signature. If no KeyInfo element is present in the assertion, this property is used to resolve the certificate that signed the assertion.
com.eibus.security.samlassertion.artifact	true	By default, the SAML Assertions contain an element for SAML artifact. When this property is false, the saml artifact element is not added to the SAML assertions.
com.eibus.security.saml.cache.maxSize	1000	Number of SAML assertions cached by an SSO service container. It indicates the maximum

<b>Property name</b>	<b>Default value</b>	<b>Description</b>
		possible assertions that the service container can accommodate at any given moment. In the event of greater inflow, the limit is maintained by automatic removal of the least used assertions from the cache. For the removed assertions, the users have to log on afresh.
com.eibus.sso.artifactstore.size	10000	Number of SAML artifact cached by an SSO service container. The SAML artifact is a reference to the actual cached SAML assertion. This property indicates the maximum possible SAML artifacts that the service container can

<b>Property name</b>	<b>Default value</b>	<b>Description</b>
		accommodate at any given moment. In the event of greater inflow, the limit is maintained by automatic removal of the least used SAML artifact from the cache.
com.eibus.security.StrictIdPIsolation	false	Whether the user of any configured IdP needs to re-authenticate when switching between different organizations. Valid values are true and false. When true, the user has to provide the credentials to re-authenticate. When false, the user can switch to different organizations without re-authenticating. In that case, the

<b>Property name</b>	<b>Default value</b>	<b>Description</b>
		generated SAML assertion is not limited to a specific organization.

## XML parser (NOM) properties

The following properties available for the XML parser can be set in the `wcp.properties` file.

<b>Property</b>	<b>Default value</b>	<b>Description</b>
<code>bus.xml.nom.normalizedata.enabled</code>	true	<p>Sets the behavior of the XML parser while creating a data node. The possible values are:</p> <ul style="list-style-type: none"> <li>■ false <ul style="list-style-type: none"> <li>• The data content of an element node is placed in a single data node if it does not contain special characters such as non-ASCII characters and '\n'.</li> <li>• A separate data node is created for the special character if the data content of an element node contains a special character. In addition, two sibling data nodes are created for the content displayed before and after the special character.</li> </ul> </li> <li>■ true <ul style="list-style-type: none"> <li>• Characters in the XML data node are combined into a single node regardless of the special characters. Each node can contain characters up to the set buffer size. The default size is 5120 characters. If the character size exceeds the set buffer size, a new sibling data node is created.</li> <li>• Buffer size is set using</li> </ul> </li> </ul>

<b>Property</b>	<b>Default value</b>	<b>Description</b>
		bus.xml.nom.normalizedata.buffersize.
bus.xml.nom.normalizedata.buffersize	5120	Sets the maximum number of characters in a data node or buffer size. The buffer size is used when bus.xml.nom.normalizedata.enabled is true.
bus.xml.vm.maxsize	256 MB	Limits the maximum value of the virtual memory allowed to create XML nodes. Memory allocated for this purpose is shared by all the XML documents of the com.eibus.xml.nom.Document class. The virtual memory size is considered in megabytes (MB). By default, 256 megabytes (MB) of memory is allocated. The value of this property does not consider any suffix appended to it. <b>Tip:</b> Provide just the number without any suffix, for example, 512. Any suffix, such as 512k, 512M, 512 MB, 512m, and 512mb is converted to 512 MB.
bus.xml.nom.sharedmemorypool.size	true	Sets the size of the shared memory in the number of nodes per document when NOM memory pool is shared across multiple documents (bus.xml.nom.sharedmemorypool.enabled =true). The value specifies the number of unused nodes that can be kept private to a NOM document. Any unused node beyond this value is moved to the shared pool where other documents can also use it.
bus.xml.nom.suppress.invaliduri.error	false	Manages the errors when invalid URIs are used in the namespace declarations, for example, using a URI with a space. The default value is false. If the flag is set to true, the errors reported during parsing of such XML are suppressed and ignored.
bus.xml.nom.sharedmemorypool.enabled	true	When set to true, NOM enables shared

<b>Property</b>	<b>Default value</b>	<b>Description</b>
abled		memory pool mechanism that helps achieve efficient utilization of the NOM memory across various documents in a process. You can set the size of the shared memory available per document using the bus.xml.nom.sharedmemorypool.size property. The default value is 15000 nodes. If a document contains more than 15000 nodes that it is not using, the unused nodes are eligible for use by other documents. You can modify the size to control the memory available per document.
bus.xml.nom.preservecomments	true	When set to true, NOM preserves the comments during the XML parsing. When set to false, it does not load any comment while parsing the XML.
bus.xml.nom.preservespace	false	When set to true, NOM preserves the space between the two element nodes. For normal usage, OpenText recommends not using this option.
bus.xml.nom.exceptions.enabled	true	When set to true, NOM throws exceptions (com.eibus.xml.nom.XMLException) when a NOM operation results in an error. When set to false, NOM suppresses the error and does not throw any exception.

## UDDI connector properties

This table shows the UDDI connector properties.

<b>Property</b>	<b>Description</b>
uddi.http.connection.host.maximum	Maximum number of connections allowed for a provided host configuration.
uddi.http.connection.total.maximum	Maximum number of connections allowed for a provided connection manager.
uddi.keystore	Secure key store location, which is used by the UDDI Connector when communicating with https-enabled registries. This can also include a client

<b>Property</b>	<b>Description</b>
	certificate, if required.
uddi.keystore.password	Password for the key store file provided for the property <code>uddi.keystore</code> . Must be in base64 encoded format.

## Web gateway

### Properties

<b>Property</b>	<b>Default value</b>	<b>Description</b>
com.eibus.web.gateway.queuesize	1000	<p>Enables you to configure the waiting queue size for requests in the gateway. It considers an integer. Using this property, you can restrict the request queue size to 1000. The gateway processes 1000 requests at a time; the connection to the requests beyond 1000 is blocked until these requests are added to the queue. If this property is set to 0, the queue size is unlimited.</p> <p><b>Note:</b> This is applicable only to the Web server.</p>
com.eibus.web.gateway.timeout	30000	Specifies the time in milliseconds after which the gateway times out.
com.eibus.web.maxrequestsize	0	<p>Provides information to the Upload gateway about the maximum request size (in kilobytes), which decides if the request must be parsed in a single chunk or multiple chunks. In case of parsing by multiple chunks, the Upload gateway writes the uploaded file content to the temporary directory mentioned in the property <code>com.eibus.web.tools.upload.UploadWritePath</code>. For example, <code>com.eibus.web.maxrequestsize=128</code>.</p>

<b>Property</b>	<b>Default value</b>	<b>Description</b>
com.eibus.web.maxresponsesize	0	Denotes the maximum size of a response chunk in the Web gateway. The gateway response is split into multiples of maxresponsesize.
com.eibus.web.tools.download.DownloadReadPath		Provides information to the Download gateway about the directory containing the download file placed by the service container. See <a href="#">Changing the location of the upload and download folders</a> .
com.eibus.web.tools.upload.UploadWritePath		Provides information to the Upload gateway about the directory in which the temporary copy of the uploaded file is created. See <a href="#">Changing the location of the upload and download folders</a> .
com.eibus.web.wsdl.gateway.hostname (deprecated)		Composes the WSDL gateway's end point URL. If you set this property, the WSDL's end point URL contains the host name as specified by this property value. If this property is not set, the WSDL's end point URL contains the host name provided by the HTTP Request (if needed using headers, such as X-Forwarded-Host).
com.gateway.header.strip	true	Removes user information from SOAP requests or responses. The SOAP response headers contain details about their senders and recipients. This information can be used for sending malicious requests and in DoS attacks. Removing these details also increases the network bandwidth. By default, all details are removed and only the message ID is retained in the header.
cordys.gateway.inject.baseurl	true	Setting this property to true enables the injection of the base URL into

<b>Property</b>	<b>Default value</b>	<b>Description</b>
		the XForms. This is done by AppWorks Platform XGateway. Setting it to false might result in additional round trips while opening an XForm, thereby lowering the performance.
cordys.gateway.protocol.validation		Validates the SOAP protocol of a SOAP request.
cordys.gateway.request.validation		Validates the SOAP request of a SOAP message.
cordys.gateway.response.validation		Validates the SOAP response of a SOAP request.
web.root	<cordys home>/web root	Location of the Web server document root.
web.server.portnumber	80	Port number assigned to the Web server
web.server.vendor	Not set for tomcat	

## Classpath

The procedure to add a third party JAR to the Web gateway classpath is different for every Web server.

- Ensure that the JAR is accessible to the user who is accessing the Web server.
- Restart the Web server after adding the JAR to the classpath.

<b>Web server</b>	<b>Action</b>
TomEE	Place the third party JAR in the \$CORDYS_HOME/webroot/WEB-INF/lib directory.

## General properties

This table shows general WCP properties.

<b>Property</b>	<b>Description</b>
bus.jvm.heapsize	Controls the heap size of the JVMs started

<b>Property</b>	<b>Description</b>
	<p>by the Web gateway (ISAPI/mod_cordys) or the monitor service.</p> <p><b>Note:</b> This property is available for backward compatibility. If the property <code>bus.vm.options.&lt;component&gt;</code> is set, this property can be ignored.</p>
<code>bus.ldap.directory.schemadefinition</code>	<p>Controls the class name used for the LDAP schema. By default, OpenText CARS (OpenLDAP) schema is used. Use this property only when not using the standard OpenText CARS LDAP, which is currently not supported.</p>
<code>bus.ldap.processor.host</code>	<p>Contains the name of the currently used LDAP server.</p>
<code>bus.ldap.processor.keystore</code>	<p>Contains the keystore to use to connect to the LDAP server using an SSL connection.</p>
<code>bus.ldap.processor.maximumnoofsearchresult</code>	<p>Controls the maximum number of search results returned by the LDAP server.</p>
<code>bus.ldap.processor.password</code>	<p>Contains the password (Base64 encoded) to connect to the LDAP server.</p>
<code>bus.ldap.processor.port</code>	<p>Contains the port number on which the LDAP server is listening for connections.</p>
<code>bus.ldap.processor.ssl</code>	<p>Indicates whether the connection to LDAP is established using SSL.</p>
<code>bus.ldap.processor.user</code>	<p>Contains the username used to connect to the LDAP server.</p>
<code>bus.ldaprequest.timeout</code>	<p>Increases the time-out value of the LDAP service container. The value must be 180000 (milliseconds) when SSL for the LDAP service container is enabled. Default value is 10000 and used for LDAP entry searches.</p>
<code>bus.ldapwrapper.monitorinterval</code>	<p>Controls the interval during which the monitor tries to reach the LDAP server again after it is disconnected.</p>
<code>bus.monitor.ssu.statepublishtimeout</code>	<p>Identifies the interval for committing the transactions to the local registry. Default value is 2147483647. Value is specified in</p>

<b>Property</b>	<b>Description</b>
	milliseconds.
bus.queue.read.timeout	<p>Timeout in milliseconds meant for reading messages from a queue.</p> <p><b>Note:</b> This property is deprecated. Use the connection-point specific settings to control this.</p>
bus.services.guid.GuidFactory	<p>Allows you to influence the class that generates the GUIDs. These IDs are primarily used for generating the message ID for SOAP messages. To provide a different GUID factory, your custom factory must implement the interface <code>com.eibus.util.guid.GuidFactory</code>.</p> <p><b>Note:</b> Most of the classes in AppWorks Platform use <code>Native.createGuid()</code> to create their GUIDs.</p>
bus.socket.connectioninvalidtime	Specifies the time, which is added to the current time to indicate that a connection is not valid.
bus.socket.localloop	Used on systems that have a frequently changing IP address. If set to true, the entire AppWorks Platform environment uses 127.0.0.1 to access the SOAP processors.
bus.socket.minimumdowntime	Specifies the minimum time required to invalidate a middleware connection before it is recreated.
bus.ssl.keystore	Specifies the location of the keystore containing certificates to connect to LDAP.
bus.ssl.keystorepassword	Specifies the password of the keystore containing certificates to connect to LDAP.
bus.ssl.truststore	Specifies the location of the keystore containing the trusted certificates. This store validates the LDAP Server certificate.
bus.ssl.truststorepassword	Specifies the password of the keystore containing the trusted certificates. This store validates the LDAP Server

<b>Property</b>	<b>Description</b>
	certificate.
bus.transport.socket.sotimeout	Specifies the timeout for reading from a socket. Default value is 100000 (100 Seconds). Value should be specified in milliseconds.
bus.vm.debug.<component>	Makes the WCP monitor service or the ISAPI extension pop up a dialog box. Name of the component can be monitor or debug. <b>Note:</b> Windows only.
bus.vm.options.default	Passes Java Virtual Machine (JVM) or profiler arguments. The arguments are passed after the JVM is loaded. However, the -server and -client arguments are not passed if the JVM is loaded. <b>Note:</b> For any operating system, the default is server-jvm. If server-jvm is not present, it considers client-jvm.
com.eibus.management.jmxPassword	Password required for the user to sign in.
com.eibus.management.jmxUser	To enable management through JMX. User is a chosen name.
com.eibus.management.rmiregistry.port	Contains the TCP port number on which the RMI registry is exposed. The RMI registry contains all the JMX components.
com.eibus.management.useHierarchicalName	Builds a list of key properties for the object name of this component. The names follow the JMX best practice naming convention as described in <a href="http://java.sun.com/products/JavaManagement/best-practices.html">http://java.sun.com/products/JavaManagement/best-practices.html</a> .
com.eibus.port.range	AppWorks Platform must use some ports to communicate properly, especially when running as a cluster. This property defines which ports are used by AppWorks Platform. The property can be set as a-b, where a is inclusive and b is exclusive. The default value is 10000-32767; if the specified range is not valid, the default range is considered.

Property	Description
com.eibus.port.range.anonymous	AppWorks Platform also uses some anonymous ports (used for short-lived communication) to communicate properly, especially when running as a cluster. This property defines which anonymous ports are used by AppWorks Platform. The property can be set as a-b, where a is inclusive and b is exclusive. The default value is all ephemeral ports (it differs per operating system); if the specified range is not valid, the default range is considered.
com.eibus.processor.outputstream.buffersize	An output buffer of 2 KB is stored and is used when JVM crashes and is not loaded. This size of output buffer can be configured by setting this property.
com.cordys.scheduler.engine.load.async	When set to true, it enables the active schedules in the Business Process Management service to be loaded asynchronously by default at start up. However, to load the active schedules in the Business Process Management service synchronously, set the property to false.
com.eibus.scpjar.name	<p>Enables you to provide the list of your custom JARs to load AppWorks Platform Framework instead of the default JARs. Set the property as:</p> <pre>com.eibus.scpjar.name = &lt;name of the jar list&gt;</pre> <p><b>Caution:</b> Replaces the classpaths provided in the list by AppWorks Platform Framework and may jeopardize the loading of AppWorks Platform. Therefore, OpenText recommends adding <code>cordyscp.jar</code> that contains the list of CordysFramework JARs to the extended JAR.</p>
com.eibus.security.ac.dll	<p>Specifies the name of the DLL or SO file that contains the security routines.</p> <p><b>Note:</b> Do not change this property</p>

<b>Property</b>	<b>Description</b>
	unless explicitly needed. It can cause the system to stop functioning.
com.eibus.transport.groupmembership.udp.mxpacketsize	Contains the UDP packet size for the multi-cast packets sent by AppWorks Platform. The minimum size is 4096.
com.eibus.transport.groupmembership.udp.ttl	Sets the time-to-live on the MultiCast socket. The value ranges from 0 to 255.
com.eibus.transport.middleware.queue.size	Indicates the maximum size for the inbound queue. The default is the value of com.eibus.transport.middleware.threads times a thread-multiplier. By default, this thread-multiplier is set to 1. The XForms service container uses a thread-multiplier of 4. This means the XForms connector can hold 40 (default value of property com.eibus.transport.middleware.threads) inbound requests.
com.eibus.transport.middleware.threads	<p>Sets the number of worker threads for each configured middleware. Default is 10. This means that each service container can handle 10 requests concurrently. If you need to handle more requests concurrently, do any of the following:</p> <ul style="list-style-type: none"> <li>▪ Increase this property</li> <li>▪ Scale up by adding a new service container</li> </ul> <p><b>Note:</b> By setting this property in wcp.properties, all the service containers on this computer use this value. You can change the value for a single service container by setting – Dcom.eibus.transport.middleware.threads=20 in the additional JVM parameters.</p>
com.eibus.transport.msmq.dll	<p>Specifies the name of the DLL/SO file that contains the msmw routines.</p> <p><b>Caution:</b> Do not change this property unless explicitly needed. It can stop the</p>

<b>Property</b>	<b>Description</b>
	functioning of the system.
com.eibus.util.multicast.packet.size	Defines the packet size for SOAP-based multicasting.
com.eibus.util.remotesyncup.maxapplicationpendingmessages	Defines the queue size for the messages received by the Remote Syncup. <b>Caution:</b> This property must not be changed and is useful only for debugging.
com.eibus.util.sleep.roundedvalue	Sets the scaling for sleep times. Sleep times below 10 ms can cause the Windows clock to run faster. OpenText recommends setting at least 50 ms. If you pass a value of 120 to the com.eibus.util.SleepWrapper.roundedSleep() method, it returns a value of 100. This means the sleep time is rounded to a multiple of the value of com.eibus.util.sleep.roundedvalue.
com.eibus.web.license.LicReportGateway.licuser	Sets the AppWorks Platform user for use by the license report gateway. The default is thewcpLicUser.
com.opentext.process.platform.statement.cache.size	Sets the prepared statement cache size of the database connection pool. The default value for the Oracle database vendor is 0; for other vendors, it is 500.
com.opentext.process.platform.enable.distributed.transaction	This property can be used to change the database connection type to XA. For information about the settings to be done on the database server for XA connections, see <a href="#">Configuring AppWorks Platform service containers for distributed transaction</a> .
Con:bus.ldap.cache.maxSize	Controls the local LDAP cache for each process. Reduces the amount of LDAP entries to cache. By default, all entries are cached, since the default size is 0x7ffffffffffffL.
cordys.baseurl.protocol	If the AppWorks Platform server is configured with a proxy server, determines whether the communication

<b>Property</b>	<b>Description</b>
	between the client and proxy servers is HTTPS or HTTP.
java.home	Path to a JRE used to start OpenText AppWorks Platform (<instance name>) and service containers. If a JDK is installed, the path to the JRE inside the JDK must be specified.
ldap.cache.size	Controls the number of DNs in the local LDAP cache. Each process that uses AppWorks Platform connectors have an instance of this cache. <b>Note:</b> This property is deprecated and must not be used.
ldap.cache.timeout	Specifies the LDAP cache timeout. <b>Note:</b> This property is deprecated and must not be used.
ldap.multicast.ip	Sets the multicast IP address for the multicast ring between the LDAP service containers. The default value is the IP address of the server specified by ldap.server.
ldap.multicast.port	Specifies the multicast port for the LDAP service containers to communicate with each other. The default value is 3899 and is a UDP Port.
ldap.port	Specifies the port on which the LDAP server is running. For non-SSL, it is port 3899. Usually OpenText CARS running in SSL mode uses port 6366.
ldap.root	Specifies the root DN for the LDAP server to find the AppWorks Platform schema. The property is set by the installer and defaults to cn=cordys,o=<name-of-the-network-suffix>.
ldap.server	Sets the name of the LDAP server that is used.
ldap.soap.processor.dn	Contains the LDAP DN of the LDAP service container. The monitor uses this property to find the configuration details of the

<b>Property</b>	<b>Description</b>
	LDAP service container to enable it to start this service container before the other service containers.
log.config.file	Specifies the location of the Log4J configuration file that is used to initialize the logger.
sslsocket.public.keystore.pwd	Specifies the password for the truststore. The value must be in Base64 Encode format.
sslsocket.public.keystore.type	Specifies the type for the keystore. It defaults to the JKS format.

## Debugging

This topic describes the properties for debugging.

The following table contains the debugging properties:

<b>Property</b>	<b>Value</b>	<b>Description</b>
bus.vm.options.monitor	-agentlib:jdwp=transport=dt_socket,server=y,suspend=n,address=8001	Debug the monitor.

## SSL options on OpenText CARS

This table show the SSL options configured by the installer.

<b>Property</b>	<b>Default value</b>	<b>Description</b>
bus.ldap.processor.ssl	True	Connect OpenText CARS in secure mode.l
bus.ssl.keystore	<AppWorks_Platform_installdir>\certificates\keystore\keystore.jks	Location of the key store file.
bus.ssl.keystorepassword	<base64 encoded password>	Password for the key store file specified for the

<b>Property</b>	<b>Default value</b>	<b>Description</b>
		property ssl key store.
bus.ssl.truststore	<AppWorks_Platform_installdir>\certificates\truststore\truststore.jks	Location of the trust store file.
bus.ssl.truststorepassword	<base64 <encoded password>	Password for the trust store file specified for the property ssl key store.

## License properties

This table lists the license properties.

<b>Property</b>	<b>Default value</b>	<b>Description</b>
com.eibus.license.LogDir.key	<AppWorks_Platform_installdir/license/log/key>	Directory to store the license keys received.
com.eibus.license.LogDir.report	<AppWorks_Platform_installdir/license/log/report>	Directory to store the reports sent.
com.eibus.license.MasterLogDir.key	<AppWorks_Platform_installdir/license/master/log/key>	Directory to store the master license keys received.
com.eibus.license.MasterLogDir.r	<AppWorks_Platform_installdir/license/master/log/r	Directory to store

<b>Property</b>	<b>Default value</b>	<b>Description</b>
report	<report>	the master reports sent.
com.eibus.license.LicMasterStoreDir	<AppWorks_Platform_installdir/license/masterstore>	Directory to store the license information for contributors.
com.eibus.license.installedDate	<Date>	Date on which AppWorks Platform is installed.

## Port ranges within AppWorks Platform

A networked server, must use server ports to communicate between Java processes, especially when running as a cluster. It uses two types of ports:

- Configured ports, that are reused after a restart
- Anonymous ports

The range of ports that must be taken can be limited, if needed. Modify the following properties in [General properties](#).

<b>Property</b>	<b>Description</b>	<b>Default value</b>
com.eibus.port.range	For configured ports, only the ports within this range are used.	10000-32767
com.eibus.port.range.anonymous	For anonymous ports, only the ports within this range are used.	Ephemeral port range as per the operating system (see the table below)

### Example

com.eibus.port.range=10000-32767

com.eibus.port.range.anonymous=49152-65535

**Note:** Do not overlap the port ranges of the two types of ports, as it may lead to unexpected results.

The following table lists the OS specific ranges of ephemeral ports:

OS	Ephemeral port range	URL
Linux	Most of the Linux kernels use the port range 32768 to 61000.	<a href="https://en.wikipedia.org/wiki/Ephemeral_port">https://en.wikipedia.org/wiki/Ephemeral_port</a>
Windows	Windows Server 2008 and later versions use a default range of values 49152 to 65535.	<a href="https://support.microsoft.com/en-in/help/929851/the-default-dynamic-port-range-for-tcp-ip-has-changed-in-windows-vista">https://support.microsoft.com/en-in/help/929851/the-default-dynamic-port-range-for-tcp-ip-has-changed-in-windows-vista</a>

## ***Backwards compatibility***

Previously, the ports that influence the port ranges were set as follows:

### **Anonymous port ranges (deprecated)**

If no value is provided for the anonymous range, it is retrieved from:

```
bus.anonymous.ports.minLength=  
bus.anonymous.ports.maxLength=
```

## Chapter 34

# Application Package Deployment utility

The Application Package Deployment utility is a collection of Ant tasks that facilitates the deployment of application packages. It also helps in the creation and management of service group, service container, and database configuration. Using these Ant tasks, you can also automate some of the manual tasks performed while loading application packages.

**Note:** The Application Package Deployment utility supports the .cap application package deployment format and does not conform to the application package format of previous versions.

## List of Ant tasks

Name	Description
deployapplicationpackage - Ant task	Application package deployment task
undeployapplicationpackage - Ant task	Application package undeployment task
createdatabaseconfiguration - Ant task	Database configuration creation task
deletedatabaseconfiguration - Ant task	Database configuration deletion task
createservicegroup - Ant task	Service group creation task
deleteservicegroup - Ant task	Service group deletion task
createservicecontainer - Ant task	Service container creation task
deleteservicecontainer - Ant task	Service container deletion task
createconnectionpoint - Ant task	Connection point creation task
deleteconnectionpoint - Ant task	Connection point deletion task
cloneservicecontainers - Ant task	Task to clone service containers

# Configuring the server to use the Ant tasks

## Before you begin:

- The user who is executing the script from a command prompt or terminal client:
  - Must be present either in LDAP as an AppWorks Platform user or one of the existing user's OS identities must be mapped with this user
  - Must have setupUser role assigned

## To configure the server to use the Ant tasks:

1. Depending on your operating system, perform the following configurations:

### Windows

- Create an environment variable called CORDYS\_HOME and point it to <AppWorks Platform\_installdir>/InstanceName.
- Add cordyscp.jar to system classpath.
- Add the AppWorks Platform libraries path (CORDYS\_HOME/lib) to PATH.

### Linux

- Create an environment variable called CORDYS\_HOME and point it to <AppWorks Platform\_installdir>/InstanceName.
- Add cordyscp.jar to system classpath.
- Add the AppWorks Platform libraries (CORDYS\_HOME/lib) to LD\_LIBRARY\_PATH.

2. Use the Ant tasks in the build script file.

The above Ant tasks are custom tasks provided for use along with the AppWorks Platform installation. Since these tasks are not part of the standard Ant utility, they need to be specifically included into the Ant script for the task to be invoked correctly.

Include the following lines in your build script file.

```
<property environment="env"/>
<property name="bop.install.dir" value="${env.CORDYS_HOME}"/>
<taskdef resource="com/cordys/deployment/ant/taskdef.xml"
classpath="${bop.install.dir}/components/anttasks/anttasks.jar" />
```

3. View the detailed error messages to debug these Ant tasks.

Detailed logging for the utility is available through the Log4j framework. The Log4jConfiguration.xml file is available at CORDYS\_HOME/config and can be viewed in the console or in a logfile as shown below:

Add one of the following to the Log4jConfiguration.xml file:

**View in console** - Prints messages to the console.

```
<appender name="ConsoleAppender" class="org.apache.log4j.ConsoleAppender">
<layout class="org.apache.log4j.PatternLayout">
    <param name="ConversionPattern" value="%-4r [%t] %-5p %c %x - %m%n" />
</layout>
</appender>
<category name="com.cordys.deployment.ant">
<priority value="debug"/>
    <appender-ref ref="ConsoleAppender"/>
</category>
```

**View in logfile** - Logs files to the file General.xml located in the CORDYS\_HOME/Logs folder.

```
<category name="com.cordys.deployment.ant">
    <priority value="debug"/>
</category>
```

## Sample build file

### build.xml

```
<project basedir=". " default="testAll" name="Test-Antscripts">
    <!-- Run tests based on these properties -->
    <property file="test.properties"/>
    <!-- Add the task definition to include the custom tasks -->
    <property environment="env"/>
    <property name="bop.install.dir" value="${env.CORDYS_HOME}"/>
    <path id="anttask.classpath">
        <pathelement
location="${bop.install.dir}/components/anttasks/anttasks.jar"/>
    </path>
    <taskdef resource="com/cordys/deployment/ant/taskdef.xml">
        <classpath refid="anttask.classpath"/>
    </taskdef>
    <!-- call the tests -->
    <target name="testAll">
        <antcall target="testApplicationPackageDeploy"/>
        <antcall target="testApplicationPackageUndeploy"/>
    </target>
    <target name="testApplicationPackageDeploy">
        <deployapplicationpackage buildNumber="${cordys.buildnumber}">
            failonerror="${cordys.failonerror}"
            ldapRoot="${cordys.ldaproot}"
            name="${cordys.applicationpackagename}"
            port="${cordys.port}" protocol="${cordys.webprotocol}"
            revertOnfailure="${cordys.revertonfailure}"
            server="${cordys.server}" timeout="${cordys.timeout}"
```

**build.xml**

```

        version="${cordys.version}" webserver="${cordys.webserver}"/>
    </target>
    <target name="testApplicationPackageUndeploy">
        <undeployapplicationpackage failonerror="${cordys.failonerror}"
            ldapRoot="${cordys.ldaproot}"
            name="${cordys.applicationpackagename}" server="${cordys.server}"/>
    </target>
</project>
```

**test.properties**

```

#General properties.
cordys.server=dummy
cordys.ldaproot=cn=cordys,cn=defaultInst,o=cordys.com
cordys.failonerror=true

#Package Deployment properties
cordys.webserver=
cordys.webprotocol=http
cordys.port=80
cordys.applicationpackagename=ABC
cordys.timeout=30000
cordys.revertonfailure=false
cordys.version=2.0
cordys.buildnumber=0
```

## cloneservicecontainers - Ant task

The `cloneservicecontainers` Ant task clones a list of service containers to the target computer. The list of service containers to be cloned is identified based on the search criteria definition. The search criteria can be defined across the set of organizations and the Shared space. This task can be used to clone the service containers to the same computer or to a different node in a cluster. In the latter case, the task can be executed from any node in a cluster other than the target node.

### Prerequisites

Before cloning a Collaborative Workspace (CWS) service container to another node, it is assumed that the node-independent configuration settings of the target node match those of the source node.

**Note:** For multi-node AppWorks Platform clusters, the CWS synchronization and build folder must be configured on a shared file system. The [AppWorks Developer community](#) provides additional resources that you may find helpful.

## Sample usage

A sample usage of the `cloneservicecontainers` task follows.

```
<cloneservicecontainers containerNameSuffix="UniqueSuffixIdentifier"
    sourceNodeName="serverA" targetNodeName="serverB">
    <searchCriteria>
        <organizations>
            <include>system</include>
            <include>Organization B</include>
            <exclude>Organization C</exclude>
            <exclude>Organization D</exclude>
        </organizations>
        <serviceGroups>
            <include>*</include>
            <exclude>Collaborative Workspace</exclude>
        </serviceGroups>
    </searchCriteria>
</cloneservicecontainers>
```

## Parameters

Attribute	Description	Type	Required
sourceNodeName	Name of the computer where the configuration of the service containers is read for cloning	String	Yes
targetNodeName	Name of the computer where the cloned service containers are created.	String	Yes
containerNameSuffix	A unique identifier value that is appended to the name of the existing service containers. If the name of the service container to be cloned is MyContainer and the containerNameSuffix value is set to Clone2, the new name of the cloned service container is MyContainerClone2.	String	Yes

## Parameters as nested elements

The Ant task has a set of child elements to define a search criteria. The search criteria enables identification of the set of service containers across the organizations and the shared space, and cloning them to the target computer.

## searchCriteria

The searchCriteria element contains the two child elements described in the following sections.

### *organizations*

This is an optional element that has two nested elements: include and exclude. Use the include element to clone the service containers of an organization. To exclude an organization from cloning, add an exclude element with the name of the organization as its value. You can define any number of include and exclude elements. You can also use simple search patterns to filter the list of organizations.

#### **Usage pattern:**

1. To include all the organizations without any exclusions, remove the searchCriteria element:

```
<cloneservicecontainers containerNameSuffix="UniqueSuffixIdentifier"
    sourceNodeName="serverA" targetNodeName="serverB"/>
```

2. To clone all the service containers in Organization A but not in Organization B and C:

```
<searchCriteria>
    <organizations>
        <include>Organization A</include>
        <exclude>Organization B</exclude>
        <exclude>Organization C</exclude>
    </organizations>
</searchCriteria>
```

3. If there are five organizations and you need to clone the service containers only in Organization A, the search criteria definition below excludes all the other organizations:

```
<searchCriteria>
    <organizations>
        <include>Organization A</include>
    </organizations>
</searchCriteria>
```

4. To clone the service containers in all the organizations whose name starts with Ame:

```
<searchCriteria>
    <organizations>
        <include>Ame*</include>
    </organizations>
</searchCriteria>
```

## serviceGroups

The serviceGroups element has two nested elements: include and exclude. Use the include element to add the name of the service groups whose service containers must be cloned. To exclude a service group from cloning, add an exclude element with the name of the service group as its value. You can define any number of include and exclude elements. You can also use simple search patterns to filter the list of service groups.

### Note:

- The serviceGroups element and its child elements are applicable to all the organizations defined in the searchCriteria filter for the organizations.
- If there is more than one service container for a specific service group, cloning is done for all the service containers.
- Cloning is skipped if a service container to be cloned already exists in LDAP.
- If there is an OS Process configured for a service container to be cloned to a target node, the corresponding OS Process is created on the target node if it does not exist.
- The Monitor service container is not cloned.
- If you have service groups with simpleFailover as the routing algorithm and clone the service containers associated with them, the cloned service containers will have the same preference values. The system administrator must review the preference values and assign values accordingly using the System Resource Manager.

### Usage pattern:

1. If the name of the service group is defined as 'Business Process Management' to clone the corresponding service container:

```
<include>Business Process Management</include>
```

2. Remove the serviceGroups element to include all the service groups without any exclusions.

**Note:** The cloning of the LDAP Service through the Ant task is not supported. It is recommended to perform it manually through the System Resource Manager. Refer to the *AppWorks Platform High Availability Deployment Guide* for more information on the manual steps.

## Supported search patterns

The following search patterns are case-sensitive:

Pattern	Description
*	Any string

<b>Pattern</b>	<b>Description</b>
system*	Any string beginning with 'system'
*system	Any string ending with 'system'
system	Exact name

## createconnectionpoint - Ant task

The `createconnectionpoint` Ant task creates a connection point for the specified service container.

### Sample usage

The following code shows the procedure to create a connection point called `MyConnectionPoint` for the service container called `WS-AppServer` container.

```
<createconnectionpoint
    ldapRoot="cn=cordys,cn=bopcu7,o=myorg.com"
    name="MyConnectionPoint"
    organization="myorganization"
    override="true"
    server="dab202409"
    servicecontainerName="WS-AppServer container"
    servicegroupName="WS-AppServer Test"/>
```

## Parameters

<b>Attribute</b>	<b>Description</b>	<b>Type</b>	<b>Required</b>
name	Name of the Connection Point.	String	Yes
organization	Name of the organization in which the connection point will be created.	String	Yes
servicegroupName	Name of the Service Group that contains the Service Container.	String	Yes
servicecontainerName	Name of the Service Container under which the Connection Point will be created.	String	Yes
connectionType	Indicates the type of Connection Point. Possible value: ■ TCP/IP	String	No
protocol	Possible values are:	String	No

<b>Attribute</b>	<b>Description</b>	<b>Type</b>	<b>Required</b>
	<ul style="list-style-type: none"> <li>■ socket (both compression and ssl disabled) (default)</li> <li>■ zipsocket (compression enabled)</li> <li>■ sslsocket (ssl enabled)</li> <li>■ zipsslsocket (both compression and ssl enabled)</li> </ul>		
port	Possible values are: <ul style="list-style-type: none"> <li>■ auto ( Automatically connects to the available port) (default)</li> <li>■ any valid port number</li> </ul>	Integer	No
server	Name of the server of AppWorks Platform instance.	String	Yes
ldapRoot	LDAP root of the AppWorks Platform instance.	String	Yes
override	Property to update the existing connection point.	Boolean	No
failOnError	Boolean flag aborts the build when an exception occurs in the task.	Boolean	No

## createdatabaseconfiguration - Ant task

The `createdatabaseconfiguration` Ant task creates a database configuration based on the parameters provided.

### Parameters

<b>Attribute</b>	<b>Description</b>	<b>Type</b>	<b>Required</b>
name	Name of the database configuration.	String	Yes
description	Description of the database configuration. If the value is empty, name will be taken as description.	String	No
server	Name of the system where AppWorks Platform is installed.	String	Yes
ldapRoot	LDAP Root DN of AppWorks Platform instance.	String	Yes
organization	Name of the organization where database	String	Yes

<b>Attribute</b>	<b>Description</b>	<b>Type</b>	<b>Required</b>
	configuration needs to be created.		
testConnectivity	<ul style="list-style-type: none"> <li>■ If true, checks the status of database connection and creates the database configuration only if the connection is successful.</li> <li>■ If false, skips the connection status and creates database configuration directly.</li> </ul> <p>Default value is true.</p>	Boolean	No
override	Flag to update the database configuration if it is already present. Default value is false.	Boolean	No
failOnError	Flag that aborts the build when an exception occurs in the task. Default value is true.	Boolean	No

## Parameters for database and driver

It has two nested elements called 'database' and 'driver'. These attributes are described in the following table.

### *Attributes of database element*

<b>Attribute</b>	<b>Description</b>	<b>Type</b>	<b>Required</b>
defaultDatabaseName	The name of the database. For Oracle, the attribute is the schema name.	String	Yes
dbUserName	The database user ID/ user name.	String	Yes
dbPassword	The database user password.	String	Yes
createDatabase	Indicates whether a new database should be created while creating the database configuration. Default value is false.	Boolean	No
dbaName	Database administrator name to create a new database. This property is needed only if createDatabase is set to true.	String	No
dbaPassword	Database administrator password to create a new database. This property is needed only if createDatabase is set to	String	No

<b>Attribute</b>	<b>Description</b>	<b>Type</b>	<b>Required</b>
	true.		
createTablespace	A flag that is specific to Oracle and createDatabase = true. This should be set to true if tablespace is created while creating database, otherwise set to false. Default value is false.	Boolean	No
tablespace	Tablespace name. This is needed only if createDatabase = true, createTablespace = true and dbVendor = Oracle.	String	No
tablespacePath	Tablespace path. This is needed only if createDatabase = true, createTablespace = true and dbVendor = Oracle.	String	No

### **Attributes of driver element**

**If the driver is JDBC:**

<b>Attribute</b>	<b>Description</b>	<b>Type</b>	<b>Required</b>
dbVendor	Name of the database vendor. Currently supported values are: <ul style="list-style-type: none"> <li>■ MS SQL Server 2005</li> <li>■ MS SQL Server 2008</li> <li>■ Oracle 11g</li> <li>■ Oracle 12c</li> <li>■ PostgreSQL</li> <li>■ Others</li> </ul>	String	Yes
dbConnectorType	Type of the connector used. Possible value is JDBC.	String	Yes
jdbcDriverClass	Qualified name of the JDBC driver class. If it is empty, the default values will be: <ul style="list-style-type: none"> <li>■ oracle.jdbc.driver.OracleDriver if dbVendor is Oracle</li> <li>■ com.microsoft.sqlserver.jdbc.SQLServerDriver if dbVendor is Microsoft SQL Server</li> <li>■ org.postgresql.Driver if dbVendor is PostgreSQL</li> </ul>	String	No

<b>Attribute</b>	<b>Description</b>	<b>Type</b>	<b>Required</b>
jdbcDriverXAClass	<p>Qualified name of the JDBC driver XA class. If it is empty, the default values are:</p> <ul style="list-style-type: none"> <li>■ oracle.jdbc.xa.client.OracleXADataSource if dbVendor is Oracle</li> <li>■ com.microsoft.sqlserver.jdbc.SQLServerXADataSource if dbVendor is Microsoft SQL Server</li> <li>■ org.postgresql.xa.PGXADatasource if dbVendor is PostgreSQL</li> </ul>	String	No
connectionString	<p>For Oracle OCI, the value will be  <code>jdbc:oracle:oci:@&lt;TNS NAMES ALIAS&gt;</code>  TNS NAMES ALIAS =&gt; tnsnames.ora file entry</p> <p>For Oracle Thin, the value will be  <code>jdbc:oracle:thin:@&lt;db_server_name&gt;:&lt;port_no&gt;:&lt;SID(or) Service_Name&gt;</code></p> <p>For MS SQL Server , the value will be  <code>jdbc:sqlserver://&lt;db_server_name&gt;:&lt;port_no&gt;</code></p> <p>For PostgreSQL, the value will be  <code>jdbc:postgresql://&lt;&lt;db_server_name&gt;&gt;:&lt;&lt;port_no&gt;&gt;/&lt;&lt;db&gt;&gt;</code></p>	String	Yes

**Note:** A new database cannot be created if ODBC Connection is used.

## createservicegroup - Ant task

The `createservicegroup` Ant task creates a service group in an organization of AppWorks Platform instance.

### Sample usage

The following example shows how to create a service group WS-AppServer Test in the organization myorganization.

Add the following XML code to `build.xml` to run the service group deployment silently through Ant task.

```
<createservicegroup name="WS-AppServer Test" organization="myorganization"
ldapRoot="cn=cordys,cn=bopcu7,o=myorg.com" override="false"
ignoreWhiteSpaces="false" routingUIAlgorithm="failover"
routingUIType="loadbalancing" routingNumProcessors="10000"
```

```

routingAlgorithm="com.eibus.transport.routing.DynamicRouting"
validationProtocol="true" validationPayload="true">> <WebserviceInterface
name="cn=WS-AppServer Development Method Set,cn=Cordys WS-AppServer"
namespace="http://schemas.cordys.com/WS-AppServerDevelopment/1.0" />
</createservicegroup>

```

## Parameters

Attribute	Description	Type	Required
name	Name of the service group.	String	Yes
organization	Name of the organization.	String	Yes
description	Description of the service group.	String	No
ignoreWhiteSpaces	An indicator of whether to ignore white spaces. Default value is false.	Boolean	No
routingUIAlgorithm	Routing UI algorithm value. Possible values are: <ul style="list-style-type: none"><li>■ failover</li><li>■ classic</li></ul> Default value is failover.	String	No
routingUIType	Routing UI type value. Possible values are: <ul style="list-style-type: none"><li>■ loadbalancing</li><li>■ failover</li><li>■ classic</li></ul> Default value is loadbalancing.	String	No
routingNumProcessors	Number of processor value. Default value is "10000".	Integer	No
routingAlgorithm	Routing_algorithm qualified class name. Possible values are: <ul style="list-style-type: none"><li>■ com.eibus.transport.routing.Classic Routing</li><li>■ com.eibus.transport.routing.Dynamic Routing</li></ul> Default value is com.eibus.transport.routing.DynamicRouting.	String	No
validationProtocol	If set to true, the request is validated against	Boolean	No

<b>Attribute</b>	<b>Description</b>	<b>Type</b>	<b>Required</b>
	the SOAP and AppWorks Platform Protocol schemas. The default value is false.	an	
validationPayload	If set to true, the request is validated against the Schema in WSDL of the AppWorks Platform Web service operation. The default value is false.	Boolean	No
authenticatorImplementation	Authenticator implementation's qualified class name.	String	No
ldapRoot	LDAP root of the AppWorks Platform instance.	String	Yes
override	Property to update a Service group. If set to true, the existing service group will be overwritten with the latest configuration information. Default value is set to false.	Boolean	No
failOnError	Aborts the build when an exception occurs in this task. The default value is true.	Boolean	No

## Parameters for WebserviceInterface

This attribute has one nested element WebserviceInterface which should contain details of the Web service interfaces to be attached to the Service Group. There should be at least one WebserviceInterface to attach to this Service group. This element should be repeated for every method set that needs to be attached. The attributes are explained below.

### Attributes of WebserviceInterface element

<b>Attribute</b>	<b>Description</b>	<b>Type</b>	<b>Required</b>
name	Names of the method sets along with the package name (excluding the LDAP root).	String	Yes
namespace	Namespace of the method set that should be attached to this service group.	String	Yes

## createservicecontainer - Ant task

The `createservicecontainer` Ant task creates a service container in an organization of AppWorks Platform instance.

## Sample usage

The following example describes a procedure to create a Service Container WS-AppServer container in Service Group WS-AppServer Test group which is available in myorganization.

```
<createservicecontainer abortTime="4" autoStartCount="3"
    gracefulCompleteTime="14" ldapRoot="cn=cordys,cn=fplcu7,o=myorg.com"
    name="WS-AppServer container" organization="myorganization"
    override="true" server="cin4005991" servicegroupName="WS-AppServer Test"
    startUpType="true" timeOut="40000">
    <jreconfig>
        <param value="-Xmx256M"/>
        <param value="-XX:PermSize=5m"/>
        <param value="-cp /opt/jopl.jar"/>
    </jreconfig>
    <applicationConnector
        configurationFile="D:\applicationConnectorConfig.xml"
        htmlfile="/cordys/com/cordys/wsappserver/admin/wsappservercodnfig.caf"
        implementation="com.cordys.cpc.bsf.connector.BsfConnector"/>
</createservicecontainer>
```

## Parameters

Attribute	Description	Type	Required
name	Name of the Service Container.	String	Yes
organization	Name of the organization in which Service Group should be created.	String	Yes
servicegroupName	Name of the Service Group.	String	Yes
osProcess	Name of the OS process.	String	No
startupType	If the value is true, the Service Container starts automatically. If the value is false, start the Service Container manually.  <b>Note:</b> After creating the Service Container using Ant task, auto start requires the Service Container to be restarted.	Boolean	No
timeout	Timeout value for handling requests of incoming SOAP requests. Default value is 30000 (ms).	Integer	No
ldapRoot	LDAP root of the AppWorks Platform instance.	String	Yes

<b>Attribute</b>	<b>Description</b>	<b>Type</b>	<b>Required</b>
server	Name of the server of AppWorks Platform instance where the Service Container should be created (used for distributed environment).	String	Yes
override	Property to update a Service Container. If set to true, the existing Service Container configuration will be overwritten with the latest configuration information	Boolean	No
autoStartCount	Number of attempts to auto-start a Service Container. Default number of attempts is 3. To set infinite, set the value to 0.	Integer	No
gracefulCompleteTime	Time in seconds to complete SOAP transactions gracefully. Default value is 15 (sec).	Integer	No
abortTime	Time in seconds to abort SOAP transactions.	Integer	No
failOnError	A flag that indicates whether to abort the build when an exception occurs in this task.	Boolean	No

## Parameters as nested elements

This task has two nested elements applicationConnector and jreConfig. The properties of these elements are described below.

### ***applicationConnector***

The Ant task can contain any number of applicationConnector elements. The minimum number of occurrences should be 1.

<b>Attribute</b>	<b>Description</b>	<b>Type</b>	<b>Required</b>
implementation	Fully qualified name of the application connector class.	String	Yes
<a href="#">configurationFile example</a>	File path of the configuration file that contains application connector specific properties.	String	No
htmlfile	Qualified name of the caf file that is used for providing application connector configuration details.	String	No

## jreconfig

- The element jreconfig is optional. Maximum number of occurrences is 1.
- It has a child element called param. The param element can be defined any number of times. It has one property called value, which is mandatory. Provide the JRE parameter of the Service Container here.

## configurationFile example

Every application connector has a configuration xml associated with it. The configuration xml is different for different application connectors, based on the information required for the application connector. This configuration xml for each application connector can be obtained from LDAP after creating a service group with the application connector.

A sample configuration xml for one of the application connector follows.

### applicationConnectorConfig.xml

```
<configuration>
    <startupDependency>
        <namespace>http://schemas.cordys.com/1.0/xmlstore</namespace>
    </startupDependency>
    <ruleEnabled>false</ruleEnabled>
    <auditEnabled>off</auditEnabled>
    <initializeDB>true</initializeDB>
    <multiTenant>false</multiTenant>
    <initializeEIS>false</initializeEIS>
    <customInitializer/>
    <applicationInitializer/>
    <autoCleanup>false</autoCleanup>
    <!-- configuration for database connection pool -->
    <component name="WS-AppServerConnectionPool">
        <xmlEncoding>false</xmlEncoding>
        <precedence/>
        <cursorCacheSize>50</cursorCacheSize>
        <cursorCacheRefreshInterval>30</cursorCacheRefreshInterval>
        <queryCacheSize>50</queryCacheSize>
        <queryCacheRefreshInterval>3600</queryCacheRefreshInterval>
        <maximumWriteConnections>5</maximumWriteConnections>
        <maximumReadConnections>5</maximumReadConnections>
        <minimumReadConnections>1</minimumReadConnections>
        <minimumWriteConnections>1</minimumWriteConnections>
        <connectionPoolRefreshInterval>3600</connectionPoolRefreshInterval>
        <multibyte>false</multibyte>
        <multithreaded>true</multithreaded>
        <sharedPool>false</sharedPool>
        <xsiNilForNullData>true</xsiNilForNullData>
        <datasource>
            <type>Relational</type>
            <name>Cordys System#System</name>
        </datasource>
```

**applicationConnectorConfig.xml**

```
</component>
</configuration>
```

## deleteconnectionpoint - Ant task

The `deleteconnectionpoint` Ant task deletes the connection point from a service container.

### Sample usage

The following XML code shows the procedure to delete the Connection Point called `MyConnectionPoint` from the Service Container called `WS-AppServer` container that exists in the organization called `myorganization`.

```
<deleteconnectionpoint organization="myorganization"
name="MyConnectionPoint" servicecontainerName="WS-AppServer container"
servicegroupName="WS-AppServer Test" ldapRoot =
"cn=cordys,cn=fplcu7,o=myorg.com" />
```

### Parameters

Attribute	Description	Type	Required
<code>name</code>	Name of the connection point.	String	Yes
<code>organization</code>	Name of the organization where the service group is available.	String	Yes
<code>servicegroupName</code>	Name of the service group.	String	Yes
<code>servicecontainerName</code>	Name of the service container in which the connection point is configured.	String	Yes
<code>ldapRoot</code>	LDAP root of the AppWorks Platform instance.	String	Yes
<code>failOnError</code>	A flag that aborts the build when an exception occurs in the task. Default value is true.	Boolean	No

## deletedatabaseconfiguration - Ant task

The `deletedatabaseconfiguration` Ant task deletes a database configuration from the AppWorks Platform instance.

## Sample usage

The following task describes the procedure to delete a database configuration northwind present in the system organization.

```
<deletedatabaseconfiguration ldapRoot="cn=cordys,cn=bopcu7,o=myorg.com"
    name="northwind" organization="system" server="cin4003331"/>
```

## Parameters

Attribute	Description	Type	Required
server	Name of the AppWorks Platform instance.	String	Yes
ldapRoot	LDAP Root DN of AppWorks Platform instance.	String	Yes
name	Name of the database configuration that needs to be deleted.	String	Yes
organization	Name of the organization where the database configuration is present.	String	Yes
failOnError	A flag that aborts the build when an exception occurs in the task. Default value is true.	Boolean	No

## Deleteservicecontainer - Ant task

The `Deleteservicecontainer` Ant task deletes a service container from an organization in an AppWorks Platform instance.

## Sample usage

The following code describes the procedure to delete a service container called WS-AppServer container in a service group called WS-AppServerTest group which is available in an organization called myorganization.

```
<Deleteservicecontainer organization="myorganization" name="WS-
AppServercontainer" servicegroupName="WS-AppServerTest" ldapRoot =
"cn=cordys,cn=fplcu7,o=myorg.com" />
```

## Parameters

Attribute	Description	Type	Required
name	Name of the Service Container.	String	Yes
organization	Name of the organization.	String	Yes
servicegroupName	Name of the Service Group.	String	Yes
ldapRoot	LDAP root of the AppWorks Platform instance.	String	Yes
failOnError	A flag that aborts the build when an exception occurs in the task. Default value is true.	Boolean	No

## Deleteservicegroup - Ant task

The `Deleteservicegroup` Ant task deletes a service group from an organization in an AppWorks Platform instance. Deleting a service group also deletes the associated service containers and their connection points.

### Sample usage

The following example shows the procedure to delete a service group called `MyService Group` in the organization called `myorganization`.

```
<Deleteservicegroup organization="myorganization" name="MyService Group"
ldapRoot = "cn=cordys,cn=fplcu7,o=myorg.com" />
```

## Parameters

Attribute	Description	Type	Required
name	Name of the service group to be deleted.	String	Yes
organization	Name of the organization.	String	Yes
servicegroupName	Name of the service group.	String	Yes
ldapRoot	LDAP root of the AppWorks Platform instance.	String	Yes
failOnError	A flag that aborts the build when an exception occurs in the task. Default value is true.	Boolean	No

## deployapplicationpackage - Ant task

The `deployapplicationpackage` Ant task deploys an application package to the specified AppWorks Platform instance. The `.cap` file should be placed in the `<Cordys_HOME>\capcontent\packages` folder. This task does not copy the `.cap` file. If you need to transfer a file, run the ANT in-built copy or other equivalent task before running this task.

For cluster environments, the `deployapplicationpackage` Ant task must be invoked explicitly for all the nodes by pointing the `server` parameter to the individual nodes.

### Sample usage

#### Without input parameters

```
<deployapplicationpackage name="My Company Project" server="cin4005991"
    ldapRoot = "cn=cordys,cn=fp1cu7,o=myorg.com" />
```

#### With input parameters

```
<deployapplicationpackage ldapRoot="cn=cordys,cn=fp1cu7,o=myorg.com"
    name="My Company Project" organization="myorg"
    revertOnFailure="false" server="cin4005991" userInputFile="E:\\My
    Folder\\CAPUserInput"/>
```

#### In a cluster environment

If you are deploying application packages in a cluster environment, the application packages must be deployed on all nodes. Multiple `deployapplicationpackage` elements with the corresponding names of all nodes must provided as the value to the `server` parameter.

A sample with two nodes, `server1` and `server2`, is shown below:

```
<target name="testApplicationPackageDeploy">
    <deployapplicationpackage ldapRoot="cn=cordys,cn=fp1cu7,o=myorg.com"
        name="My Company Project" organization="myorg"
        revertOnFailure="false" server="server1" userInputFile="E:\\My
        Folder\\CAPUserInput"/>
    <deployapplicationpackage ldapRoot="cn=cordys,cn=fp1cu7,o=myorg.com"
        name="My Company Project" organization="myorg"
        revertOnFailure="false" server="server2" userInputFile="E:\\My
        Folder\\CAPUserInput"/>
</target>
```

## Parameters

<b>Attribute</b>	<b>Description</b>	<b>Type</b>	<b>Required</b>
name	Name of the application package to be deployed.	String	Yes
timeout	Timeout value that will be used while loading the application package. Value should be specified in milliseconds. Default value is 30000 (ms).	Integer	No
userInputFile	Location path of the XML file that contains the input to be passed to the application package while loading.	String	No
ldapRoot	LDAP root DN of AppWorks Platform instance.	String	Yes
server	Name of the server on which the application package has to be deployed. Refers to the server on which Monitor service container is running.	String	Yes
webserver	Name of the server from which CAP can be downloaded (provided for distributed installation). Default value is the server attribute, which indicates that both the Monitor service container and Web server are configured on the same system.	String	No
failOnError	True if the build needs to be aborted when an exception occurs. Default value is true.	Boolean	No
protocol	Protocol used for downloading CAP from the server. Supported values are http and https. Default protocol is http.	String	No
port	Port number of the Web server. Default port number is 80 for http and 443 is https protocol.	Integer	No
isUpgrade	Change the value to true if the application package needs to be upgraded. Default value is false.	Boolean	No
version	Version of the application package	String	No

<b>Attribute</b>	<b>Description</b>	<b>Type</b>	<b>Required</b>
	that needs to be deployed. By default, it will take the higher version of the application package. Specify this value only if a specific version of application package needs to be deployed.		
buildNumber	Build number of the application package that needs to be deployed. This value is mandatory if version is provided.	Double	No
revertOnFailure	A flag that reverts the application package to its previous state if an error occurs while deploying the application. Default value is true.	Boolean	No
organization	Name of the organization to which the package should be deployed.	String	No
acceptLicenseAgreements	True to accept the license agreement of the application package.	Boolean	No

**Note:**

- The name parameter should not include .cap file extension. The name should match 'Package Name' given in the CWS project package properties.
- The default port depends upon the protocol value and the protocol value depends upon the default port.
- The buildNumber parameter will be ignored if version is empty.
- You need to follow additional steps to deploy an application package in an organization space. Refer to Managing Application Packages In Organization Space for detailed steps.

The possible assumptions of the values are shown in the following table:

<b>If</b>	<b>Then</b>
port = empty and protocol = empty	port = 80, protocol = http
port = 80, protocol = empty	port = 80, protocol = http
port = 443, protocol = empty	port = 443, protocol = https
port = empty, protocol = http	port = 80, protocol = http
port = empty, protocol = https	port = 443, protocol = https

## Sample user input file

To complete the process of loading the application package, you may need to provide the user input for some packages. As Ant tasks do not run in an interactive mode, you should provide the input data through parameters. The parameter values should be specified in key-value pairs in the inputItem tag. There can be any number of inputItem tags that are required for loading the application package.

The input key and value depends upon the application package. See the application guide for details.

### Sample XML (Works OpenText Cordys 10.6.7 onwards)

```
<?xml version="1.0" encoding="UTF-8"?>
<UserInputs xmlns="http://schemas.cordys.com/cap/1.0">
<input artifactType="com.cordys.database.schema"
xmlns="http://schemas.cordys.com/cap/1.0">
    <inputItem key="databaseConfig.component@name">Connection Pool</inputItem>
    <inputItem key="databaseConfig.component.maximumWriteConnections">5</inputItem>
    <inputItem key="databaseConfig.component.maximumReadConnections">5</inputItem>
    <inputItem key="databaseConfig.component.minimumReadConnections">1</inputItem>
    <inputItem key="databaseConfig.component.minimumWriteConnections">1</inputItem>
</input>
<input artifactType="com.cordys.database.schema"
xmlns="http://schemas.cordys.com/cap/1.0">
    <inputItem key="databaseConfig.component@name">Connection Pool</inputItem>
    <inputItem key="databaseConfig.component.maximumWriteConnections">5</inputItem>
    <inputItem key="databaseConfig.component.maximumReadConnections">5</inputItem>
    <inputItem key="databaseConfig.component.minimumReadConnections">1</inputItem>
    <inputItem key="databaseConfig.component.minimumWriteConnections">1</inputItem>
</input>
</UserInputs>
```

### Sample XML(Deprecated in OpenText Cordys 10.6.7)

```
<?xml version="1.0" encoding="UTF-8"?>
<input artifactType="com.cordys.database.schema"
xmlns="http://schemas.cordys.com/cap/1.0">
    <inputItem key="databaseConfig.component@name">Connection Pool</inputItem>
    <inputItem key="databaseConfig.component.maximumWriteConnections">5</inputItem>
    <inputItem key="databaseConfig.component.maximumReadConnections">5</inputItem>
    <inputItem key="databaseConfig.component.minimumReadConnections">1</inputItem>
    <inputItem key="databaseConfig.component.minimumWriteConnections">1</inputItem>
</input>
```

## undeployapplicationpackage - Ant task

The undeployapplicationpackage Ant task removes the specified application package from an AppWorks Platform instance.

**Note:** The application package name should not include the version and build number details.

## Sample usage

```
<undeployapplicationpackage ldapRoot="cn=cordys,cn=bopcu7,o=myorg.com"
    name="My Package"
    organization="myorg"
    server="cin4003331"/>
```

## Parameters

Attribute	Description	Type	Required
server	Name of the system where AppWorks Platform is installed.	String	Yes
ldapRoot	LDAP root DN of AppWorks Platform instance.	String	Yes
name	Name of the application package that must be undeployed.	String	Yes
failOnError	A flag that aborts the build when an exception occurs in the task. Default value is true.	Boolean	No
organization	Name of the organization from which the package should be undeployed	String	No

# Chapter 35

## CoBOC browser

**Note:** CoBOC browser is deprecated in this version.

Use the CoBOC browser to create and maintain business objects. It enables you to:

- Create and edit business objects.
- View properties of a business object, and its XML representation.
- Filter business objects based on one or more templates.

See the following links for more information on using the CoBOC Browser:

- [Creating a business object](#)
- [Managing a business object](#)

## Creating a business object

This topic describes the procedure to create business objects, using the CoBOC browser.

**Note:** Object Template and CoBOC Browser are deprecated in Cordys BOP 4.1. You must now use WS-AppServer Custom class. See Creating a custom class in the *AppWorks Platform Advanced Development Guide*.

You can create and edit business objects using the CoBOC Browser. See [CoBOC browser](#).

### To create and edit a business object using the CoBOC Browser:

1. On the **Welcome** page, click  (CoBOC Browser).  
The choose a Service dialog box is displayed.
2. Select the required CoBOC service and click **OK**.  
Based on the CoBOC service you select, the CoBOC Browser page displays all the folders in the <user> space and <organization> space.
3. Right-click the required folder, select **New > Object**.  
The New Object page is displayed.
4. Enter the required information.  
Based on the template selected, the XML representation of the template is shown in the Object text box.

5. Enter the values for the elements of the object and click .  
The business objects are created.

## Managing a business object

**Note:** Cordys BOP 4.1 still supports the existing templates upgraded from earlier AppWorks Platform releases to ensure backward compliance. Use WS-AppServer Custom class instead. See Using WS-AppServer custom class as an alternative to object template in the *AppWorks Platform Advanced Development Guide*. However, you cannot create new object templates any more. Therefore, it is recommended to adopt the suggested alternative to object templates, that is, WS-AppServer Custom class as the future releases may not support object template and CoBOC.

**Note:** CoBOC Browser is deprecated in Cordys BOP 4.1.

### To manage a business object:

1. On the Welcome page, click  (Manage Business Objects).  
The Select a Service dialog box is displayed.
2. Select the required service and click **OK**.  
Based on the CoBOC service you select, the CoBOC Browser page displays all the folders in the <user> space and the <organization> space.
3. Do one of the following:
  - To view objects in a particular folder, click the required folder in the CoBOC Browser page.
  - To view the objects of selected templates, click the required folder, click  in the CoBOC Browser toolbar, select the required templates from the **Choose Filter(s)** dialog box, and click **OK**.  
The objects of the template are displayed in the CoBOC Folder pane.
4. Do any of the following to manage the objects:
  - To view the XML of an object, select the check box of the required object, and click .
  - To edit an object and its properties, select the check box of the required object, and click .
  - To delete one or more objects, select the check box of the objects you want to delete, and click .
  - To view the object and its properties, double-click the object.

The business objects are managed in this way using the CoBOC Browser.

# Chapter 36

## Signing packages

A package is signed to ensure that the authorship and integrity of the packages is loaded into the AppWorks Platform environment.

For each file in the application, an entry (XML node) is created containing the hash and name of the file with its relative path. A timestamp is added to this node. The whole XML is signed and a signature containing the certificate chain of the signing certificate is placed in it. The certificate chain has a sequence of certificates in which each certificate is issued by its subsequent Certificate Authorities (CA). The last certificate is a self-signed CA certificate (root certificate). The certificate chain is present in the .PFX file which is used to sign the packages.

The timestamp helps to identify the release date of the package. The certificate chain helps in identifying who signed it and in tracing who issued the certificates.

After a package is signed, it can be assured that the package is from the expected entity and the certificate can be used for signing packages.

Based on your operating system, you can sign the packages in one of the following ways:

- [Application signing on Windows](#)
- [Application signing on Linux](#)

### Application signing on Windows

You can sign one or more packages at a time and also provide appropriate password of a .pfx file to sign the packages.

#### Before you begin:

- OpenText recommends that the computer contains Java 7 to sign packages using the Package Signer application.
- Ensure that the computer has Java Cryptography Extension (JCE) Unlimited Strength Jurisdiction Policy files 1.6 to support better key lengths.
  - Download the JCE policy files (version 1.4.2) from the following location:  
<http://java.sun.com/javase/downloads/index.jsp>.

- Extract the jar files to the <java runtime environment>/lib/security folder.  
**Note:** Replace existing files, if any. If you have multiple Java Runtime Environment (JRE) folders in your system, make the changes in all of them.
- Add %JAVA\_HOME%\bin to the path.

You can sign packages using:

- A graphical user interface tool
- A batch file from the command prompt
- A Java command

## Sign a package using a graphical user interface tool

The graphical user interface tool is a standalone tool used to sign one or more packages at a time.

### To sign a package using a graphical user interface:

1. Click **Start > Programs > <AppWorks Platform version> > < Instance Name> > Tools > Package Signer**. Alternatively, you can navigate to the folder in which the <AppWorks Platform Installation> zip file is extracted, and double-click the `PackageSigner.bat` file.  
The Application Package Signer tool opens.
2. For:

Field	Action
Folder/application packages	Click  and select the package files. To select multiple packages, press <b>Ctrl</b> on the keyboard, and select the required package files.
Keystore Location	Click  and select the required signing certificate. This tool supports .PFX (.P12) format certificates only.
Keystore Password	Type the private key password of the certificate that is selected in the previous step. The private key password of a certificate is used to protect the private key of the signing certificate.
Alias	Optional. Type an alias name. <b>Note:</b> If the alias name is password protected, select <b>Requires Keypass</b> , and type the password in <b>Key Password</b> .
Requires Keypass	Optional. Applicable only if the alias name is password protected. Select <b>Requires Keypass</b> , and then type the password in <b>Key Password</b> .

Field	Action
Use Timestamp	<p>Optional. Select <b>Use Timestamp</b> and do one of the following:</p> <ul style="list-style-type: none"> <li>■ Select <b>URL</b> and type the URL of the Timestamp Authorization (TSA) in Timestamp Value.</li> <li>■ Select <b>Certificate</b> and type the alias of the TSA certificate in Timestamp Value.</li> </ul> <p><b>Note:</b> While signing a package, a timestamp is appended to the signature. As a part of deployment, the verification logic uses this timestamp value to check if the certificate is valid when the package is signed.</p>

3. Click **Sign**. If the certificate used for signing the package does not have the Code-Signing value in KeyUsage of the certificate, an alert is displayed but the package is signed with the certificate. The alert informs you that the purpose of the certificate is not digitally signing the packages.

The packages are signed and the existing packages are replaced with the signed packages.

## Sign a package by running a batch file from the command prompt

The command prompt option is used to build a framework for signing applications. After building the packages, the sign package task is called from the build framework.

### To sign a package by running a batch file:

1. Open the command prompt, and navigate to the folder containing the tool.
2. Type the following command in the command prompt and press Enter:

```
PackageSigner.bat -packagepath <location of the package folder or  
location of single package file or locations of multiple package files  
separated by ';'> -keystore <Signing certificate path> -storepass <key  
password> -alias <aliasname> -tsa <tsa url>
```

### For example:

- To sign all the packages in a folder:

```
PackageSigner.bat -packagepath "D:\SignApplication\packages" -keystore  
"D:\Signapplication\Signing Certificate.pfx" -storepass "secretpassword" -alias  
"aliasname"
```

- To sign multiple packages regardless of their location:

```
PackageSigner.bat -packagepath "D:\SignApplication\packages\audit.cap;  
D:\SignApplication\packages\documentation.cap" -keystore  
"D:\Signapplication\Signing Certificate.pfx" -storepass "secretpassword" -alias  
"aliasname"
```

- To sign packages with TSA:

```
PackageSigner.bat -packagepath "D:\SignApplication\applications\audit.cap;  
D:\SignApplication\applications\documentation.cap" -keystore  
"D:\Signapplication\Signing Certificate.pfx" -storepass "secretpassword" -alias  
"aliasname" -tsa "http://services.globaltrustfinder.com/adss/tsa"
```

**Note:** This tool supports .PFX (.P12) format certificates only.

If the certificate used for signing the package does not have the Code-Signing value in KeyUsage of the certificate, an alert is displayed but the package is signed with the certificate. The alert informs you that the purpose of the certificate is not digitally signing the packages.

The packages are signed and the existing packages are replaced with the signed packages.

## Sign a package using a Java command

The Java command is an alternative to run the batch file from the command prompt. However, you must set the classpath before signing using the Java command.

### To sign a package using a Java command:

1. Open the command prompt.

2. Set the classpath in one of the following ways:

- set classpath=<AppWorks\_Platform\_Package\_Signer\_Directory>\jars\packagesigner.jar;%classpath%;
- set classpath=<AppWorks\_Platform\_installdir>\components\packagesigner\packagesigner.jar;%classpath%

3. Type the following command at the command prompt and press **Enter**:

```
java com.cordys.tools.signpackages.ArchiveSigner -packagepath <location of the  
package folder or location of single package file or locations of multiple package  
files separated by ';'> -keystore <Signing certificate path> -storepass <key  
password> -alias <aliasname> -tsa <tsa url>
```

### For example:

- To sign all the packages in a folder:

```
java com.cordys.tools.signpackages.ArchiveSigner -packagepath  
"D:\SignApplication\packages" -keystore "D:\Signapplication\Signing  
Certificate.pfx" -storepass "secretpassword" -alias "aliasname"
```

- To sign multiple packages regardless of their location:

```
java com.cordys.tools.signpackages.ArchiveSigner -packagepath  
"D:\SignApplication\packages\audit.cap;D:\SignApplication\packages\documentation.ca  
p" -keystore "D:\Signapplication\Signing Certificate.pfx" -storepass  
"secretpassword" -alias "aliasname"
```

- To sign packages with TSA:

```
java com.cordys.tools.signpackages.ArchiveSigner -packagepath
"D:\SignApplication\packages\audit.cap;D:\SignApplication\packages\documentation.ca
p" -keystore "D:\Signapplication\Signed Certificate.pfx" -storepass
"secretpassword" -alias "aliasname" -tsa
"http://services.globaltrustfinder.com/adss/tsa""
```

**Note:** This tool supports .PFX (.P12) format certificates only.

If the certificate used for signing the package does not have the Code-Signing value in KeyUsage of the certificate, an alert is displayed but the package is signed with the certificate. The alert informs you that the purpose of the certificate is not digitally signing the packages.

The packages are signed and the existing packages are replaced with the signed packages.

## Application signing on Linux

You can sign one or more packages at a time and also provide an appropriate password of a .pfx file to sign the packages.

### Before you begin:

- OpenText recommends that the computer contains Java 7 to sign packages using the Package Signer application.
- Ensure that the computer has Java Cryptography Extension (JCE) Unlimited Strength Jurisdiction Policy files 1.6 to support better key lengths.
  - Download the JCE policy files (version 1.4.2) from the following location:  
<http://java.sun.com/javase/downloads/index.jsp>.
  - Extract the jar files to the <java runtime environment>/lib/security folder.  
**Note:** Replace existing files, if any. If you have multiple Java Runtime Environment (JRE) folders in your system, make the changes in all of them.
  - Add %JAVA\_HOME%\bin to the path.

You can sign packages using:

- A graphical user interface tool
- A batch file from the command prompt
- A Java command

## Sign a package through a graphical user interface tool

The graphical user interface tool is a standalone tool used to sign one or more packages at a time.

### To sign a package using a graphical user interface:

1. Navigate to the folder in which the <AppWorks Platform Installation> zip file is extracted, and double-click the PackageSigner.sh file.  
The Application Package Signer tool opens.

## 2. For:

Field	Action
Folder/application packages	Click  and select the package files. To select multiple packages, press <b>Ctrl</b> on the keyboard, and select the required package files.
Keystore Location	Click  and select the required signing certificate. This tool supports .PFX (.P12) format certificates only.
Keystore Password	Type the private key password of the certificate that is selected in the previous step. The private key password of a certificate is used to protect the private key of the signing certificate.
Alias	Optional. Type an alias name. <b>Note:</b> If the alias name is password protected, select <b>Requires Keypass</b> , and type password in Key <b>Password</b> .
Requires Keypass	Optional. This is applicable only if the alias name is password protected. Select <b>Requires Keypass</b> , and then type the password in Key Password.
Use Timestamp	Optional. Select <b>Use Timestamp</b> and do one of the following: <ul style="list-style-type: none"> <li>■ Select <b>URL</b> and type the URL of the Timestamp Authorization (TSA) in Timestamp Value.</li> <li>■ Select <b>Certificate</b> and type the alias of the TSA certificate in Timestamp Value.</li> </ul> <b>Note:</b> While signing a package, a timestamp is appended to the signature. As a part of deployment, the verification logic uses this timestamp value to check if the certificate is valid when the package is signed.

3. Click **Sign**. If the certificate used for signing the package does not have the Code-Signing value in KeyUsage of the certificate, an alert is displayed but the package is signed with the certificate. The alert informs you that the purpose of the certificate is not digitally signing the packages.

The packages are signed and the existing packages are replaced with the signed packages.

## Sign a package by running a batch file from the command prompt

The command prompt option is used to build a framework for signing applications. After building the packages, the sign package task is called from the build framework.

### To sign a package by running a batch file:

1. Open the command prompt, and navigate to the folder containing the tool.
2. Type the following command in the command prompt and press **Enter**:

```
PackageSigner.sh -packagepath <location of the package folder or location of single package file or locations of multiple package files separated by ';'> -keystore <Signing certificate path> -storepass <key password> -alias <aliasname> -tsa <tsa url>
```

### For example:

- To sign all the packages in a folder:

```
PackageSigner.bat -packagepath "/opt/SignApplication/packages" -keystore "/opt/SignApplication/Signing Certificate.pfx" -storepass "secretpassword" -alias "aliasname"
```

- To sign multiple packages regardless of their location:

```
PackageSigner.bat -packagepath "/opt/SignApplication/packages/audit.cap; /opt/SignApplication/packages/documentation.cap" -keystore "/opt/SignApplication/Signing Certificate.pfx" -storepass "secretpassword" -alias "aliasname"
```

- To sign packages with TSA:

```
PackageSigner.bat -packagepath "/opt/SignApplication/packages/audit.cap; /opt/SignApplication/packages/documentation.cap" -keystore "/opt/SignApplication/Signing Certificate.pfx" -storepass "secretpassword" -alias "aliasname" -tsa "http://services.globaltrustfinder.com/adss/tsa"
```

**Note:** This tool supports .PFX (.P12) format certificates only.

If the certificate used for signing the package does not have the Code-Signing value in KeyUsage of the certificate, an alert is displayed but the package is signed with the certificate. The alert informs you that the purpose of the certificate is not digitally signing the packages.

The packages are signed and the existing packages are replaced with the signed packages.

## Sign a package using a Java command

The Java command is an alternative to run the batch file from the command prompt. However, you must set the classpath before signing using the Java command.

### To sign a package using a Java command:

1. Open the command prompt.
2. Set the classpath in one of the following ways:
  - `export CLASSPATH=<AppWorks_Platform_Package_Signer_Directory>/jars/packagesigner.jar:$CLASSPATH`
  - `export CLASSPATH=<AppWorks_Platform_installdir>/components/packagesigner/packagesigner.jar:$CLASSPATH`

3. Type the following command at the command prompt and press **Enter**.

```
java com.cordys.tools.signpackages.ArchiveSigner -packagepath <location of the  
package folder or location of single package file or locations of multiple package  
files separated by ';'> -keystore <Signing certificate path> -storepass <key  
password> -alias <aliasname> -tsa <tsa url>
```

**For example:**

- To sign all the packages in a folder:

```
java com.cordys.tools.signpackages.ArchiveSigner -packagepath  
"/opt/SignApplication/packages" -keystore "/opt/Signapplication/Signing  
Certificate.pfx" -storepass "secretpassword" -alias "aliasname"
```

- To sign multiple packages regardless of their location:

```
java com.cordys.tools.signpackages.ArchiveSigner -packagepath  
"/opt/SignApplication/packages/audit.cap;  
/opt/SignApplication/packages/documentation.cap" -keystore  
"/opt/Signapplication/Signing Certificate.pfx" -storepass "secretpassword" -alias  
"aliasname"
```

- To sign packages with TSA:

```
java com.cordys.tools.signpackages.ArchiveSigner -packagepath  
"/opt/SignApplication/applications/audit.cap;  
/opt/SignApplication/packages/documentation.cap" -keystore  
"/opt/Signapplication/Signing Certificate.pfx" -storepass "secretpassword" -alias  
"aliasname" -tsa "http://services.globaltrustfinder.com/adss/tsa""
```

**Note:** This tool supports .PFX (.P12) format certificates only.

If the certificate used for signing the package does not have the Code-Signing value in KeyUsage of the certificate, an alert is displayed but the package is signed with the certificate. The alert informs you that the purpose of the certificate is not digitally signing the packages.

The packages are signed and the existing packages are replaced with the signed packages.

## Loading a signed package

A package can be loaded during the AppWorks Platform installation or after the AppWorks Platform installation. However, in each case, the procedure for loading a signed package is different.

### During AppWorks Platform installation

If you want to load a signed package while installing AppWorks Platform, all the signing certificates along with the certificates in their certificate chain that you trust must be placed in the 'certificates' folder. The 'certificates' folder is located in the folder in which the <AppWorks Platform installation> ZIP file is extracted.

During the AppWorks Platform installation, these certificates in the certificate folder are placed in the trust store. While loading the signed package, these certificates are considered as trusted. See the *AppWorks Platform Installation Guide* for more information.

## After AppWorks Platform installation

If you want to load a signed package, after installing AppWorks Platform, the signing certificates till the root certificate must be configured in the 'Certificate Manager'. For more information, navigate to **Administrator > Managing Security in AppWorks Platform > Digital Certificates and trust stores > Signing package** in the AppWorks Platform documentation.

If you as an AppWorks Platform Administrator want to change the default settings for verifying packages, the following options are available in the order of priority for the security settings. The list is in the decreasing order of priority.

- Unsigned: Packages that are not signed using a certificate.
- Tampered: Signed packages that are altered. This includes addition, removal, and update of the files in the application.
- Certificate not trusted: Packages signed using certificates that are not in the trust store. The signing certificate and its corresponding certificate chain are not configured in the trust store.
- Trusted Certificate without valid KeyUsage: Packages signed by a certificate that does not have Code-Signing extended KeyUsage. This certificate may be trusted, but the certificate is unintended for signing packages.

Each of the options can be assigned any of the following values. The values are listed in the order of their decreasing priority.

- Disallow: Package loading is stopped.
- Prompt: While loading the package, the choice is given to the user. The user receives a notification asking if the loading should be continued.
- Allow: Package loading is done without any prompts.

### Note:

- The default value for the package-verification settings is 'Disallow'. It is recommended to retain these settings.
- The value of 'Unsigned Packages' depends on the value of 'Tampered Packages'. Similarly, the value of 'Certificate not trusted' depends on the value of 'Trusted certificate without valid keyusage'.

# Chapter 37

## Formatting XML data

The XML Formatter parses and formats XML strings. An XML string is entered in the text area. The tool first parses the string and if any errors are encountered, reports them to the user. If there are no errors, the tool formats the XML string with proper indentation.

It does not support certain special characters, for example, &, <, >, @, and %.

**Note:** This is a formatting tool. It is not a debugging tool. It only indicates if there is an error in the XML string, and cannot identify the location or nature of the error. It also supports some Unicode characters. However, to use this feature, the Arial Unicode MS font is required for displaying multibyte characters. This font is available with MS Office 2000, FrontPage 2000, and Office XP. It is also available by default on Windows 2000 and Windows XP, but not on Windows NT.

### To format XML data:

1. Click **Start > Programs > AppWorks Platform > Instance Name > Tools > XML Formatter.**  
The XML Formatter window opens.
2. Enter the XML string to be formatted in the text area, and click **Format.**  
The XML data is formatted.

**Note:** Change the font of the text by clicking **Font** and selecting the required font. To close the tool, click **Exit**.

# Chapter 38

## HealthCheckURL

A solution based on AppWorks Platform can be made high available by creating a cluster of two or more nodes. Each node is a full installation of AppWorks Platform. This implies that the node has its own Web gateway and monitor. The Web gateway runs in the Web server, is the entry point to AppWorks Platform, and handles users and requests. AppWorks Platform can be deployed in a failover or load-balancing configuration. This is also known as an active/passive and active/active cluster. In a failover deployment, the users and requests must be sent only to the Web gateway on the active cluster node, and to all available Web gateways on all nodes in a load-balancing deployment.

To achieve this, you must use an IP Load Balancer that is placed in front of the Web servers in the cluster. All users and requests are sent to the IP Load Balancer, which sends them to the appropriate Web server and Web gateway. The IP Load Balancer needs to know the status of the Web gateway based on which it must send the users and requests. The IP Load balancer must redirect the requests to all the Web gateways that are up and running and skip those Web gateways that are a part of a node that is down.

However, in AppWorks Platform, it is not sufficient to know that the Web gateways are working. The Web gateway must be ready to process requests. Therefore, the AppWorks Platform Web gateway and Monitor must be available to process requests.

To provide the IP Load Balancer with the status of the Web gateway and monitor, a utility has been provided in AppWorks Platform called HealthCheckURL. All decent IP Load Balancers can send an HTML request to a Web server to see if that Web server can handle requests successfully. This HealthCheckURL utility can be invoked by using the below URL. The HealthCheckURL utility is always node-(computer name) specific:

```
http://<computer  
name>/cordysguest/com.eibus.web.tools.healthCheck.HealthCheckURL.wcp
```

This URL returns 200 OK / System is Up and Running if both the Web gateway and the monitor are available. Otherwise, it returns another HTML status code that can be different based on the error scenario. This utility sends a request to the monitor anonymously. If the monitor responds with a SOAP response or a SOAP fault, those responses are considered as positive signals of the health of the monitor.

The IP Load Balancer must redirect traffic only to the nodes where the HealthCheckURL response is HTML status code 200. In case of all other HTML status codes, no traffic must be sent. The HealthCheckURL must be invoked by the IP Load Balancer regularly to provide automatic recovery. The IP Load Balancer check must always be performed on HTML status codes and not on the text in the body such as **System is Up and Running**. For more information on AppWorks Platform high availability, see the *Appworks Platform High Availability Deployment Guide*.

# Chapter 39

## LDAP Explorer

The LDAP Explorer utility enables administrators to access LDAP objects and modify their attributes. Although the Management Console offers system administrators the capability to view and manage LDAP entries, the LDAP Explorer has the following advantages over the Management Console tool:

- It is easier to set ACLs on LDAP objects using the shortcut menu option that opens the ACL Editor.
- The LDAP Explorer enables system administrators to remotely manage the LDAP entries. In other words, an ECX installation on the machine is not mandatory for using the LDAP Explorer (as in the case of the Management Console).
- The LDAP Explorer is safer and more reliable to use since it enables the administrator to manage LDAP entries based on the current privileges.

This section contains the following topics:

- [Modifying single-value attributes of LDAP objects](#)
- [Modifying multi-value attributes of LDAP objects](#)
- [Modifying XML File Attributes of LDAP Objects](#)

### Modifying single-value attributes of LDAP objects

System administrators can modify the single-value attributes (the attributes that can take a single value) of an LDAP object in the [LDAP Explorer](#).

#### Before you begin:

- You must have the role of systemAdmin or orgAdmin to modify single-value attributes of LDAP objects.

#### To modify the single value attributes of an LDAP object:

1. On the Welcome page, click My Applications >  (LDAP Explorer).  
The LDAP Explorer window displays the list of LDAP objects (organizations, authenticated users and the Applications) with their content items.
2. Click the required object in the LDAP Explorer pane.

The properties of the LDAP object are displayed in the Properties pane.

3. To change the value, click the field and modify it.

**Note:** New values cannot be added for this attribute. Only the existing values can be modified.

## Modifying multi-value attributes of LDAP objects

You can modify the multi-value variables (the attributes that can take several values) of an LDAP object in the [LDAP Explorer](#).

### Before you begin:

- You must have the role of systemAdmin or orgAdmin to modify multi-value attributes of LDAP objects.

### To modify the multi-value variables of LDAP objects:

1. On the Welcome page, click  (LDAP Explorer).  
The LDAP Explorer pane displays the list of organizations, authenticated users, and the applications with their content items.
2. Go to the required object in the LDAP Explorer pane and select it.  
The properties of the LDAP object are displayed in the Properties pane.
3. Use any of the following options for modifying a multi-value variable:

Option	Description
To modify a value	Click the value field and change the value.
To add a new value	Click  and enter a value in the newly created field.
To delete a value	Select the field containing the value and click  .

**Note:** All modifications that are done using LDAP Explorer are automatically saved after every change is made.

## Modifying XML File Attributes of LDAP Objects

You can modify the XML file attribute of an LDAP object in the [LDAP Explorer](#).

### Before you begin:

- You must have the role of systemAdmin or orgAdmin to modify XML file attributes of LDAP objects.

**To modify the XML file attributes of LDAP objects:**

1. On the Welcome page, click My Applications >  (LDAP Explorer).  
The LDAP Explorer window displays the list of LDAP objects.  
It displays the list of organizations, authenticated users and the Applications with their content items.
2. Click the required object in the LDAP Explorer pane.  
The properties of the LDAP object are displayed in the Properties pane.
3. To change the value of an XML attribute, click  (Edit).  
The XML Editor dialog box opens.
4. Modify the data in the XML file and click  (Save).

# Chapter 40

## XMLStore Explorer

The XMLStore utility enables customization of XML objects. Different versions of the same object can exist in the XMLStore.

You can define standard XML objects that are available across all organizations and customize XML objects for a specific organization or a specific user.

The possible versions of XML objects are:-

- isv (): These are the original XML objects provided by the application without any customization. They can be read by all AppWorks Platform users. In other words, these are objects available for all users.
- organization (): These are application objects that have been customized for an organization or that have been created for an organization. An object with version as 'organization' can be read-only by the users in that organization.
- user (): These are application or organization objects that have been customized for a user or they may be objects that have been created for a user. These objects can be accessed or updated only by that user.

Applications must maintain a folder structure to define or create content in the XMLStore. The folder structure for an object is based on the application that is defining the content. The root folder must be a folder with the application name (for example, Cordys). This must be followed by a folder with the product name (for example, the AppWorks Platform folder contains a sub-folder called WCP). This folder will now contain sub-folders for all the different content types for the product (for example, the Cordys\WCP folder contains a sub-folder called Application Package Content). The sub-folder of a particular content type will contain all items of that type within it (for example, the Cordys\WCP\Application Package content folder contains all the AppWorks Platform Applications within it).

**Note:** File extensions are not recommended, since the type of the object is not defined by the extension but by its location. The application defining the content can also choose to have versioning by providing another folder for the version.

The application defining the content can specify the format of the object and the folder structure for storing the objects. For example, the application defining the content can specify that all objects of a specific type will be contained in the root folder or a folder structure created within the root folder. For the standard content types within AppWorks Platform, the folder structure specified by the application package is created within the

Cordys\WCP folder. However, if it is a new content type being defined by the application , a new folder structure must be implemented.

This section contains the following topics:

- [Adding Items to XMLStore](#)
- [Adding Folders to XMLStore](#)
- [Deleting items from the XMLStore](#)
- [Modifying items in the XMLStore](#)
- [Renaming items in the XMLStore](#)

## Adding Items to XMLStore

You can add an item to [XMLStore Explorer](#) in the XML Store Explorer.

### Before you begin:

- You must have the role of systemAdmin or orgAdmin to add items to XMLStore.

### To add items to XMLStore:

1. On the Welcome page, click  (XMLStore Explorer).  
The XMLStore Explorer window opens and displays the Collection folder containing all the files and directories in the XMLStore.
2. Expand the Collection folder.  
The folders containing the objects of the ISV or product are displayed in a tree structure.
3. Right-click a folder and select **New Item**.  
The User Prompt dialog box opens.
4. In **Enter Name of Item**, type a name for the item you want to create and click **OK**.  
A new item is created within the parent folder.

## Adding Folders to XMLStore

You can add a folder to [XMLStore Explorer](#) in the XMLStore Explorer.

### Before you begin:

- You must have the role of systemAdmin or orgAdmin to add folders to XMLStore.

### To add folders to the XMLStore:

1. On the Welcome page, click  (XMLStore Explorer).  
The XMLStore Explorer window opens and displays the Collection folder containing all the files and directories in the XMLStore.
2. Expand the Collection folder.  
The folders containing the objects of the ISV or product are displayed in a tree structure.

3. Right-click a folder and select **New Folder**.  
The User Prompt dialog box opens.
4. In **Enter Name of folder**, type a name for the folder you want to create and click **OK**.  
A new folder is created within the parent folder.

**Note:** To delete a folder, you must first delete the items beneath it. See [Deleting items from the XMLStore](#).

## Deleting items from the XMLStore

You can delete an item from [XMLStore Explorer](#) in the XMLStore Explorer.

### Before you begin:

- You must have the role of systemAdmin or orgAdmin to delete items from XMLStore.

**Note:** You cannot delete items that are of ISV version.

1. On the Welcome page, click  (XMLStore Explorer).  
The XMLStore Explorer window opens and displays the Collection folder containing all the files and directories in the XMLStore.
2. Expand the Collection folder.  
The various folders containing objects of the application or product, are displayed in a tree structure.
3. Expand a folder, right-click an item and select Delete from the menu displayed.

**Note:** To delete all the items of Organization version or User version, right-click the folder, select **Delete** and then select Organization Version or User Version as per your requirement. The associated items are deleted at a time.

## Modifying items in the XMLStore

You can modify items in [XMLStore Explorer](#) in the XMLStore Explorer.

### Before you begin:

- You must have the role of systemAdmin or orgAdmin to modify items in XMLStore.

### To modify items in the XMLStore:

1. On the Welcome page, click My Applications >  (XMLStore Explorer).  
The XMLStore Explorer window opens and displays the Collection folder containing all the files and directories in the XMLStore.
2. Expand the Collection folder.  
The folders containing the objects of the ISV or product are displayed in a tree structure.
3. Expand a folder, right-click an item and select **Edit** from the context menu.  
Alternatively, you can click the item to open the XML Editor.

4. Modify the data and click  (Save).  
The XML data is saved as Organization version.

## Renaming items in the XMLStore

You can rename items in the [XMLStore Explorer](#) in XMLStore Explorer. You cannot rename an item that belongs to an ISV version.

### Before you begin:

- You must have the role of systemAdmin or orgAdmin to rename items in the XMLStore.

### To rename items in the XMLStore:

1. On the Welcome page, click My Applications >  (XMLStore Explorer).  
The XMLStore Explorer window opens and displays the Collection folder containing all the files and directories in the XMLStore.
2. Expand the Collection folder.  
The various folders containing the objects of the ISV or product are displayed in a tree structure.
3. Expand a folder, right-click an item and select **Rename**.  
The User Prompt dialog box is displayed.
4. In Enter Name of Item, type a new name for the item and click **OK**.  
The item is renamed.



## Part V

# How Tos



# Chapter 41

## How Tos

The topics in this section aim to help you accomplish specific tasks in AppWorks Platform. The topics provide clear and concise instructions and include examples to assist you with the tasks. The How Tos in this section are categorized as follows.

- Administration (How Tos)
- Business Activity Monitoring (How Tos)
- Extended ECM (How Tos)

# Chapter 42

## Administration

Administration is the activity of managing and maintaining system related resources, applications, and objects. Administrative activities are performed by system or organizational administrators. The How To tasks for administration provide a quick reference for performing common administrative tasks. These tasks can belong to varied categories of installation, configuration, and maintenance.

- [Configuring AppWorks Platform in DMZ](#)
- [Configuring key store and trust store manually for LDAP service container](#)
- [Configuring single sign-on for failover](#)
- [Configuring virtual memory for XML NOM documents](#)
- [Changing the password for the key store and trust store of the LDAP client connection](#)
- [Changing the database configuration for service containers](#)
- [Creating a plug-in for a content management system](#)
- [Enabling event logging at the system and individual component level](#)
- [Enabling SSL communication](#)
- [Managing chain SOAP requests](#)
- [Removing header details from SOAP responses](#)
- [Access restriction to public SOAP operations](#)
- [Running the AppWorks Platform \(<instance name>\) in different user context in Linux](#)
- [Setting up start-up applications in the AppWorks Platform desktop](#)
- [Starting or stopping AppWorks Platform monitor from the command prompt](#)
- [Starting the service containers from the command prompt](#)
- [Using Windows authentication to access SQL server](#)
- [Removing stacktrace details from SOAP responses](#)
- [Changing the location of the upload and download folders](#)
- [Configuring document store connector with Apache CXF library](#)
- [Configuring access URLs](#)
- [Configuring Content Server with AppWorks Platform](#)
- [Configuring HTTP connector](#)

- Configuring the JMS connector
- Configuring the public URL path prefix
- Defining a custom renderer class to log messages
- Implementing projects combining Capture and AppWorks Platform
- Configuring the E-mail connector

## Configuring AppWorks Platform in DMZ

AppWorks Platform services open some anonymous sockets internally for transmission of SOAP messages. These sockets are opened on any freely available port. However, for a Demilitarized Zone (DMZ) machine, only specified ports are made available by the administrator. When AppWorks Platform is installed on a DMZ machine, there may be problems in transmission of SOAP messages, as the freely available ports may be restricted by the administrator.

**Note:** Deploying AppWorks Platform on a DMZ is not a recommended setup, as a lot of ports need to be opened in the firewall. You can use a reverse proxy to open only a couple of ports. This depends on the requirements and usage of SSL offloading. For example, only the ports that are being used by the AppWorks Platform webserver need to be opened.

## Configuring an active directory

Configure AppWorks Platform to authenticate against any active directory. This describe the steps for installing and configuring the Active Directory Application Mode (ADAM). ADAM is a Lightweight Directory Access Protocol (LDAP) directory service developed by Microsoft. Replicate these steps with the active directory you are using in your application.

1. To create a self-signed certificate:
  - a. Download and install IIS Resource Kit from  
<http://www.microsoft.com/downloads/details.aspx?FamilyID=56fc92ee-a71a-4c73-b628-ade629c89499&displaylang=en>
  - b. Click **Start > Programs > IIS Resources > SelfSSL > SelfSSL**.  
The SelfSSL command prompt opens.
  - c. Create a self-signed certificate for computer `my.computer.com`. Enter the following in the command prompt, with your computer's fully qualified domain name, port 636 (default ssl port of ADAM) with 365 days of validity:  

```
cd C:\Program Files\IIS Resources\SelfSSL> selfssl /T
/N:CN=my.computer.com /P:636 /V:365
```
  - d. Enter **y** when you are prompted for resetting the SSL settings.

2. To check the self-signed certificate with IIS:
  - a. In the command prompt, enter **iisreset** to reset IIS.
  - b. Close the command prompt window.
  - c. Enter `https://localhost:636` in the address bar of the browser. The Web server recognizes the certificate.
3. Download and install ADAM from  
<http://www.microsoft.com/downloads/details.aspx?FamilyId=9688F8B9-1034-4EF6-A3E5-2A2A57B5C8E4&displaylang=en>.
4. To create ADAM instance:
  - a. Click **Start > Programs > ADAM > Create ADAM Instance**. The Active Directory Application Mode Setup Wizard opens.
  - b. Click **Next**.
  - c. Select **A unique instance** and click **Next**.
  - d. Enter a name in **Instance Name** and click **Next**.
  - e. You can use the default values for the ports (636, 389), and click **Next**.
  - f. Select **Yes, create an application directory partition** and enter the Partition Name, for example, CN=MyOrganization,DC=MyCountry,DC=COM.
  - g. Click **Next**.
  - h. You can use the default values for file location and click **Next**.
  - i. Select **Network Service Account** and click **Next**.
  - j. Select Current logged on user: <user name> and click **Next**. The user installing ADAM will have Administrator privileges on the current instance of ADAM.
  - k. Select **Import the selected LDIF** for this instance of ADAM, all the LDIF files in the **Available files** area, and move them to the **Selected LDIF files** section using the **Add** option.
  - l. Click **Next**. The **Ready to Install** page opens.
  - m. Verify the information in the **Ready to Install** page and click **Next**. If the details are incorrect, click **Back** to go back and edit the information..
  - n. Click **Finish** to complete the installation.
5. To add a user to ADAM Instance:
  - a. Navigate to **Start > Programs > ADAM > ADAM ADSI Edit**.
  - b. Select **ADAM ADSI Edit** in the tree.
  - c. Click **Action > Connect to**. The Connection Settings dialog box opens. Enter a name in **Connection name**.
  - d. Select the **Distinguished name** option and enter an ADAM instance partition name. For example, CN=MyOrganization,DC=MyCountry,DC=COM.

- e. Click **OK**.
  - f. In the tree structure, navigate to the Partition Name entered.
  - g. Click **Action > New > Object**.... The Create Object wizard opens.
  - h. Select a user in **Select as class** and click **Next**.
  - i. Enter a name in **Value** and click Next and then click **Finish**.
  - j. Click **Action > Connect to**. The Connection Settings dialog box opens. Enter a name in **Connection name** and click **OK**.
  - k. Navigate to **CN=Configuration, CN=<serial number> > CN=Services > CN=Windows NT**, right click **CN=Directory Service**, and click **Properties**.
  - l. Select the attribute **dsHeuristics** and click **Edit** to change the value to **0000000001001** to enable password resetting without a secure connection.

**Note:** Ensure that you clear this value after resetting the password.
  - m. Select the created user under the **Partition Name**. Click **Action > Reset password**. Enter a password.
6. To configure ADAM for a generated certificate:
    - a. Change the security of the new key to readable for a user running the ADAM service (Network Service).
    - b. To identify the new key, sort the files in the following directory by date:  
C:\Documents and Settings\All Users\Application Data\Microsoft\Crypto\RSA\MachineKeys
    - c. Enter **mmc** in the command prompt. The Console window opens.
    - d. To add the Computer Account Snap in, click **File > Add/Remove Snap In**.
    - e. Click **Add** and select **Certificates**.
    - f. Click **Add > Computer Account** > follow the wizard.
    - g. To add ADAM service Snap in, click **File > Add/Remove Snap In Click Add**.
    - h. Select **Certificates**.
    - i. Click **Add** and select **Service Account**.
    - j. Follow the wizard and select **Local computer > ADAM Instance**.
    - k. Find certificates on the local machine.
    - l. Select certificate: my.computer.com
    - m. Copy Selected certificate to **Certificates - Service ('ADAM Instance Name')** on Local Computer - 'ADAM Instance Name'\Personal
    - n. To export certificate, click **Action > All Tasks > Export**.
    - o. Click **Next** and select **No, do not export the private key** option.
    - p. Select **Der encoded**, select **c:\tmp\root.cer**.
    - q. Click **Finish** and restart the ADAM service instance.

7. Enter the following in the command prompt to load the exported certificate in Java key store:

```
cd %JAVA_HOME%\jre\lib\security
keytool -import -alias test -file
c:\tmp\root.cer -keystore cacerts
```

- a. Change the default password of the Java key store.
- b. Select **Yes**.The default password is **changeit**.
- c. Enter the following in the command prompt to remove an installed certificate:

```
keytool -delete -alias test -keystore cacerts
```

## Configuring key store and trust store manually for LDAP service container

Secure Sockets Layer (SSL) is a new feature of AppWorks Platform, in which AppWorks Platform is integrated with Java security features. SSL over the AppWorks Platform Bus provides transport layer security on the AppWorks Platform Enterprise Service Bus. This security supports data integrity and privacy for messages.

SSL is enabled by default by the installer. However, if the installer is unable to create a key store and a trust store, which are required to enable SSL, you must create these files manually.

1. To create the key store file for AppWorks Platform with a Self-Signed Certificate:

- Use the following command to create the key store and the keys:

```
keytool -genkey -keysize 1024 -keyalg RSA -alias cordys -dname
'CN=<hostname>, OU=Cordys, O=vanenburg.com' -validity 1825 -
keystore KeyStore.jks -keypass <keypassword> -storepass
<keystorepassword> -storetype JKS
```

- To generate the self-signed certificate for the generated key store, use the following command :

```
keytool -selfcert -alias cordys -validity 1825 -keystore
KeyStore.jks -keypass <keypassword> -storepass
<keystorepassword>
```

- Copy the .jks file created to the <AppWorks Platform\_installdir>\certificates\keystore folder.

2. Replace the <hostname> with the fully qualified domain name of the server.
3. Choose a password for the private key and replace the <keypassword> with it in the specified commands.

4. Select a password for the key store and replace the <keystorepassword> with it in the specified commands .
5. Import certificates from OpenText CARS (multiple .cer files) to the AppWorks Platform Trust store.
  - a. Copy each <hostname>-cert.cer from each OpenText CARS installation into a temp folder on the AppWorks Platform server.
  - b. Issue the following command for each certificate to be imported from the temp folder into the AppWorks Platform trust store:  

```
keytool -import -alias <OpenText CARS1.2> -keystore TrustStore.jks -storepass <password> -import -file <hostname>-cert.cer -trustcacerts
```
  - c. Answer Yes to the prompt.
  - d. Copy the .jks file from the temp folder to the <AppWorks Platform\_installdir>\certificates\truststore folder.
6. Replace the <hostname> with the fully qualified domain name of the server.
7. Select a password for the trust store and replace <truststorepassword> with this command.
8. Configure the Key Store and Trust Store for AppWorks Platform:

- a. Generate the Base 64 encoded password:

Windows	<ul style="list-style-type: none"> <li>• Open the command line.</li> <li>• Enter cd &lt;OpenText CARS_installdir&gt;\bin EncodePassword &lt;password&gt;</li> </ul>
Linux	<ul style="list-style-type: none"> <li>• Open the command line.</li> <li>• Enter cd &lt;OpenText CARS_installdir&gt;\sbin .EncodePassword.sh &lt;password&gt;</li> </ul>

Enter this encoded password in bus.ssl.keystorepassword/  
bus.ssl.truststorepassword in wcp.properties.

- b. Configure the key store and trust store in AppWorks Platform by modifying the following attributes in the wcp.properties file:
  - bus.ldap.processor.ssl = true
  - bus.ssl.keystore =<AppWorks Platform\_installdir>\certificates\keystore\keystore.jks
  - bus.ssl.keystorepassword=<base64 encoded password>
  - bus.ssl.truststore=<AppWorks Platform\_

```
installdir>\certificates\truststore\truststore.jks
• bus.ssl.truststorepassword=<base64 encoded password>
```

## Configuring single sign-on for failover

The AppWorks Platform Single Sign-on (SSO) feature must always be available. As an administrator, you must provide a mechanism to handle any failures. Failover is a mechanism to provide a standby for the existing service. In case of failure of one service, the other takes over. This switch is automated and ensures continuous availability of the service.

The failover mechanism for SSO consists of creating two or more SSO service containers within the existing SSO service group. While creating the service containers, you must specify the custom algorithm routing option.

After creating the service containers, specify a preference for each service container. In case of failure of the main service container, the remaining service containers take over in the order of preference. In this example, one service container is being created.

### To configure single sign-on for failover:

1. Add a new SSO service container to the SSO service group and name it Single Sign-On2.
2. Ensure that the Startup Type for this service container is **Automatic**. For information on creating a service container, see [Creating a service container](#).

**Note:** While creating the service container, enter the preference for the new service container in **Preference**. A new SSO service container is created and the System Resource Manager window opens.

3. Double-click the SSO service group.  
A list of SSO service containers and the **ServiceGroup Properties - Single Sign-On** is displayed.
4. From the Routing Algorithm list, select **Custom Algorithm**.
5. Click  (Save).
6. Repeat these steps for each SSO service container.

SSO is now configured for a failover.

## Configuring virtual memory for XML NOM documents

When the Java Virtual Machine (JVM) loads XML libraries such as `eibxml.dll` or `libleibxml.so`, it requires some memory to create the XML nodes. The memory allocated for this purpose is shared by all XML documents of the `com.eibus.xml.nom.Document` class. The virtual memory is specified in megabytes (MB). The default memory allocated is 256 MB.

You can allot a higher memory for creating the XML nodes by specifying it in the `wcp.properties` file. This is useful when loading multiple XML nodes simultaneously on a computer that has a large Random Access Memory (RAM). The virtual memory can also be added as -D in the JRE configuration in the Service Container Configuration Interface to configure the virtual memory for a specific service container. See Service Container Configuration Interface in the *AppWorks Platform Advanced Development Guide*.

**Note:** Guidelines to configure the virtual memory for XML NOM documents:

- The virtual memory (in MB) that you specify must be lower than the RAM used by the computer.
- If you specify a negative or invalid value, the default memory (256 MB) is allotted.
- If you specify a value that is an odd number, it is rounded off to the next even number.
- This memory is only allotted to the service container or application that loads the XML library.

### To configure the virtual memory:

1. In a Windows computer, click Start > Programs > AppWorks Platform > Instance Name > Tools > Management Console.  
Or  
In a Linux computer, click Applications > AppWorks Platform > Instance Name > Tools > Management Console.  
The Management Console window opens.
2. On the Management Console window, click **Platform Properties**.  
The Platform Properties dialog box opens.
3. In the list of properties, identify the `bus.xml.vm.maxsize` property and assign the required memory value to it.

**Note:**

If the property is not available, create it and assign a value in MB to it.

4. Click  (Save).  
The new property is added to Platform Properties.
5. Restart AppWorks Platform.  
The virtual memory for XML NOM documents is configured.

**Note:** Alternatively, you can configure virtual memory to use the `com.eibus.util.system.EIBProperties` class available in the SDK. It contains an API that creates new properties and sets values for them.

## Changing the password for the key store and trust store of the LDAP client connection

The AppWorks Platform installer assigns a default password, **secret**, for the private key, key store, and trust store. You can change this password.

### 1. To change the key store password:

- a. Enter the following in the command prompt:

```
keytool -keypasswd -alias bcp -keystore <full path of the key  
store file>
```

**Note:** The full path of the key store file must include the file name. For example, D:\Program Files\Cordys\certificates\keystore\cordys-srv-del-6c-ks.jks.

- b. Enter the current and new password for the key.
- c. Enter the following in the command prompt:

```
keytool -storepasswd -keystore <full path of the key store  
file>
```

**Note:** The full path of the key store file must include the file name. For example, D:\ProgramFiles\Cordys\certificates\keystore\cordys-srv-del-6c-ks.jk.

- d. Enter the current and new password for the key store.

**Note:** The new key password and the key store password must be the same.

### 2. To change the trust store password:

- a. Enter the following in the command prompt:

```
keytool -storepasswd -keystore <full path of the trust store file>
```

**Note:** The full path of the trust store file must include the file name. For example, D:\ProgramFiles\Cordys\certificates\truststore\cordys-srv-del-6c-ks.jks.

- b. Enter the current and new password for the trust store.

### 3. To modify the password in wcp.properties:

- a. Generate the Base 64-encoded password. Depending on the operating system, do one of the following:

Operating system	Procedure
Windows	Enter the following in the command line: cd <OpenText CARS_installdir>\bin EncodePassword <password>
Linux	Enter the following in the command line: cd <OpenText CARS_installdir>\sbin .EncodePassword.sh <password>

- b. Note the encoded password and enter it in `bus.ssl.keystorepassword` / `bus.ssl.truststorepassword` in the `wcp.properties` file.
- c. To change the key store and trust store passwords in AppWorks Platform, modify the following attributes in the `wcp.properties` file:
  - `bus.ssl.keystorepassword=<base64 encoded password>`
  - `bus.ssl.truststorepassword=<base64 encoded password>`

## Creating a plug-in for a content management system

AppWorks Platform can work with any Content Management System (CMS). It can work with CMS that are JSR 170-compliant and ones that are not. Jackrabbit is a JSR 170-compliant CMS. AppWorks Platform provides a plug-in that acts as a client of Jackrabbit for storing documents. While installing AppWorks Platform, the AppWorks Platform repository is configured as the default CMS.

### Before you begin:

- Use Jackrabbit instead of the AppWorks Platform repository in your application, since there may be an existing CMS. You can create a plug-in that can be used with AppWorks Platform to connect to CMS.
- To create a plug-in for a CMS that does not comply with JSR 170 standards, implement the `IDocumentStore` interface.
- To create a plug-in for a JSR 170-compliant CMS:

Extend the `JSRDocumentStoreClient` class. For example, in Jackrabbit implementation, the `JSRDocumentStoreClient` class was extended to get the `JackRabbitClient` class.

The following table lists methods you must override, with the expected output:

Method name	Expectation
<code>getSession</code>	Create a session using this method. <code>Repositoryobject</code> is passed as the

Method name	Expectation
(Repository repository)	parameter, and it retrieves the <code>theSessionobject</code> property. For example, <code>getSession() : session = this.repository.login(new SimpleCredentials(this.userName, this.password.toCharArray()));</code>
getRepository (String url)	Create the repository object based on the CMS specification. The URL of the repository is passed as the parameter for this method and it retrieves a JSR 170 -compliant repository object. For example, <code>repository = (Repository) new URLRemoteRepository(this.repositoryURL);</code>

The plug-in for a JSR 170-compliant CMS is created.

- To create a plug-in for a CMS that does not comply with JSR 170 standards:

Implement the `IDocumentStore` interface. This interface contains the following methods:

Method name	Expectation
<code>CreateDocument</code>	Use this method to create documents.
<code>DeleteDocument</code>	Use this method to delete documents.
<code>GetDocument</code>	Use this method to retrieve the details of a document.
<code>GetDocumentsInfo</code>	Use this method to retrieve the details of a document. If the name of a folder is passed, the method retrieves the details of the collection of files in the folder. If the name of a file is passed, the method retrieves the details of the file.
<code>initializeStore</code>	Implement the <code>initializeStore</code> method, which is part of the <code>IDocumentStore</code> interface. The <code>StoreConfigurationobject</code> is passed as the parameter for this method and initializes the repository settings. The <code>StoreConfigurationobject</code> contains the method <code>getExtConfigXML</code> . Use this to retrieve the XML content provided in the External Configuration XML field, while creating the service container for this repository. For more information on service containers, see <a href="#">Managing service containers</a> .
<code>UpdateDocument</code>	Use this method to update an existing document.

The following plug-ins are created:

- Plug-in for a CMS that does not comply with JSR 170.
- A new plug-in to work with the repository.

# Enabling event logging at the system and individual component level

You must enable event logging in AppWorks Platform at the system level, and the individual component level. This is enabled by default at the system level. You can also enable it at an individual component level by configuring the service container to enable event logging.

## Enabling event logging

Level	Default setting	Steps to enable
System	Enabled by default at the system level.	To customize logging, modify the <code>Log4jConfiguration.xml</code> file located at <code>&lt;AppWorks Platform_installdir&gt;\config</code> folder
Individual component		<p><b>To enable the logging of events at an individual component or service container level:</b></p> <ol style="list-style-type: none"> <li>On the <b>Welcome</b> page, click  (System Resource Manager). The System Resource Manager window opens and lists the available service containers in the Service Containers App Palette.</li> <li>Go to the required <b>Service Group &gt; Service Container</b>.</li> </ol> <p><b>Note:</b> Ensure that event logging is enabled.</p> <ol style="list-style-type: none"> <li>Double click the service container. The corresponding properties are displayed in the <b>Service Properties - &lt;Service Name&gt; App Palette</b>.</li> </ol> <p><b>Note:</b> The default tab is General.</p> <ol style="list-style-type: none"> <li>Make the required changes and click  (Save). For more information on properties, see <a href="#">Modifying a service container</a>.</li> </ol>

## Enabling SSL communication

Secure Sockets Layer (SSL) is an open, nonproprietary protocol for securing data communication across computer networks. Following are some features of SSL:

- Enables communication between the application protocol, such as HTTP, and the connection protocol, such as TCP/IP.
- Provides server authentication, message integrity, data encryption, and optional client authentication for TCP/IP connections.
- Provides transport layer security on the AppWorks Platform Enterprise Service Bus, if it is over the AppWorks Platform Bus. This security supports data integrity and privacy for messages.

**Note:**

- SSL is superseded by the Transport Level Security (TLS).
- TLS over Enterprise Service Bus (ESB) is enabled by default. To disable it, add the property `sslsocket.public.keystore=to wcp.properties`.

## Managing chain SOAP requests

In AppWorks Platform, all requests are routed to service containers. There may be instances when one service container may request another service container for information. In turn, the second service container may query yet another service container. This forms a chain SOAP request, spanning several service containers. Such messages must be handled differently from the normal SOAP messages.

**Before you begin:**

- All service containers, by default, run under system in the case of Windows, or root login in the case of Linux. Therefore, when the SOAP Message is created, it automatically takes the default user context as system or root. To avoid this, the user must ensure that the correct user context is taken while sending the SOAP Message. The `soapTransaction.getUserCredentials().getName()` Web service operation retrieves the name of the user or component that has initiated the SOAP request and can be used for the purpose. Setting the correct user context is important because the system user privileges may be different from the privileges of the user who initiated the request, and this may affect the way the SOAP request is handled.
- Every service container has its own connector object, which can be used for sending and receiving SOAP messages. The developer must carefully select the mode of sending the SOAP messages (either synchronous or asynchronous) since that would affect the way the request is handled. For example, a synchronous `sendAndWait` request would block the SOAP requests that are queued and not resend it to the intended service container.

**To manage chain SOAP requests spanning several service containers with user context:**

1. Create the SOAP message.
  - Using the application connector code, obtain the connector object of the processor through `processor.getConnector()`.
  - Using the APIs provided in `com.eibus.connector.nom.Connector`, create the required SOAP message.
2. Maintain the user context. While creating the SOAP message, maintain the user context by obtaining the user name (of the initiator of the request) from the transaction object `soapTransaction.getUserCredentials().getName()`.
3. Select the mode of sending the SOAP message. Select the correct mode of sending the SOAP message to the required service container. This could be synchronous (`send`, `sendAndWait`) or asynchronous (`sendAndCallback`).

## Access restriction to public SOAP operations

AppWorks Platform Web Gateway manages sending and receiving SOAP requests and responses from the service groups. At first, the gateway determines the user's authenticity in the AppWorks Platform environment. Later, based on the Web service interface details and the namespace contained in the request, the gateway sends the SOAP request to the service container without checking for permissions that the user needs to access the Web service interface or namespace. This process involves the risk of exposing all the public Web service operations on LDAP.

As a security measure, AppWorks Platform now provides a restricted environment for sending requests to service groups, based on the sandbox enabled model approach. Before sending the request to the service container, the gateway checks if the Web service interface and namespace have required permissions for the user. When the request is sent, the gateway prepares a permission object containing user reference, Web service interface and namespace. This permission object is acted upon by the Access Controller.

The access controller verifies the permission object against the Security Policy. The security policy comprises of:

- Black list - Containing users and Web service operations that are denied access
- White list - Containing users and Web service operations that are granted access.

The request is sent to the service group only when the Web service operations have necessary permissions.

## Running the AppWorks Platform (<instance name>) in different user context in Linux

When AppWorks Platform is installed on a Linux machine, the OpenText AppWorks Platform (<instance name>) runs in the AppWorks Platform user context by default. This might not be required at all times. The user can run the OpenText AppWorks Platform (<instance name>) in a different user context.

The user context for the AppWorks Platform (<instance name>) in Linux can be changed by using the `/etc/init.d/wcpd` command. This program changes the user context to the specified user.

Run the `/etc/init.d/wcpd` command with the user being specified.

### To change the user context for the AppWorks Platform (<instance name>) in Linux:

Run this command:

```
/etc/init.d/wcpdinstancename {start [-u <user>] | stop | restart [-u <user>]}
```

The arguments are:

- stop: to stop the daemon.
- start: to start the daemon.
- restart: to stop and start the daemon.
- The -u option available with the start and restart arguments can be used to specify the user ID under whose context the AppWorks Platform daemon or Monitor must be running. This user is the owner of the AppWorks Platform (<instance name>) process and its child processes.

By default, the monitor runs in the root user's context unless this option is specified.

**Important:** Restart the AppWorks Platform (<instance name>) every time the user context is changed, for the changes to be effective.

## Setting up start-up applications in the AppWorks Platform desktop

You can specify any organization as a start-up organization. You can also specify the organization and user versions of the application, if available in your default organization, as the start-up application.

The applications to be started by default are defined as application definitions, and are specified as an XML store menu item at `/Cordys/WCP/Desktop/automatic` in the AppWorks Platform installation. All such applications are specified with `automatic="true"`.

Sample automatic menu:

```

<menu>
    <Application automatic="true" taskbar="false" title="false">
        <description>Taskbar</description>
        <caption>Taskbar</caption>
        <url>/Cordys/WCP/taskbar.htm</url>
        <id>Taskbar</id>
        <frame>footerToolbar</frame>
    </Application>
    <Application automatic="true" display="current" taskbar="false">
        <description>Style Monitor</description>
        <caption>Style Monitor</caption>
        <id>Style Monitor</id>
        <url>/Cordys/WCP/style/styledaemon.htm</url>
        <frame>sub</frame>
        <data/>
    </Application>
</menu>

```

### To set up and customize the start-up applications:

1. The automatic menu in the XML store must be modified accordingly.
2. Add the required application as an application definition in the menu file. In this case, a mybar taskbar is added to the automatic menu file:

```

<Application automatic="true" display="visible" taskbar="false">
    <id>Custom Taskbar</id>
    <icon>/Cordys/WCP/theme/default/images/bar.gif</icon>
    <url>/Cordys/WCP/mybar.htm</url>
    <description>Taskbar</description>
    <caption>Taskbar</caption>
    <frame>left</frame>
</Application>

```

The taskbar mybar is added .

The menu refers to the navigation tree on the left. It is also an application that is defined as another application definition within the automatic menu. You can also modify or remove the menu from the start up by modifying the automatic menu.

## Loading an application with system environment variables content

On a Windows-based machine, to load an application with system environment variables, Dynamically Linked Library (DLL) registration, or virtual folder creation content, the 'Cordys <Instance Name>' user must have Windows administrator rights on that machine.

The '**Cordys <Instance Name>**' user has a minimal set of privileges. To run the OpenText AppWorks Platform (<Instance Name>) service, the user needs Windows administrator rights.

#### To set administrative privileges and load the application:

1. Elevate the access rights of the user to **Windows Administrator**.  
For more information, see [Changing access rights of users](#).
2. Restart **AppWorks Platform (<Instance Name>)**, to run it in the context of the new role of the user as a Windows administrator. For more information, see [Restarting AppWorks Platform monitor](#).
3. Load the application.
4. Revoke the Windows administrator privileges from the user.
5. To run the monitor in the context of the previous role as a user, restart it.  
An application with the system environment variables is loaded.

## Restarting AppWorks Platform monitor

You must restart the AppWorks Platform Monitor for configuration changes to take effect. The procedure to restart it varies for Windows and Linux based computers.

#### In Windows:

1. Click **Start > Run**, type `services.msc` and press ENTER.  
The Services window opens and displays all the services that are running on your computer.
2. In the list, right-click OpenText AppWorks Platform (<Instance Name>), and then select **Restart**.

#### In Linux:

1. Launch the terminal.
2. Type `service wcpd<instancename> restart` and press ENTER.

The OpenText AppWorks Platform (<Instance Name>) is restarted.

## Starting or stopping AppWorks Platform monitor from the command prompt

In some cases, you might need to start or stop the OpenText AppWorks Platform (<Instance Name>) using the command prompt.

- Set the following environment variables, depending on the operating system.

<b>OS type</b>	<b>Environment variables</b>
Linux	<ul style="list-style-type: none"> <li>■ <b>Export</b> AppWorks_Platform_installdir=&lt;AppWorks_Platform_installdir&gt;</li> <li>■ <b>Export</b> CLASSPATH=\$AppWorks_Platform_installdir/cordyscp.jar:\$CLASSPATH</li> <li>■ <b>Export</b> LD_LIBRARY_PATH=\$AppWorks_Platform_installdir/lib:\$LD_LIBRARY_PATH</li> </ul>
Windows	<ul style="list-style-type: none"> <li>■ <b>SET</b> AppWorks_Platform_installdir=&lt;AppWorks_Platform_installdir&gt;</li> <li>■ <b>SET</b> CLASSPATH=%AppWorks_Platform_installdir%\cordyscp.jar;%CLASSPATH%</li> <li>■ <b>SET</b> PATH=%AppWorks_Platform_installdir%\lib;%PATH%</li> </ul>

- Launch the OpenText AppWorks Platform (<Instance Name>) using this command:  
`java com.eibus.applicationconnector.monitor.Launcher`

**Note:** To stop the monitor, press **ENTER** in the command prompt window.

## Starting the service containers from the command prompt

In some cases, you might need start or stop the service containers using the command prompt. For example, the Explorer is not loaded; therefore, the service containers cannot be started or stopped using the AppWorks Platform administrative tools.

### Before you begin:

- You must add the <AppWorks\_Platform\_installdir>/cordyscp.jar that contains the mandatory JAR files.

- Set the following environment variables depending on the operating system.

OS type	Environment variables
Windows	<ul style="list-style-type: none"> <li>■ <b>Set</b> AppWorks_Platform_installdir=&lt;AppWorks_Platform_installdir&gt;</li> <li>■ <b>Set</b> CLASSPATH=%AppWorks_Platform_installdir%\cordyscp.jar;%CLASSPATH%</li> <li>■ <b>Set</b> PATH=%AppWorks_Platform_installdir%\lib;%PATH%</li> </ul>
Windows	<ul style="list-style-type: none"> <li>■ <b>Export</b> AppWorks_Platform_installdir=&lt;AppWorks_Platform_installdir&gt;</li> <li>■ <b>Export</b> CLASSPATH=\$AppWorks_Platform_installdir/cordyscp.jar:\$CLASSPATH</li> <li>■ <b>Export</b> LD_LIBRARY_PATH=\$AppWorks_Platform_installdir/lib:\$LD_LIBRARY_PATH</li> </ul>

- Invoke the service container using the following command, with the name of the required service container as the parameter.  

```
Java class com.eibus.soap.Processor
```
- Launch **java -cp <the mandatory cordyscp.jar and the JARs specific to the corresponding service container> com.eibus.soap.Processor with the DN attribute of service container.** For example:

```
java -cp <full Classpath> com.eibus.soap.Processor -dn 'cn=AuditServiceProcessor,cn=AuditService Service,cn=soap nodes,o=Acme,cn=cordys,o=vanenburg.com'
```

**Note:** Ensure that the java executable is in the PATH and the class files required are in the CLASSPATH. If not, include them and restart the computer for the changes to be effective. This enables the service container to run in the command prompt.

The service container is started from the command prompt.

**Note:** To stop a service container, press **CTRL+C**. This stops the service container and returns the control to the command prompt.

## Using Windows authentication to access SQL server

AppWorks Platform, by default, runs under the TomEE user context.

**To access SQL Server database using Windows authentication:**

1. Connect to SQL Server using the SQL Server Enterprise Manager.
2. Right-click **Security > Logins**, and then select **New Login**.  
The Login dialog box opens.
3. In **User Name**, type <computername>\<TomEE user name> .  
**Note:** If AppWorks Platform and SQL Server are installed on different computers, you must create a user on the computer where SQL Server is installed. The user name and password must be the same as that of the TomEE user in AppWorks Platform.
4. Verify whether the Windows Authentication option is selected and click **OK**.  
The user is created and displayed under the Logins folder.
5. Right-click <computername>\<TomEE user name> and select **Properties**.  
The Login Properties dialog box opens.
6. Click **Server Roles** and select the roles required for your application. For example, for the user to create a database, select the **dbcreator** role. See SQL Server documentation for other role details.
7. Click **User Mapping**, select the required databases from the User mapped to this login section, and then click **OK**.
8. Open the AppWorks Platform Explorer.
9. From the Welcome page, click  (System Resource Manager).  
The System Resource Manager window opens.
10. Click **Manage Database Configurations**.  
The Manage Database Configurations dialog box opens.
11. Click .  
A new row is appended to the table and a section to provide database configuration details is displayed in the bottom pane.
12. In **Name**, type the name of the database configuration.
13. Optional. In **Description**, describe the database configuration.
14. From the **JDBC Driver** list, select **MS SQL Server**.  
The fields are automatically populated with the appropriate values.
15. Append the value in the Connection String with the following:  
`;integratedSecurity=true`
16. In **Default Database**, type a database name.
17. Leave the username and the password fields blank.
18. To ensure AppWorks Platform can connect to the database with the provided parameters, click  (Test Connectivity).
19. Click .  
The SQL server can now be accessed from AppWorks Platform using Windows authentication.

## Removing stacktrace details from SOAP responses

For security reasons, you may not want the Web service response to contain stack traces or other technical detailed information. This topic describes how sensitive or unwanted content in error messages can be removed from the SOAP responses.

You can remove sensitive data from the SOAP response using:

- Strip stacktrace feature
- MessageCode blacklist

### gateway.fault.stripstacktrace

When a request results in an error situation, the response contains a Java stacktrace. To remove such a Java stacktrace, the property `gateway.fault.stripstacktrace` can be set, which causes the stacktrace to be removed from the SOAP response.

With this property enabled, AppWorks Platform completely removes the `cordys:FaultRelatedException` tag from the SOAP fault in the SOAP response. Its default value is **false**.

**Note:** This property is available from Cordys BOP 4.2.18 and later versions.

An example usage of the property in the `wcp.properties` is as follows:

```
gateway.fault.stripstacktrace=true
```

### gateway.fault.blacklist

The property `gateway.fault.blacklist` can be used to define a blacklist of SOAP faults, which must not be sent to the client. When a SOAP response contains a message code that is present in the blacklist then the message is replaced with a generic response.

Sample usage of the property in the `wcp.properties`:

```
gateway.fault.blacklist=[messageCodeA messageCodeB|regular expression]
```

The message codes are separated by white spaces. By default, no message codes are blocked.

It is also possible to specify a regular expression to match a set of message codes. The following is an example to match all the ACLLib message codes:

```
gateway.fault.blacklist=Cordys.ACLLib.Messages.*
```

## Example

To determine the message code, consider the following message bundle example:

```
<MessageBundle xmlns="" id="Cordys.ACLLib.Messages">
    <Message id="couldNotUpdateLDAP">
        <MessageText>Unable to update LDAP</MessageText>
    </Message>
    <Message id="ldapEntryIsNotPresent">
        <MessageText>LDAP entry is not present. The data is changed by another user.</MessageText>
    </Message>
    <Message id="objectWasAlreadyCleanedUp">
        <MessageText>Cannot operate on the object as it was already cleaned up.</MessageText>
    </Message>
</MessageBundle>
```

The message code is a combination of the MessageBundle ID and the Message ID. For example, Cordys.ACLLib.Messages.couldNotUpdateLDAP.

## Execution

When a SOAP fault occurs, its message code is matched against the blacklist. If it matches, it is replaced by a generic message and the original fault is logged in the Gateway log.

### Generic SOAP fault

Generic SOAP fault which replaces the blacklisted messages:

```
<SOAP:Fault>
    <faultcode>SOAP:Server</faultcode>
    <faultstring xml:lang="en-US" >An internal server error occurred. Contact your administrator for more information.</faultstring>
    <detail>
        <cordys:FaultDetails xmlns:cordys="http://schemas.cordys.com/General/1.0/">
            <cordys:LocalizableMessage>

<cordys:MessageCode>Cordys.WebGateway.Messages.serverError</cordys:MessageCode>
            </cordys:LocalizableMessage>
        </cordys:FaultDetails>
    </detail>
</SOAP:Fault>
```

# Changing the location of the upload and download folders

To enable uploading a file to a service, the installer creates a folder in the <AppWorks Platform\_installdir> (<AppWorks Platform\_installdir>/content/uploadcontent) with the appropriate permissions during installation. See Uploading a file to a service in the *AppWorks Platform Advanced Development Guide*.

Similarly, the installer also creates a dedicated folder for downloading a file from a service (<AppWorks Platform\_installdir>/content/downloadcontent). See Downloading a file from a service in the *AppWorks Platform Advanced Development Guide*.

If you wish to move these folders to a different location, for example, when setting up a highly available cluster, these folders must be moved to a shared file system.

## To move the folders to a different location:

### Modify wcp.properties

Set the related properties in `wcp.properties`:

Property name	Description
<code>com.eibus.web.tools.upload.UploadWritePath</code>	Directory in which the temporary copy of the uploaded file is created. In case of a multinode cluster, all nodes should set this to a path that maps to the same shared file system location.
<code>com.eibus.web.tools.download.DownloadReadPath</code>	Directory containing the download file placed by the service container. In case of a multinode cluster, all nodes must set this to a path that maps to the same shared file system location.

### Set the appropriate permissions

The Web server requires access to the upload and download folders. In the download folder, the service container requires write access and the Web server requires read access. In the upload folder, the web server requires write access and the service container requires read access.

Following are the access rights:

User	DownloadReadPath	UploadWritePath
<AppWorks Platform	read/write	read

User	DownloadReadPath	UploadWritePath
user>		
<Web server user>	read	read/write

where:

- <AppWorks Platform user>: User running AppWorks Platform.
- <Web server user>: User running Web server.

## Document store

Documents are usually stored in a content repository. AppWorks Platform provides Document Store as the facility to work with any content repository. Any document irrespective of the content can be stored in these document stores.

In medical applications, a document can be a soft copy of an X-ray, MRI scans, and so on. Document Store supports the document infrastructural needs for any feature within the AppWorks Platform.

Documents can be attached to the Case Models, Business Process Models, and so on. Document Store can be easily used and requires no configuration after installing the AppWorks Platform. Documents can be added as attachments and can be retrieved later. Documents can also be viewed, uploaded, or downloaded.

Document Store enables you to plug in a content repository so that any existing content repository can be used. AppWorks Platform can work with any content repository server, which has exposed its repository API according to [JSR 170](#) specification or [CMIS](#) specification. There are some repositories, which provide a true implementation of [JSR 170](#) specification, such as the [Jackrabbit](#) from Apache Software foundation. Other repository servers exposing their repository service through a JSR 170 compliant wrapper can also be plugged in to AppWorks Platform.

AppWorks Platform provides the support for following document stores:

<b>Repository</b>	While installing AppWorks Platform, the user is prompted to provide database configuration details. When this option is selected, the repository used by AppWorks Platform is used. For a richer set of document store features, it is recommended to use a commercial document store such as OpenText Content Server.
<b>Jackrabbit</b>	When this option is selected, AppWorks Platform behaves like the client of Apache Jackrabbit.
<b>OpenText Content Server</b>	When this option is selected, AppWorks Platform acts as a Content Server client and all the requests are directed to the configured Content Server.

<b>OpenText Archive Center</b>	Select this option, if the requirement is to integrate with Archive Center using OTDS based authentication.
<b>CMIS</b>	When this option is selected, AppWorks Platform acts as a CMIS client, which can connect to any CMIS 1.1 compliant repository.
<b>OpenText Documentum</b>	When this option is selected, AppWorks Platform acts as a Documentum client and all the requests are directed to the configured Documentum server.
<b>Custom implementation</b>	Select this option if you are already using any content repository or want to use a repository other than the above listed ones.

While installing AppWorks Platform, Repository is chosen as the default repository. See [Creating and modifying document store repository](#) for modifying the default repository setting or creating a new document store application connector. See Document Store configuration interface in the *AppWorks Platform Administrator's Guide* for configuring the required document store type.

Document Store provides you with APIs so that you can work with documents. See Document Store API in the *AppWorks Platform API Guide* for information on document store APIs.

**Note:** By default, Document Store is part of Repository service container. When the Web server and Repository service container are working on the same computer, the following properties point to the locations as indicated:

- `com.eibus.web.tools.upload.UploadWritePath` - Points to <Installation directory>\content\uploadcontent
- `com.eibus.web.tools.download.DownloadReadPath` - Points to <Installation directory>\content\downloadcontent

Ensure that the user under which the web server runs has write permissions on the <Installation directory>\content\uploadcontent folder and the <Installation directory>\content\downloadcontent folder. When the Web server and the Service Container are running on different computers, create a shared location and provide access to both. Also ensure that the above properties point to the shared location. These properties are part of the Web Gateway properties. For more information on Platform Properties, see Global configuration properties for AppWorks Platform in the *AppWorks Platform Administration Guide*.

**Caution:** Upload Gateway stores the uploaded files in the shared directory and passes the directory name to the service container to process the request. A hacker can continuously send upload requests where no service container is available to handle them. In such a case, the disk space will be full in no time. To avoid such a scenario, it is strongly recommended that you specify a size-limit on the shared directory.

## Creating and modifying document store repository

While installing AppWorks Platform, the database configuration details provided by you is chosen by default to place documents. It is recommended to use any repository other than the default AppWorks Platform repository for storing documents. AppWorks Platform provides you Jackrabbit with basic document management support, or OpenText Content Server and OpenText Archive Center with advanced document management capabilities for the purpose of storing documents. It is also possible that a repository is maintained for a specific purpose. For example, while building an application for medical insurance claims, you can use a repository for storing these claims. In such cases, modify the document repository details to point to the repository that you are using.

**Important:** If you are using System organization, you can either modify the repository settings or create a new Document Store application connector to point to a different repository in the System organization. However, it is recommended to create a new Document Store connector.

**Note:** Based on the modifications you make to the document store repository settings, you must do the following:

- If you modify the document store type, restart TomEE after the changes are complete.
- If you modified any other settings, run the following REST APIs:
  - `http://<hostname>:<port>/home/<organization name>/app/documentservices/rest/cache/reset`
  - `http://<hostname>:<port>/home/<organization name>/app/documentservices/rest/cache/flushOrgEntry`

### To modify the repository settings in the System organization:

1. On the Welcome page, click  (System Resource Manager).  
The System Resource Manager window opens, displaying a list of service containers.
2. In the Service Containers area, double-click the **Repository** service container.  
The Repository Properties pane opens.
3. Click the **Document Store** tab.  
The current repository details are displayed.
4. Modify the document repository settings as required.  
For more information on the fields in the Properties - Repository window, see [Document Store Configuration interface](#).
5. Click .  
The changes are saved.

The documents that are created from now on are stored in the specified repository.

### To create a new Document Store application connector (System organization)

You must first remove the Document Store interfaces from the repository service container, and then create a new Document Store application connector.

**To remove the Document Store interfaces:**

1. On the Welcome page, click **System Resource Manager**.  
All the service containers are listed.
2. Click **Show > All Service Groups**.  
All the service groups are listed.
3. Double-click the **Repository** service group.  
The Repository Service Group Properties pane opens.
4. In the Web Service Interfaces area, select Web service interfaces of the type **Document Store**, and then click **Delete**.
5. Click **Save**.

**To create a new Document Store application connector (non-System organization):**

1. Create a Document Store group. See [Creating a service group](#).
2. Configure the Document Store application connector. See [Document Store Configuration interface](#) for different repositories.

**Note:**

- For a non-system organization, perform the above steps to create a Document Store application connector for the required repository. However, if the document store application connector is not configured in a non-system organization, then fallback is done on the System organization.

***Document Store Configuration interface***

The following fields are displayed in the Store Details area of the Configuration page.

Field	Description
Store Name	Type a name for the repository configuration.
Store Type	Type of repository. Select one of the following: <ul style="list-style-type: none"><li>■ <a href="#">AppWorks Platform Repository</a></li><li>■ <a href="#">OpenText Content Server (OTCS)</a></li><li>■ <a href="#">OpenText Archive Center</a></li><li>■ <a href="#">Jackrabbit</a></li><li>■ <a href="#">CMIS</a></li><li>■ <a href="#">OpenText Core</a></li><li>■ <a href="#">OpenText Documentum</a></li><li>■ <a href="#">Others</a> (Select to use any other repository for managing documents)</li></ul>

Field	Description
Brava Server URL	<p>Optional.</p> <p>Type the Brava Server URL to use for viewing documents stored in the repository, for example, <code>http://&lt;server&gt;:&lt;port&gt;</code>.</p> <p><b>Note:</b> If you configure the Brava Server URL, then in the application, after you open an entity instance, the Open In Viewer tab is displayed in the business workspace, business attachment, and contents panels. Upload a document, select it, and then click <b>Open In Viewer</b>. The document opens in Brava Viewer.</p>

Depending on the selected store type, different fields are displayed in the Repository Configuration Details area.

### AppWorks Platform Repository

The following fields are displayed in the Repository Configuration Details - XDS area when you select the AppWorks Platform Repository option.

Field	Description
Database Configuration	<p>Select the database configuration. See <a href="#">Managing database configurations</a>.</p> <p><b>Note:</b> The Cordys System option contains the database configuration information that was provided when installing AppWorks Platform.</p>
Repository Root	Optional. Repository root folder of the document.

### OpenText Content Server (OTCS)

#### Before you begin:

- Ensure that OpenText Content Server is installed on your computer and you have administrator privileges to connect to Content Server.
- Add the AppWorks Platform resource and Content Server resource as a OTDS resource in Security Administration. See [Managing an OTDS resource](#).

The following fields are displayed in the OpenText Content Server Configuration Details area when you select the OpenText Content Server option:

■ Template Initialization Details:

Field	Description
Organizational administrator	<p>ID of the organization administrator. This user context is used to start the document store connector and perform some operations in Content Server, such as creating the root folder and default categories.</p> <p>This is optional. If no value is entered, the document store internally uses the <code>otadmin@otds.admin</code> user.</p> <p><b>Note:</b> This organizational administrator ID or the <code>otadmin@otds.admin</code> user must exist in AppWorks Platform and OpenText Content Server and must have the following privileges in Content Server:</p> <ul style="list-style-type: none"> <li>• Log-in enabled</li> <li>• Public Access enabled</li> <li>• Full permissions on Content Server Categories volume</li> <li>• Member of the default group.</li> <li>• Added as a user for the Category and Category Folder object types. To do this, navigate to Content Server Administrator -&gt; Object Privileges , select category and add the user to that object type. Repeat for Category Folder object type.</li> </ul>

■ OTDS Details:

Field	Description
Resource Name	OTDS resource created for Content Server. See <a href="#">Managing an OTDS resource</a> .
Space	Based on the resource name selection, this field is automatically populated.

The External Systems section contains the following details:

■ Content Server Details

Field	Description
End Point URL	<p>HTTP URL, which serves as a base URL to connect to all the Web service operations of OTCS.</p> <p>Based on the Content Server deployment on a Web server, enter the End Point URL:</p> <ul style="list-style-type: none"> <li>When Content Server is deployed on Tomcat, enter the URL as <code>http://&lt;OTCS server&gt;:&lt;portnumber&gt;/cws/services/DocumentManagement</code></li> <li>When Content Server is deployed on IIS, enter the URL as <code>http://&lt;OTCS server&gt;:&lt;portnumber&gt;/cws/DocumentManagement.svc</code></li> </ul>
Repository Root	<p>Optional.</p> <p>Repository root folder, which serves as the base folder within a specific workspace.</p> <p>If no document root is provided, the workspace is treated as the personal workspace.</p> <p><b>Note:</b> In case of xECM, this field value is not considered for business workspace creation.</p>

■ Extended ECM Configuration

Field	Description
External System Id	<p>Provide the external system ID of AppWorks Platform as configured in Content Server as shown below.</p> <ol style="list-style-type: none"> <li>Navigate to <b>Content Server &gt; Admin page &gt; Extended ECM &gt; Configure Connections to External Systems</b>.</li> <li>Copy the Logical System Name value to this field.</li> </ol> <p>After the value is entered in this field, other field values are populated automatically.</p>
API web service URL	WSDL URL to access xECM related Web services that are deployed in OTCS, for example: <code>http://&lt;computer name provided in the End Point URL&gt;:&lt;portnumber&gt;/otsapxecm/services/ECMLink</code> .
CGI Path	<p>Filled automatically based on the URL entered in End Point URL, for example: <code>http://&lt;computer name provided in the End Point URL&gt;:&lt;port_number&gt;/&lt;URL_prefix&gt;/livelink.exe</code>. This is the OTCS business workspaces URL, where:</p> <ul style="list-style-type: none"> <li>&lt;portnumber&gt; - Port on which your Web server listens.</li> </ul>

Field	Description
	<ul style="list-style-type: none"> <li>&lt;URL_prefix&gt; - URL prefix mapped to the &lt;Content_Server_Home&gt;/cgi/ folder of Content Server.</li> <li>livelink.exe - Default value. It can be any contentserver.exe, for example, cs.exe.livelink.exe.</li> </ul> <p><b>Note:</b> The .exe extension is not used for UNIX or Linux, or when Content Server works with an application server.</p>
Support Path	<p>Filled automatically based on the URL entered in End Point URL, for example: http://&lt;machine name provided in the End Point URL&gt;:&lt;portnumber&gt;/img. This holds CSS, JavaScript files, and images related to the folder browser widget. This appears with the Content Server UI module.</p> <p><b>Note:</b> The /img, also known as the virtual directory alias, is mapped to the support directory during the installation of Content Server. By default, the support directory is located at the root installation level: /installation_path/support. It is normally not necessary to change /img; however, if you modify it, you must make the change in Content Server and the Web server. The default value for this is /img.</p>
Synchronize Workspaces Manually	<p>This option is not selected by default.</p> <p>When entity instances are created in runtime, corresponding business workspace instances are created automatically in Content Server.</p> <p>If this option is selected, a business workspace is not created automatically in Content Server. You must select the record and click <b>Synchronize Workspace</b> in your application to create a business workspace in Content Server.</p>

## OpenText Archive Center

### Before you begin:

- Ensure that OpenText Archive Center is installed on your computer and you have administrator privileges to connect to the Archive Server.
- Add the AppWorks Platform resource and Content Server resource as a OTDS resource in Security Administration. See [Managing an OTDS resource](#).

The following fields are displayed in the OpenText Archive Center Configuration Details area when you select the OpenText Archive Center option.

- Template Initialization Details:

Field	Description
Organizational administrator	<p>ID of the organization administrator. This user context is used to start the document store connector and perform some operations in OpenText Archive Center, such as creating the root folder and default categories.</p> <p>This is optional. If no value is entered, the document store internally uses the user <code>otadmin@otds.admin</code>.</p> <p><b>Note:</b> The organizational administrator or the <code>otadmin@otds.admin</code> user must exist both in AppWorks Platform and OpenText Archive Center. The user must also be a member of the default group in OpenText Archive Center.</p>

- OTDS Details:

Field	Description
Resource Name	OTDS resource created for OpenText Archive Center. See <a href="#">Managing an OTDS resource</a> .
Space	Based on the resource name selection, this field is automatically populated.

■ Archive Server details:

Field	Description
Archive Center Resource	OTDS resource created for the OpenText Archive Center server. See <a href="#">Managing an OTDS resource</a> .
Binding Type	Select the CMIS binding that must be used. Based on the CMIS binding, the Repository URL changes. <b>Note:</b> To use Web Services binding, configure the document store with Apache-CXF. See <a href="#">Configuring document store connector with Apache CXF library</a> .
Repository URL	HTTP URL based on the selected CMIS binding, which serves as a base URL to connect to OpenText Archive Center. Example URLs for various bindings for Documentum content repository: <ul style="list-style-type: none"> <li>AtomPub: <code>http://&lt;Archive Center server&gt;:&lt;portnumber&gt;/&lt;as_cmis&gt;/atom</code></li> <li>WebService: <code>http://&lt;Archive center server&gt;:&lt;portnumber&gt;/&lt;as_cmis&gt;/services/cmis?wsdl</code></li> <li>Browser: <code>http://&lt;Archive center server&gt;:&lt;portnumber&gt;/&lt;as_cmis&gt;/browser</code></li> </ul>
Repository Id	Unique ID of the repository, which needs to be connected. In Archive Center, this corresponds to the archive name of the collection.
Repository Root	Optional. Repository root folder, which serves as the base folder within a specific repository denoted by the repository ID. If no document root is provided, the default root is considered.
Datasource Id	ID of the data source of type CMIS associated with the collection. Mandatory for the Archive Center 10.5 cloud edition and later releases.

### Jackrabbit

The following fields are displayed in the Repository Configuration Details - Jackrabbit area when you select the Jackrabbit option.

- Authentication Details

<b>Field</b>	<b>Description</b>
User Name	User name to access the server that stores the document.
Password	Password to access the server that stores the document.

- Repository Details

<b>Field</b>	<b>Description</b>
Repository URL	URL of the document within the repository.
Repository Root	Optional. Repository root folder of the document.

## CMIS

The following fields are displayed in the Repository Configuration Details - CMIS area when you select the CMIS option.

- Authentication Details

<b>Field</b>	<b>Description</b>
User Name	User name to access the server that stores the document.
Password	Password to access the server that stores the document.

- Repository Details

<b>Field</b>	<b>Description</b>
Binding Type	Select the CMIS binding to use. Based on the CMIS binding, the Endpoint Repository URL changes.  Note: To use Web Services binding, configure the document store with Apache-CXF. See Configuring document store connector with Apache CXF library.
Repository URL	HTTP URL based on selected CMIS binding, which serves as a base URL to connect to the content repository.  Example URLs for various bindings <ul style="list-style-type: none"> <li>• AtomPub: <code>http://&lt;content repository server&gt;:&lt;portnumber&gt;/&lt;as_cmis&gt;/atom</code></li> <li>• WebService: <code>http://&lt;content repository server&gt;:&lt;portnumber&gt;/&lt;as_cmis&gt;/services/cmis?wsdl</code></li> <li>• Browser: <code>http://&lt;content repository server&gt;:&lt;portnumber&gt;/&lt;as_cmis&gt;/browser</code></li> </ul>
Repository Id	Unique ID of the repository to be connected.
Repository root	Optional.  Repository root folder of the document.

### OpenText Core

The following fields are displayed in the Repository Configuration Details - OpenText Core area when you select the OpenText Core option.

- Authentication Details

<b>Field</b>	<b>Description</b>
User Name	User name to access the OpenText Core server.
Password	Password to access the OpenText Core server.
Authentication Handler Name	Type the authentication handler name that is configured in AppWorks Platform OTDS for OpenText Core. See and the section on Authentication Handlers in the <i>OpenText Directory Services: Installation and Administration Guide</i> .

- Repository Details

Field	Description
Binding Type	Browser option is selected by default.
Repository URL	Type the browser URL. For example, <code>http://&lt;OpenText Core server&gt;:&lt;portnumber&gt;/&lt;core_cmis_server&gt;/browser</code>
Core Base URL	Type the OpenText Core URL. For example, <code>http://&lt;OpenText Core server&gt;:&lt;portnumber&gt;</code>
Repository Root	Optional. Repository root folder of the document.
Repository Id	Unique ID of the repository to be connected.

### OpenText Documentum

The following fields are displayed in the OpenText Documentum Configuration Details area when you select the OpenText Documentum option.

- Template initialization details:

Field	Description
Organizational administrator	ID of the organization administrator. This user context will be used to start the document store connector and create the root folder in OpenText Documentum. <b>Note:</b> The user must have Create Cabinet privilege in OpenText Documentum.

- Authentication Details

Field	Description
Authentication Handler Name	Type the authentication handler name that is configured in AppWorks Platform OTDS for OpenText Documentum. See <a href="#">Configuring authentication handler for a repository</a> and the section on Authentication Handlers in the <i>OpenText Directory Services: Installation and Administration Guide</i> .

## ■ Repository Details

Field	Description
Media URL Policy	<p>Policy to set the URL from which the document content must be retrieved.</p> <ul style="list-style-type: none"> <li>• Default - the URL as configured in the Documentum repository to retrieve the document content. This option is selected by default.</li> <li>• Local - the Content Media resource URL from the REST server to retrieve the document content.</li> </ul>
Repository Root	<p>Optional.</p> <p>This value is used as the base location to store your documents. If you do not provide a repository root, then the default cabinet of the repository (the one created during the Documentum configuration) is considered as the base location.</p>
Repository REST Endpoint	<p>Type the Documentum REST Endpoint.</p> <p>For example, <code>http://&lt;OpenText Documentum server&gt;:&lt;portnumber&gt;/dctm-rest/repositories/&lt;Repository name&gt;</code></p>

## Others

AppWorks Platform provides the `IDocumentStore` interface. Implement this interface to use any repository other than the AppWorks Platform Repository or Jackrabbit.

See the code for implementing the `IDocumentStore` interface.

```
public abstract class JSRDocumentStoreClient implements IDocumentStore
{
    public boolean initializeStore(StoreConfiguration configuration) throws
    DocumentStoreException {} //more methods need to be defined
}

public class XDSDocumentStore implements IDocumentStore
{
    //more methods need to be defined
}
```

- Repository Configuration Details - Others

Field	Description
Implementation Class	Name of the class that implements the IDocumentStore interface. For more information about the IDocumentStore class, see the Document Store package in the Java documentation.
Classpath	Classpath of the class that is implementing the IDocumentStore interface.
External Configuration XML	Paste any additional information that is required to access the repository in this field in the XML format. These details are passed to the implementation class. The contents of this field pass to the StoreConfiguration object as displayed in the code for implementing the iDocumentStore interface. The StoreConfiguration object in turn passes to the initialize method of the implementation class. For more information, see the Document Store package in the Java documentation.
Repository Root	Repository root folder of the document.

## Configuring document store connector with Apache CXF library

AppWorks Platform document store connector uses JAX-WS APIs internally to integrate with various repositories, such as Content Server, and CMIS compliant repositories. Document store, by default, uses JAX-WS implementation that comes with Java; this is a basic implementation. [Apache-CXF](#) is an advanced implementation of JAX-WS and can be used with the document store connector.

### To configure the document store connector with the Apache CXF library:

Generate the `cxfcp.jar` and configure with Apache-CXF.

#### To generate the `cxfcp.jar`:

1. Download the Apache CXF binaries from <http://cxf.apache.org/download.html> and extract it to a folder on the file system.
2. Generate the `cxfcp.jar` as follows:

**Note:** Apache-CXF includes a lot of libraries and jars. To avoid using a lengthy classpath for the document store service container, the following steps are recommended to generate a single jar, such as `cxfcp.jar` that internally refers to multiple Apache CXF client jars.

- a. Ensure that [ANT](#) is installed on the system and the environment variable `ANT_HOME` is mapped to the ANT installation directory.

- b. Create an ANT build file in the < APACHE\_CXF\_HOME>\lib folder. The build file must contain all the jars in the < APACHE\_CXF\_HOME>\lib folder listed in the manifest.
- c. Generate a single `cxfcp.jar` using the ANT build script.

```
D:\apache-cxf\lib>ant -f build.xml
Buildfile: D:\apache-cxf\lib\build.xml
cxfcp_jar:
[jar] Building MANIFEST-only jar: D:\apache-cxf\lib\cxfcp.jar
BUILD SUCCESSFUL
Total time: 0 seconds
D:\apache-cxf\lib>
```

- 3. A `cxfcp.jar` is generated in the same directory < APACHE\_CXF\_HOME>\lib, which internally refers to all the Apache CXF libraries. See this jar in the CLASSPATH of any Java application wherever you intend to refer to Apache CXF JAX-WS client libraries.

#### To configure document store with Apache-CXF:

1. Select the document store service container. Right-click and open the document store service container properties.
2. Navigate to the **JRE Configuration** tab.
3. Update the JRE classpath with the absolute path to `cxfcp.jar`, which is generated earlier.
4. Add the following JVM argument to switch to the Apache CXF JAX-WS implementation.

```
-Djavax.xml.ws.spi.Provider=org.apache.cxf.jaxws.spi.ProviderImpl
```

5. Restart the service container.

## Configuring access URLs

### Introduction

You can use different URLs to access AppWorks Platform. If a cluster consists of multiple nodes, then each node has its own URL. The URL used to access the cluster can be different when accessing it from the company network or the internet, for example, when a reverse proxy is configured to support a DMZ architecture or SSL offloading.

This topic contains the different properties to define these URLs, and instructions to configure the load balancer or reverse proxy.

AppWorks Platform comprises the following URL properties :

Property name	Description
<code>com.cordys.node.url</code>	The URL of the local gateway of a

	(cluster) node.
com.cordys.internal.cluster.url	The URL that AppWorks Platform uses to talk from within the cluster to the load balancer in front of the cluster.
com.cordys.public.cluster.url	The URL that users must use when accessing the AppWorks Platform cluster.

Fall back occurs from bottom to top in this list. When the public cluster URL is not set, the internal cluster URL is used instead. If the internal cluster URL is not set, the node URL is used.

Use the [Management console](#) to change these properties.

Use the URL properties for requests initiated within AppWorks Platform. Use the HTTP headers for requests initiated by a client, for example, a browser.

**Note:** Chained reverse proxies: If SSL is enabled, and the Certificate Authority is not trusted by AppWorks Platform or the certificate is self-signed, add it to the trust store. Follow the instructions in [Adding a new certificate](#).

**Note:** Notification: Change the URL prefix in the [Notification Service configuration interface](#) properties as well. Otherwise, the task URL becomes invalid and a task delivered through email does not open.

Set it to the public cluster URL or its fall back.

**Note:** Legacy: If you change settings after migrating from an older version, remove the following:

- Legacy `BASE_URL` variable for [SAML 2.0 variables](#) and [OTDS variables](#).
- Legacy properties `web.server.ssl`, `cordys.baseurl.protocol` and `com.eibus.web.wsdl.gateway.hostname`.

## Configuring an individual node

The property `com.cordys.node.url` must point to the gateway of the local node and is set by the installer. It must be updated, for example, when SSL is enabled on the Web server or when the port number is changed.

Property name	Sample values
<code>com.cordys.node.url</code>	<code>http://localserver</code> <code>https://localserver</code> <code>https://localserver:8443</code>

## Configuring an internal load balancer

To run AppWorks Platform in high availability mode, deploy everything at least twice, with a load balancer in front of the Web servers. To ensure that load balancing and failover in a cluster setup work as expected:

1. Configure the URL of the load balancer on every cluster node, by setting the property `com.cordys.internal.cluster.url`
2. Route the HTTP requests initiated from within the cluster through the load balancer.

Property name	Sample value
<code>com.cordys.internal.cluster.url</code>	<code>http://processplatform.acme.lan:8080</code>

## Configuring a public reverse proxy

If the deployment involves multiple load balancers and reverse proxies, for example, when users access the system through a proxy, but requests from within the cluster are to be routed through a load balancer, then the external cluster URL must be configured on every cluster node, by setting the property `com.cordys.public.cluster.url`.

Property name	Sample value
<code>com.cordys.public.cluster.url</code>	<code>https://invoicing.acme.com</code>

## Configuring Content Server with AppWorks Platform

OpenText Content Server (OTCS) is an industry leading content management software with a rich set of features. AppWorks Platform is integrated with Content Server through the document store connector. This helps applications deployed on AppWorks Platform to store and retrieve documents and perform operations on them, such as check-in, check-out, and metadata.

### Before you begin:

Ensure that Content Server is installed and configured with OTDS. You must have the Resource ID of the Content Server resource that is configured in OTDS.

Configuring Content Server with AppWorks Platform involves the following steps:

- Deploying the WAR file
- Creating the AppWorks Platform resource in OTDS
- Enabling impersonation for AppWorks Platform resources in OTDS
- Creating AppWorks Platform users in OTDS
- Activating AppWorks Platform resources

- Adding the Content Server resource to AppWorks Platform
- Performing a cleanup before creating the document store service container
- Creating the document store service container
- Granting access to the root folder and its subfolders

## Deploying the WAR file

**Note:** This step is not required if Content Server is installed with Tomcat and the `cws.war` file is deployed in the Tomcat Web server.

### To deploy the WAR file:

1. Sign in to the Tomcat Web server.
2. Select **Choose File** from the War file to deploy area, browse to `<OTCS_installdir>/webservices/java/webapps`, and then select the `cws.war` file for Content Server 10.5/16.
3. Click **Deploy**.

The `cws.war` file is deployed.

## Creating the AppWorks Platform resource in OTDS

**Note:** This step is not required if AppWorks Platform is configured with OTDS.

### To create the AppWorks Platform resource in OTDS:

1. Connect to the OTDS server using OTDS Web Administration (`http://<tomcat server>:<tomcat port>/otds-admin`) and sign in as an administrator.
2. In OTDS, click **Resources** on the navigation bar.  
The Resources dialog box opens.
3. Click **Add** to create a resource.  
The New Resource dialog box opens.
4. For:

Resource Name	Type a name.
Display Name	Type a name to display.
Description	Type a meaningful description.

5. Click **Next**.  
The New Resource > Synchronization dialog box opens.

**Note:** Do not click **Save**. Wait to save this resource until all the parameters are configured.

6. Keep the default selections. Do not select **User and group synchronization**.
7. Click **Next**.  
The New Resource > Attribute Mappings dialog box opens.  
**Note:** Do not click **Save**. Wait to save this resource until all parameters are configured.
8. In the New Resource > Attribute Mappings dialog box, in the Authentication principal attribute list, select **oTExternalID1**.
9. Click **Save**.  
The Resource Activation dialog box opens.
10. In the Resource Activation dialog box, copy the resource identifier, and then click **OK**.  
**Note:** Do not click **Verify Activation**.  
The AppWorks Platform resource is created in OTDS. You must provide the resource identifier when integrating this resource with AppWorks Platform.

## Enabling impersonation for AppWorks Platform resources in OTDS

### To enable impersonation for the AppWorks Platform resource in OTDS:

1. In OTDS, click **Resources** on the navigation bar.  
The Resources dialog box opens.
2. Select the AppWorks Platform resource in OTDS, and then click **Actions > Impersonation Settings**.  
The Impersonation Settings dialog box opens.
3. Select **Allow this resource to impersonate its own users**.
4. Click **OK**.

Impersonation is enabled for the AppWorks Platform resource.

## Creating AppWorks Platform users in OTDS

**Note:** This step is not required if AppWorks Platform is configured with OTDS.

### To create AppWorks Platform users in OTDS:

1. In OTDS, click **Partitions** on the navigation bar.  
The Partitions dialog box opens. The list of available partitions is displayed.
2. Select the partition to add the user. This partition must have been added to an AppWorks Platform Resource access role.
3. To add the user:
  - a. Click **Actions > View Partition Members**.  
A dialog box opens.
  - b. Click **Add > New User**.  
The New User dialog box opens.

- c. Enter the mandatory details on the General and Account tabs. The user name entered must be the same as the AppWorks Platform user ID.

The AppWorks Platform user is added. Add all the AppWorks Platform users who must access Content Server.

## Activating AppWorks Platform resources

**Note:** This step is not required if AppWorks Platform is configured with OTDS.

### To activate the AppWorks Platform resource from AppWorks Platform:

1. Open a browser, and access **AppWorks Platform > Security Administration**.  
The Security Administration page opens.
2. Click the **OTDS Resources** tab.  
The OTDS Resources page opens.
3. Click **Add (+)**.  
The Resource Details area is displayed with the OTDS resource details.
4. In the Resource Details area, for:

Resource Name	Type the name with which the OTDS resource must be identified, for example: <b>ProcessPlatform</b>
Space	Select Shared from the list to identify the space for which the OTDS resource is active.  <b>Note:</b> An OTDS resource in the Shared space can be used in all the organizations. An OTDS resource in the Organization space only can be used only in the current organization.
OTDS Server URL	Type the URL to access the OTDS server in the following format: <code>http://&lt;OTDS host name&gt;:&lt;port number&gt;</code>  This must be a valid URL, beginning with either <code>http</code> or <code>https</code> , for example: <code>http://server1:8080</code>
Resource ID	Type the resource identifier of the previously created AppWorks Platform resource.

5. Click **Save**, and then click **Activate**.  
A notification dialog box opens indicating that the resource is activated.
6. Click **Close**.

## Adding the Content Server resource to AppWorks Platform

### To add the Content Server resource to AppWorks Platform:

1. Access **AppWorks Platform > Security Administration**.  
The Security Administration page opens.
  2. Click the **OTDS Resources** tab.  
The OTDS Resources page opens.
  3. Click **Add (+)**.  
The Resource Details area is displayed with the OTDS resource details.
  4. In the Resource Details area, for:
- |                 |                                                                                                                                                                                                                                                                                       |
|-----------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Resource Name   | Type the name with which the OTDS resource must be identified, for example: <b>AppWorksPlatform</b>                                                                                                                                                                                   |
| Space           | Select Shared from the list to identify the space for which the OTDS resource is active.<br><br><b>Note:</b> An OTDS resource in the Shared space can be used in all the organizations. An OTDS resource in the Organization space only can be used only in the current organization. |
| OTDS Server URL | Type the URL to access the OTDS server in the following format:<br><code>http://&lt;OTDS host name&gt;:&lt;port number&gt;</code><br><br>This must be a valid URL, beginning with either <code>http</code> or <code>https</code> , for example:<br><code>http://server1:8080</code>   |
| Resource ID     | Type the resource identifier of the Content Server resource in OTDS that is created for OTDS authentication with Content Server.                                                                                                                                                      |

5. Click **Save**.

## Performing a cleanup before creating the document store service container

**Note:** This step is not required if the document store service container is intended for another organization.

### To perform a cleanup before creating the document store service container:

1. Access **AppWorks Platform > System Resource Manager**.  
The System Resource Manager window opens.
2. Open the properties for the Repository service group.
3. Select the Document store Web Service Interfaces (interfaces with Organization/Package as Cordys Document Store) from the Repository service group properties, and then click **Delete**.
4. Click **Save**.

## Creating the document store service container

### To create the document store service container:

1. Click **New** on the toolbar and select the Document Store application connector from the Service Group wizard.
2. Type the name for the service group, select all the Web Service Interfaces in the table, and then click **Next**.  
The Provide Service Container details page appears.
3. Click the **JRE Configuration** tab.
4. Type the classpath of the following commons-logging and otcs-client-stubs JAR files available in the <AppWorks Platform\_installdir>/ext/, for example:

```
C:\Program Files\OpenText\ProcessPlatform\defaultInst\ext\commons-logging-1.1.1.jar;C:\Program Files\OpenText\AppWorks Platform\defaultInst\ext\otcs-client-stubs-10.0.0.2645.jar
```

5. Click **Add** and type the following JVM property:

```
-Dorg.w3c.dom.DOMImplementationSourceList=com.sun.org.apache.xerces.internal.dom.OMXSImplementationSourceImpl
```

6. Click **Next**.  
The Provide details for Document Store Connector page appears.
7. Select **OpenText Content Server** from the Store Type list.  
The OpenText Content Server Configuration Details area is displayed.
8. In the Store Details area, type a name for the Document Store and type the Brava Server URL in the corresponding fields.
9. In the OTDS Details area, for:

OpenText AppWorks Platform resource	<ol style="list-style-type: none"> <li>a. Click the browse icon for Resource Name. The OTDS_Resources dialog box opens. The list of available resources is displayed.</li> <li>b. Select the required OpenText AppWorks Platform resource, and then click <b>OK</b>. The Resource Name and Space fields are populated.</li> </ol>
OpenText Content Server resource	<ol style="list-style-type: none"> <li>a. Click the browse icon for Resource Name. The OTDS_Resources dialog box opens. The list of available resources is displayed.</li> <li>b. Select the required OpenText Content Server Resource,</li> </ol>

---

	<p>and then click <b>OK</b>. The Resource Name and Space fields are populated.</p>
--	----------------------------------------------------------------------------------------

---

10. In the Content Server Detail area, for:

End Point URL	Type the Content Server end point URL, for example: <code>http://&lt;tomcat server&gt;:&lt;tomcat port&gt;/cws/services/DocumentManagement</code>
Repository Root	Type a name for the repository root. A folder is created in Content Server with this name. Uploaded documents are stored in this Content Server folder.

---

11. Click **Next**, and then click **Finish**.

## Granting access to the root folder and its subfolders

### To grant access to the root folder and its subfolders:

1. Access **Content Server**.
2. Go to the Repository root folder created while creating the document store connector above, click **Functions**, and then select **Permissions**.
3. Under Default Access, click **Public Access**.  
The Edit Public Access Permissions area is displayed.
4. Under Access, select the **Edit Permissions**.
5. In the Apply To list, select **This Item & Sub Items**.
6. Select all the **Sub-Item** options, and then click **Update**.  
The Apply Permissions to Sub-Items window opens.
7. Click **OK**.

Access is granted to the root folder and its subfolders.

Content Server is configured with AppWorks Platform.

## Configuring HTTP connector

**Note:** After installing the HTTP connector application package, you must create a new service group and service container.

### To configure the HTTP connector:

1. On the **Welcome** page, click  (System Resource Manager).  
The System Resource Manager window opens.
2. Click  (Show All Service groups) in the toolbar.  
The Service Groups App Palette opens.

3. Click  (New) in the **Service Groups App Palette**.

The New Service Group wizard opens.

4. Select the **HTTP Connector** application connector.
5. Select a **Web Service Interface**.

**Note:** The wizard does not continue if you do not select at least one Web service interface.

6. Provide the following details to configure the HTTP connector:

**Note:** The HTTP connector supports only basic authentication.

Parameter	Description
Username	Optional: Username to authenticate at proxy server.
Port	Optional: Port of the proxy server to be used.
Password	Optional: Password to use for authentication at proxy server.
Host	Optional: Host name of the proxy server to be used. Required only when request needs to go through proxy.
Configuration File Path	Path to the configuration file in the XML store. This path is relative to <code>OpenText/HttpConnector</code> in the XML store. This is normally deployed as part of the application package and use HTTP Connection Manager in the Welcome page to modify the configuration details.

## Managing HTTP connections

When an application that uses services of the type HTTP is deployed, configuration and connection details required are deployed in the XML store at `OpenText/HttpConnector`. The configuration file path is stored in the HTTP service container and connector configuration. These configurations might require modification, for example, an update to the authentication details.

### To modify the connection configurations:

1. On the **Welcome** page, click **HTTP Connection Manager**.

**Note:** You must have administrator role to perform this task.

2. From the list, select the HTTP service group to be modified.

**Note:** Ensure that the service containers have started. All the connections available in the configuration are listed.

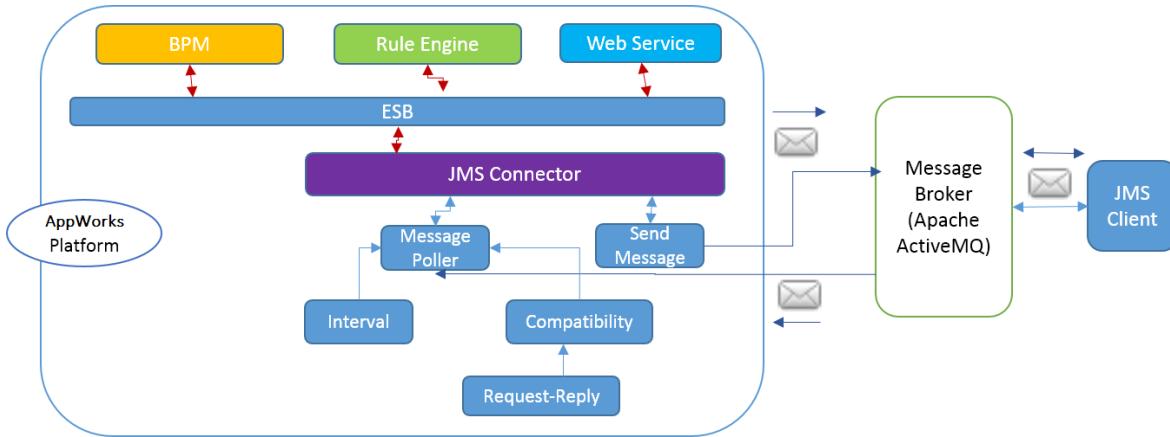
The following details are displayed:

Parameter	Description
Name	Read-only name of the connection.
Description	Optional: Description of the connection.
URL	Server endpoint URL. It can include a part or the complete URL of the service endpoint.
Username	Optional: Username for authentication at the service endpoint.
Password	Optional: Password for authentication at the service endpoint.
Authenticate Always	Optional: Always sends the authentication information. It is also known as preemptive authentication. The default value is False.
Check Certificate	Optional: Used with HTTPS connections only. Use it as an option to check the SSL certificate. In a test environment, you can choose to ignore any errors while validating the SSL certificate. When it is set to False, it accepts any certificate:valid, invalid, expired, or not expired. The default value is False.
Use Proxy	Contains the HTTP connector configuration proxy details. This is the default option. Clear the check box to ignore the proxy details in the HTTP connector configuration.

**Note:** For more information on web service implementation protocol using HTTP connector, see [HTTP connector service contract](#).

## Configuring the JMS connector

The JMS connector is an integral part of AppWorks Platform and is shipped with other connector stacks. It supports distribution transactions and message persistence functionality. The JMS connector can be used with other brokers as well, but version 3.0 is being shipped with Apache ActiveMQ support.



Following are the key features of the JMS connector:

- The JMS connector follows the pull and push mechanism to read messages from the queue or topic.
- The Interval poller reads messages based on a preset time interval following the pull mechanism. On message arrival, the broker pushes the message to the connector and triggers the Compatibility poller.
- The Request-Reply poller is an extension of the Compatibility poller. The poller performs an additional job to serve the request-reply scenarios from AppWorks Platform.

Following are the poller types:

Poller type	Description
Compatibility	When the poller is started, it opens a transacted session to the JMS provider and registers a consumer. Each message that arrives is immediately picked up, and after processing each message, the session is either committed or rolled back.
Interval	At each specified period of time, the connector checks the queue to see if there are any messages to process.
Custom	Extend one of the default pollers or write your own poller.
Request-Reply	The specialized poller handles the request-reply scenarios from AppWorks Platform.

### Before you begin:

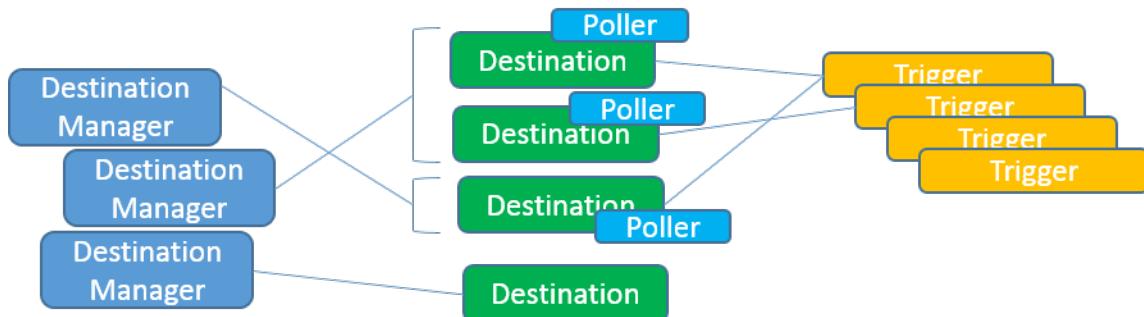
- Ensure that Java 8 is installed.
- Ensure that you have the JMS Administrator role to configure JMS.
- Configure the JMS connector for reliable messaging. See [Configuring service containers for reliable messaging](#).

## GUI configuration

JMS connector configuration is divided into three parts:

- General
- Triggers
- Destination Managers

See the relationship of the configuration elements.



### To configure JMS connector:

1. On the Welcome page, click **JMS Connector Configuration**.  
The JMS Connector Configuration window opens.
2. To create a configuration, enter the required information on the General, Triggers, and Destination Managers tabs. See [JMS Connector Configuration interface](#).

## Service container configuration

Along with the GUI configuration, you must configure the service container for JMS configuration.

### To configure the JMS service container:

1. On the Welcome page, click **System Resource Manager**.  
The System Resource Manager window opens. The available service containers are displayed.
2. Select **OpenText JMS Connector**, and then right-click **Properties**.  
The JMS properties are displayed.
3. Click the **JRE Configuration** tab, and then perform the following configurations:
  - a. In Classpath, add the ActiveMQ related jars, for example, <folder location>\transactions.jar;<folder location>\transactions-api.jar;<folder location>\transactions-jta-3.8.0.jar;<folder location>\activemq-all-5.6.0.jar.
  - b. In the JVM Property section, add:  
-

```
Dcom.cordys.transport.jms.jndifile=<installdir>\config\reliableMessa
ging.properties
```

- c. Click the **OpenText JMS Connector** tab, and then select the configuration to attach to the service container.

**Note:** The configuration values in the list are available after setting up the JMS connector through JMS Connector Configuration. See [GUI configuration](#).

4. Save and restart the configuration.

## Configuring the E-mail connector

The E-mail connector enables users to send and receive emails. It supports single mail servers for both incoming and outgoing emails. Users can send simple emails with the `SendMail` SOAP request using the operation name `SendMailoperation` and construct more complex emails with the `SendMailMIME` SOAP request using the operation name `SendMailMIME`. You can specify email attachments with either operation.

You can retrieve email on demand with the `GetMail` or `GetMails` SOAP requests using the corresponding operation names `GetMailoperation` or `GetMailsoperation`. You can also retrieve emails automatically using the mailbox configuration that defines the recipient's email address to monitor and the action to perform when an email is received for the specified recipient. You can retrieve email attachments with either on demand or automated retrieval of emails.

The E-mail connector is automatically deployed when AppWorks Platform is installed. This topic covers configuring the 3.x series of the E-mail connector.

### Before you begin:

- Ensure that the E-mail connector application package was deployed during the AppWorks Platform installation.
- Ensure that you have an MS-SQL Server, Oracle, or PostgreSQL database to hold the processed emails.

## Creating a database configuration

### Note:

- You need to create a database configuration only if you do not intend to use the Cordys System database with the E-mail connector.
- If you are creating a database configuration, you must create the required tables using the supplied SQL files specific to the database you are using. The SQL files are located at:

```
<AppWorks_Platform_installdir>\components\emailconn\dbschema.
```

You can choose when to create the database configuration:

- before the E-mail connector configuration
- during the E-mail connector configuration

This section describes creating the database configuration before configuring the E-mail connector.

**To create a database configuration before configuring the E-mail connector:**

1. On the Welcome page, click  (System Resource Manager).  
The System Resource Manager window opens and displays the service containers.
2. On the toolbar, click  (Manage Database Configurations).  
The Manage Database Configurations page displays.
3. Click  (Insert).  
A new row is appended to the table and a section displays to enter the database configuration details.
4. For Name and Description, enter the name and description of the database configuration. The description is optional.
5. Enter the relevant details in the corresponding fields. For information about the fields, see [JDBC details interface](#).

**Note:**

- To create a database, select Create New Database and enter the required details for DBA Name and DBA Password.
  - You can create a tablespace while creating a database (for PostgreSQL) or new user (for Oracle) if you select Oracle Thin/OCI or PostgreSQL from the JDBC driver list. See Creating a Tablespace.
6. Click  (Test Connectivity) at the top of the page to test the connection.
  7. Click  (Save).  
A database configuration is created and the name, description and, organization (under which the database configuration is created) are displayed in the new row.

**Note:** For a new user in Oracle, only Create permissions are granted to the user; for other permissions, you must connect to the Oracle server using the client and select the permissions explicitly.

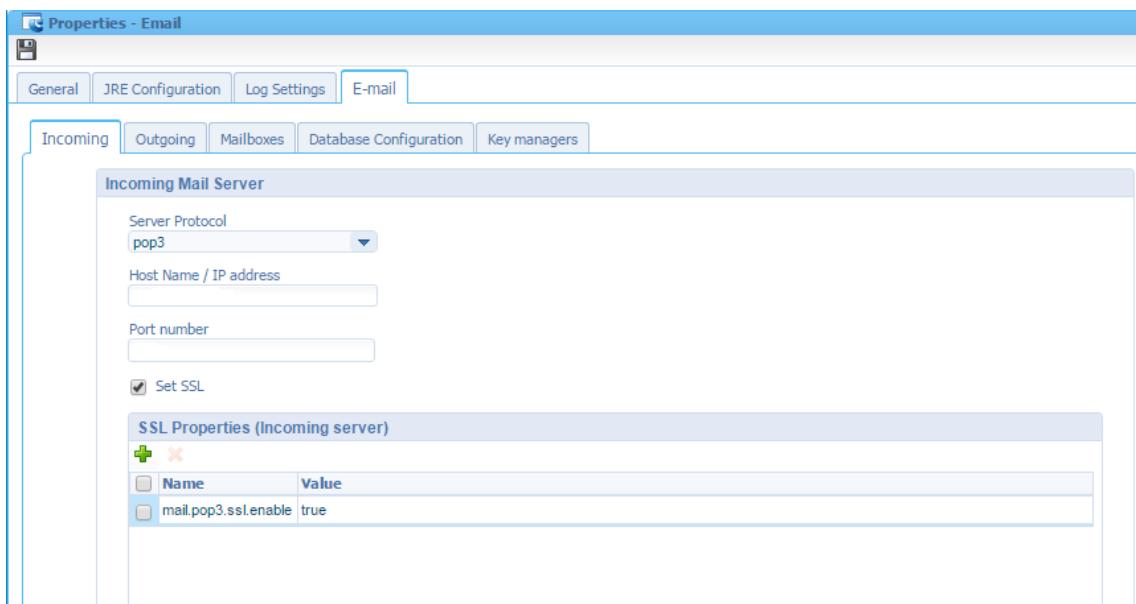
## Configuring the E-mail connector

Configuring the E-mail connector involves the following configurations:

- Inbound
- Outbound
- Mailboxes
- Database
- Key managers

### To configure the E-mail connector:

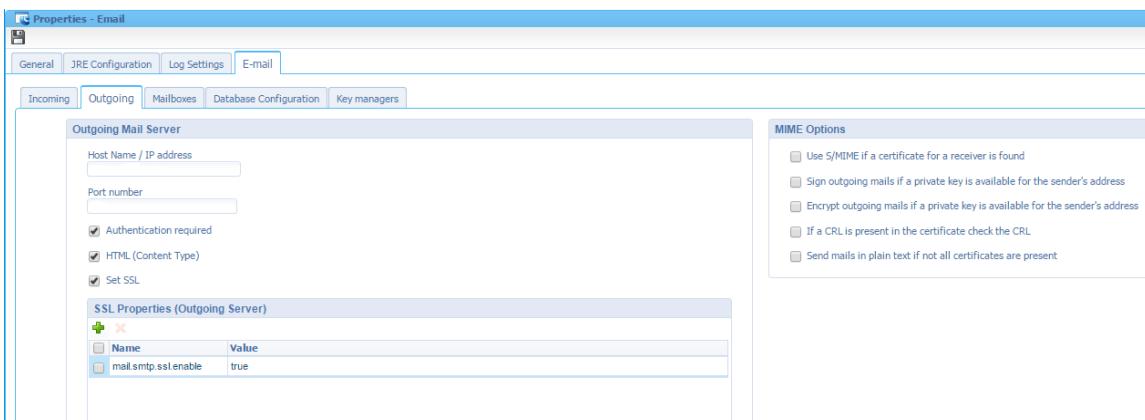
1. On the Welcome page of AppWorks Platform, click  (System Resource Manager). The Email service container is displayed in the Email service group. These are created automatically when AppWorks Platform is installed.
2. In the Service Containers pane, right-click the **E-mail** service container, and then click **Properties**.
3. To automatically restart the container, on the General tab, in the Startup Type list, select **Automatic**.
4. Click the **E-mail** tab.  
A number of tabs display.
5. On the Incoming tab, enter information for the Incoming Mail Server.  
All the values are specific to the email server instance set up in your organization. Contact your IT department for the required values.



Field	Description
Server Protocol	Network protocol to transfer data. Options available are <b>pop3</b> and <b>imap</b> . The POP3 protocol only supports retrieval of email from the default mailbox, INBOX while IMAP supports email retrieval from any mailbox configured for the email account that is specified in the email retrieval request.
Host Name / IP address	IP address of the incoming mail server.
Port	Port number of the incoming mail server.

<b>Field</b>	<b>Description</b>
number	
Set SSL	Whether to use an encrypted connection.
SSL Properties (Incoming Server)	Name and value of the incoming server. The required values are populated when the Set SSL check box is selected. Your IT department will inform you if more SSL properties are required.

6. On the Outgoing tab, enter the following information for the Outgoing Mail Server. All the values are specific to the email server instance set up in your organization. Contact your IT department for the required values.



<b>Field</b>	<b>Description</b>
Host Name / IP address	IP address of the outgoing mail server.
Port number	Port number of the outgoing mail server.
Authentication required	Whether authentication is required.
HTML (Content Type)	Whether HTML content type is set in the outgoing email header.
Set SSL	Whether to use an encrypted connection.
SSL Properties (Outgoing Server)	Name and value of the outgoing server. The required values are populated when the Set SSL check box is selected. Your IT department will inform you if more SSL properties are required.

7. On the Outgoing tab, the following MIME Options are available. These options are not mandatory.

Field	Description
Use S/MIME if a certificate for a receiver is found	Whether S/MIME is enabled.
Sign outgoing mails if a private key is available for the sender's address	Whether to sign an outgoing mail if a private key is available for the sender's address. Usually, this only works for a system email address and not for individual users because their private keys are not accessible.
Encrypt outgoing mails if a private key is available for the sender's address	Whether to encrypt mails. Mails can be encrypted when the public key of a user (which is usually stored in Active Directory) is accessible. Encryption ensures that only that user can read the mail.
If a CRL is present in the certificate to check the CRL	Whether to check if the CRL (Certificate Revocation List) is present for a certificate.
Send mails in plain text if not all certificates are present	Whether to allow sending of non-encrypted mails if S/MIME is enabled.

8. On the Mailboxes tab, enter the following information. This is required for the E-mail connector to perform automatic polling for incoming mail.

Field	Description
Number of poller threads	Number of threads to use when polling for incoming email. Each thread is a separate process. You can increase the value for more emails to be processed concurrently. Note: If you are specifying multiple mailbox configurations, you can increase the number of poller threads. However, a large number of threads can negatively impact the overall performance of your AppWorks Platform installation.
Email box configuration	XML you enter against the schema when the connector is started. The connector validates this XML. Note: If automatic polling for incoming emails is not desired, the default mailbox configuration is valid and need not be modified.

See the sample configuration XML for the Email box configuration.

```
<emailboxes xmlns="http://schemas.cordys.com/1.0/email/configuration">
    <emailbox>
        <name>SolutionStore_Ticketss</name>
        <username>test123@gmail.com</username>
        <password>test123</password>
        <folders>
            <folder>INBOX</folder>
        </folders>
        <triggers>
            <trigger appliesTo="INBOX">
                <name>AllFolders</name>
                <rules>
                    <rule section="SUBJECT">
                        <pattern type="REGEX">
                            <value>.+"</value>
                        <store>
                            <token>
                                <name>Subject</name>
                                <value>0</value>
                            </token>
                        </store>
                    </pattern>
                </rule>
                <rule section="TO">
                    <pattern type="REGEX">
                        <value>.+"</value>
                    <store>
                        <token>
                            <name>To</name>
                            <value>0</value>
                        </token>
                    </store>
                </pattern>
            </rule>
            <rule section="FROM">
                <pattern type="REGEX">
                    <value>.+"</value>
                <store>
                    <token>
                        <name>From</name>
                        <value>0</value>
                    </token>
                </store>
            </pattern>
        </rule>
        <rule section="MULTIPART">
            <pattern type="CUSTOM">
                <value>com.eibus.applicationconnector.email.sample.Base64Plugin</value>
            </pattern>
        </rule>
    </emailbox>
</emailboxes>
```

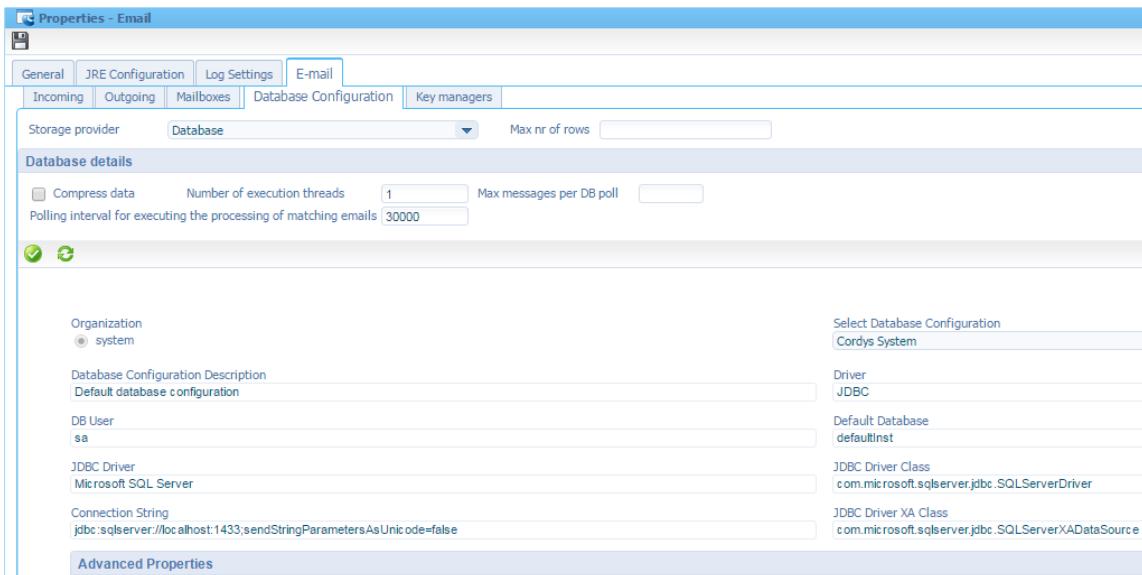
```
<store>
    <token>
        <name>Base64Attachment</name>
        <value>string</value>
    </token>
</store>
</pattern>
</rule>
</rules>
<message>
    <method>receiveEmail</method>
    <namespace>http://schemas.cordys.com/entity/email/1.0</namespace>
    <user>cn=cordys, cn=organizational
users, o=system, cn=cordys, cn=defaultInst, o=vanenburg.com</user>
    <sync>true</sync>
    <namespacemappings>
        <namespacemapping>
            <prefix>ns</prefix>

<namespace>http://schemas.cordys.com/entity/email/1.0</namespace>
            </namespacemapping>
        </namespacemappings>
        <input xmlns:ns="http://schemas.cordys.com/entity/email/1.0">
            <ns:from />
            <ns:subject />
            <ns:body />
        </input>
        <mappings
xmlns:tns="http://schemas.cordys.com/1.0/email/configuration">
            <mapping>
                <source>./ns:from</source>
                <value src="From" />
            </mapping>
            <mapping>
                <source>./ns:subject</source>
                <value src="Subject" />
            </mapping>
            <mapping>
                <source>./ns:body</source>
                <value src="Base64Attachment" />
            </mapping>
        </mappings>
    </message>
</trigger>
</triggers>
<automaticrestart
xmlns:tns="http://schemas.cordys.com/1.0/email">true</automaticrestart>
<restartinprogress
xmlns:tns="http://schemas.cordys.com/1.0/email/configuration">false</restartinprogress>
</emailbox>
</emailboxes>
```

Customize the following parameters in the XML configuration for your email account:

Parameter	Set to
<username>test123@gmail.com</username>	User ID of your email account.
<password>test123</password>	Password of your email account.
<user>cn=cordys,cn=organizational users,o=system,cn=cordys,cn=defaultInst, o=vanenburg.com</user>	User DN for the triggered process specified in the <method> tag. Replace the following: In cn=cordys, replace cordys with the username of an administrator. In o=system, replace system with your organization name. In cn=defaultInst, replace defaultInst with the instance name under which AppWorks Platform is installed on the server. In o=vanenburg.com, replace vanenburg.com with the domain name.

9. On the Database Configuration tab, enter the following information.



Field	Description
Storage provider	Storage provider for the connector. Select <b>Database</b> . <b>Note:</b> To test the connector, select the <b>In Memory (not for production)</b> storage provider because you do not require

Field	Description
	persistence. Ensure that you do not use this storage provider in any production environment because you might lose emails.
Max nr of rows	Maximum number of rows to use in the database.
Compress data	Whether the E-mail connector must zip the data before it is written to the database. This is very useful in limiting the storage.
Number of execution threads	Number of threads to use for the execution engine, that is the number of SOAP requests that are sent in parallel.
Max messages per DB poll	Number of messages to retrieve from the database when the execution engine looks for new work.
Polling interval for executing the processing of matching emails	Interval to poll the database for emails to process.

10. On the Database Configuration tab, perform the required steps based on the database you are using:

When using	Perform the following
Cordys System database	<ul style="list-style-type: none"> <li>a. In the Select Database Configuration list, select <b>Cordys System</b>. All the fields are filled automatically.</li> <li>b. Click <b>Save</b>.</li> <li>c. Select the <b>Restart Service Container</b> check box, and then click <b>Yes</b>.</li> <li>d. Click <b>Refresh</b>. All the service containers are refreshed.</li> </ul>
Another database that you created before beginning the database configuration	<ul style="list-style-type: none"> <li>a. In the Select Database Configuration list, select the database configuration you created before you began configuring the E-mail connector. All the fields are filled automatically. <b>Note:</b> You might have created a database in the previous section, Creating a database configuration.</li> <li>b. Click <b>Save</b>.</li> <li>c. Select the <b>Restart Service Container</b> check box, and then click <b>Yes</b>.</li> </ul>

When using	Perform the following
	<p>d. Click <b>Refresh</b>. All the service containers are refreshed.</p>
Another database that you are yet to create:	<p>If you have not created a database configuration before configuring the E-mail connector, you can create it here.</p> <ol style="list-style-type: none"> <li data-bbox="687 502 1423 608">a. In the Select Database Configuration list, select <b>New Database Configuration</b>. The New Database Configuration window opens.</li> <li data-bbox="687 616 1423 722">b. For Name and Description, enter the name and description of the database configuration. The description is optional.</li> <li data-bbox="687 730 1423 836">c. Enter the relevant details in the corresponding fields. For information about the fields, see <a href="#">JDBC details interface</a>.</li> </ol> <p><b>Note:</b> To create a database, select <b>Create New Database</b> and enter the required details for DBA Name and DBA Password.</p>

Under Advanced Properties, enter the required information related to setting precedence to NULL, supporting special characters in XML, database connections, and cursor cache.

**Note:** When using an Oracle database, you must select the **Support Special Characters in XML** check box.



- On the Key Managers tab, enter the key manager configuration, if necessary:

```
<keymanagers xmlns="http://schemas.cordys.com/1.0/email/configuration" >
</keymanagers>
```

**Note:** The application package contains a key manager called `com.eibus.applicationconnector.email.sample.GenLDAPConnectorKeyManager`. This is the key manager to use with the Generic LDAP connector for certificate lookup.

- Click (Save).

The E-mail connector is configured.

**After you complete:**

During the E-mail connector configuration, if you selected the Authentication required check box on the Outgoing tab, the SetProfile Web service must be invoked at least once prior to executing the GetMail or GetMails (incoming) and SendMail or SendMailMIME (outgoing) SOAP requests to provide the mail account credentials. This does not apply to mail received by the polling mechanism as configured on the Mailboxes tab since the mail account credentials are included in the <emailbox> configuration. It is not expected that an end user needs to invoke the SetProfile Web service. The service that is invoking the GetMail or SendMail SOAP requests must obtain the user credentials and invoke the SetProfile Web service as needed. However, the SetProfile Web service can be invoked with the following steps:

1. On the Welcome page, click **Web Service Interface Explorer**.  
The Web Service Interface Explorer window opens and displays the search options.
2. In Keyword, enter **setprofile**, and then click **Find**.  
The search results are displayed.
3. Select the **SetProfile** Web service operation (tagged with the Email service group and Email implementation type), right-click, and then select **Test**.  
The Operation Test Tool opens and displays the SOAP request.
4. In the SOAP request, replace the following parameters with the values required for the email account to be used for subsequent invocations of GetMail or SendMail:
  - **displayName** (optional)
  - **mailId** (use the email address from which you want to send or receive mail)
  - **password** (use the password for the email address from which you want to send or receive mail)
  - **userId** (use the user ID for the email address from which you want to send or receive mail. This is normally the email address without the '@' and anything after it)
5. Click **Invoke**.

## Configuring the public URL path prefix

In earlier versions of AppWorks Platform (4.1 and earlier), you could access AppWorks Platform using a URL such as <http://myserver/cordys>. The /cordys URL path prefix was mandatory.

BOP 4.2 version onward, the default URL path prefix is /home. You can load AppWorks Platform using a URL such as <http://myserver/home>.

You can configure the URL path prefix /home. This enables you to load AppWorks Platform using a URL such as <http://myserver/processplatform>.

**Note:** The alternative URL path prefix can contain only alphanumeric characters.

The configuration steps for different Web servers are:

- **TomEE**

**To update the TomEE web server configuration:**

1. Navigate to the folder <TomEE\_installdir>/conf/<Engine>/<host> to select the engine and host for installing AppWorks Platform.
2. Rename `home.xml`, for example, `processplatform.xml`.
3. Navigate to the TomEE app base folder <TomEE\_installdir>/webapp.
4. Rename all the WAR files starting with `home#` to the new base URL. For example, rename `home#app#processExperience.war` to `processplatform#app#processExperience.war`.

**Note:** Ensure TomEE is running when renaming the files. This ensures that TomEE automatically updates the folders containing the extracted web applications.

**▪ AppWorks Platform**

In the Management Console, change the AppWorks Platform property  
`set web.server.url.base.folder` to the new URL path prefix.

**Note:** Do not add a slash to the URL. Sample URL:  
`web.server.url.base.folder=processplatform`

Finally, restart the Web server and AppWorks Platform. You can now load AppWorks Platform using the new URL path prefix.

## Defining a custom renderer class to log messages

AppWorks Platform uses the Log4J open source framework to log service container messages. The log messages are customized in a default XML format before being written to the log file.

**To write a custom renderer class to customize a log message:**

1. Write a java class extending with the following Java API. You can access this API in <<Cordys\_HOME>>\components\managementlib\managementlib.jar  
`com.cordys.logger.message.TextRenderer;`
2. Overload the following method with implementation to define custom render logic:  
`@Override public String doRender(String message) { }`

Sample log message, if the occurrences of "\*\*\*\*" are replaced with "\n":

```
import com.cordys.logger.message.TextRenderer; public class
CustomTextRenderer extends TextRenderer { @Override public String doRender
(String message) { return super.doRender(message).replace("****","\n"); } }
```

**Note:** Refer to the [Log4j tutorial](#) for details on configuring the rendering class, layout class, and appenders.

3. Add this class to bcp.classpath in the <>Cordys\_HOME>>\config folder or system classpath.
4. Edit the Log4jConfiguration file located in the <>Cordys\_HOME>>\config folder and register this new class as renderingClass

Sample configuration file with the new class:

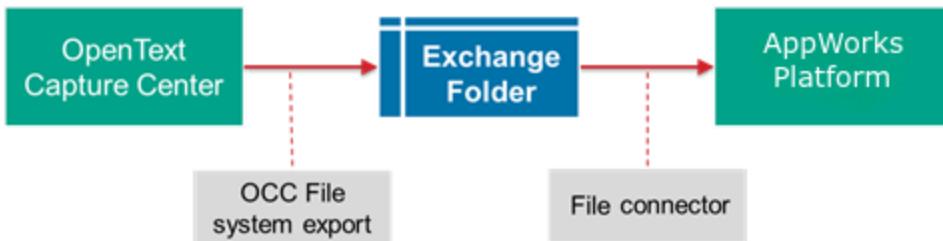
```
<!DOCTYPE log4j:configuration SYSTEM "log4j.dtd"> <log4j:configuration
xmlns:log4j="http://jakarta.apache.org/log4j/"> <renderer
renderedClass="com.eibus.util.logger.internal.LocalizableLogMessage"
renderingClass="com.eibus.util.logger.CustomTextRenderer"/> <renderer
renderedClass="com.eibus.util.logger.internal.LogMessage"
renderingClass="com.eibus.util.logger.CustomTextRenderer"/> <appender
class="com.eibus.util.logger.appenders.ProcessNamedDailyRollingFileAppender"
name="file"> <param name="File" value="%N.log"/> <param name="DatePattern"
value=".yyyy-MM-dd"/> <param name="Encoding" value="UTF-8"/> <layout
class="org.apache.log4j.PatternLayout"> <param name="ConversionPattern"
value="%d{ISO8601} [%p] %c{1} - %m%n"/> </layout> </appender> <appender
class="com.eibus.util.logger.appenders.OSBasedEventAppender"
name="OSBasedEventAppender"> <layout
class="com.cordys.logger.internal.XMLLayout"> <param name="LocationInfo"
value="true"/> </layout> </appender> <appender
class="com.cordys.cap.logger.appenders.ApplicationNameBasedAppender"
name="ApplicationNameBasedAppender"> <layout
class="com.cordys.logger.internal.XMLLayout"> <param name="LocationInfo"
value="true"/> </layout> </appender> <appender
class="com.cordys.logger.appenders.CordysZeroConfJDBCAppender"
name="dbappender"/> <category name="com.eibus.management.AlertSystem">
<priority value="info"/> <appender-ref ref="OSBasedEventAppender"/>
</category> <category name="com.eibus.management.ManagedComponent">
<priority value="info"/> </category> <root> <priority value="error"/>
<appender-ref ref="file"/> <appender-ref ref="dbappender"/> </root>
</log4j:configuration>
```

## Implementing projects combining Capture and AppWorks Platform

Capture and Workflow are a natural fit when it comes to streamlining and automating document-related business processes like claims processing in an insurance company, credit applications in banks, or invoice processing in manufacturing businesses. The capture solution classifies documents arriving from various sources and extracts relevant information, automatically using the most advanced OCR and IDR technologies. The workflow solution handles the business side of the process from approvals, to exception handling, to final posting in a backend system.

OpenText Capture Center (OCC) is the capture solution of OpenText, and AppWorks Platform is the flagship workflow product of OpenText. You can easily connect OCC and AppWorks Platform using an exchange folder, and utilizing standard product functions to read and write

from those folders. The documents are exchanged as searchable PDFs and the data is exchanged in the XML format of OCC.

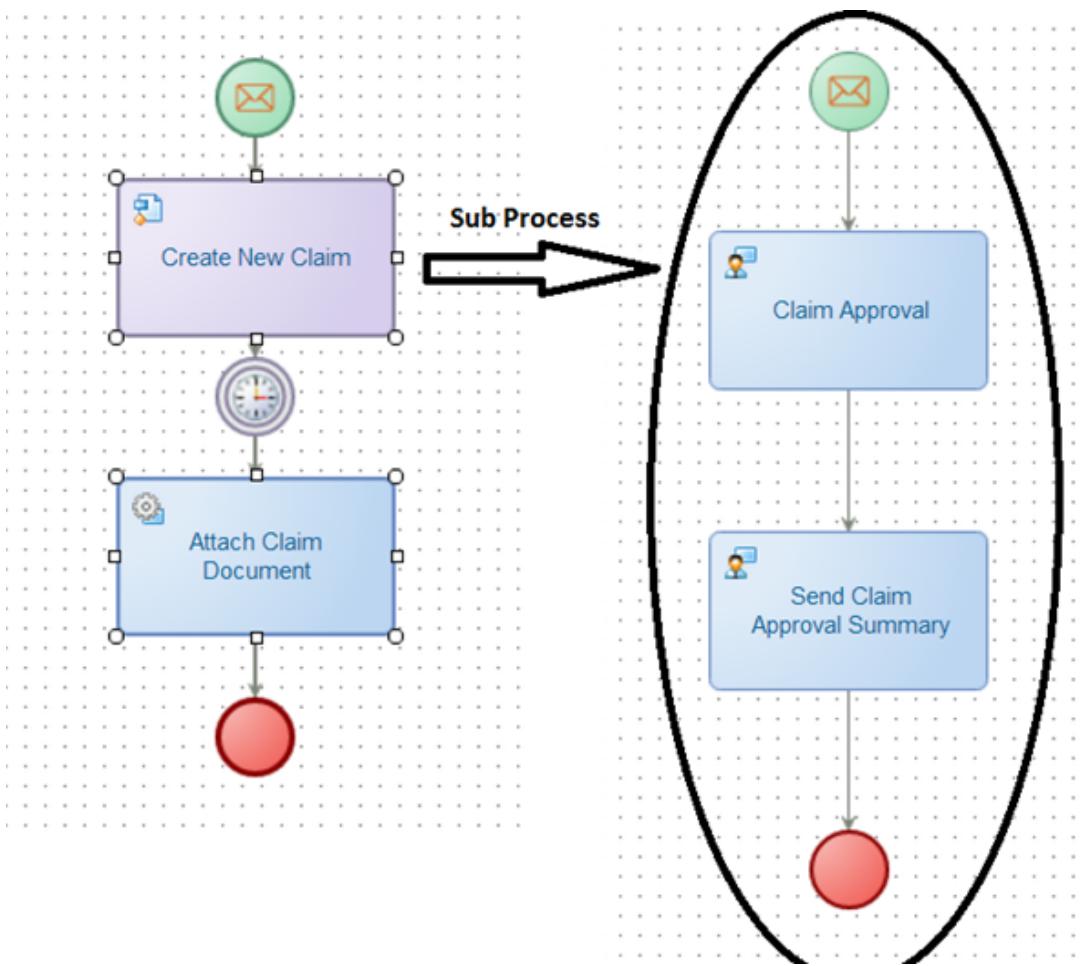


### To create the folder structure for connecting Capture and AppWorks Platform:

1. Define an **Exchange** folder in the OCC folder and create subfolders for each document class that you define in OCC.  
Each document picked up by OCC creates two files with identical names in one of these folders.
2. In OCC, define the **Export Destination** file for each document class pointing to the corresponding folder.
3. In the properties of the **Export Destination** file, set the export folder location.
4. Click **Advanced** and select **XML** and **PDF** as output.

**Note:** For the XML output, select the **One file for each document** option and set the extension as **.xml**.

In AppWorks Platform, define one process for each document class as each document class triggers a different business process. Model the business process workflow and create a Web service to initiate the business process. For example, the following process flow creates a new accident claim, attaches the claim document, and sends the claim for approval. For more information on how to design a business process, see Designing a business process model in the *AppWorks Platform Advanced Development Guide*. For more information on web service generation for the modeled BPM, see Generating a web service operation on a business process model in the *AppWorks Platform Advanced Development Guide*.



Once OCC has placed the data and the document in the designated exchange folder, AppWorks Platform FileConnector picks up the data and triggers the corresponding business process workflow.

**Note:**

- The FileConnector is not installed by default as part of the AppWorks Platform product, but is available as a separate connector. The connector might be available in the [AppWorks Developer community](#). For more information on how to deploy the FileConnector application package, see [Deploying applications](#).
- The poller of the FileConnector Directory that polls files in the configured folders needs a configuration file. The configuration is an XML file which must be stored in the XML Store. The XML Store path for the poller configuration must be provided in the property page of the **FileConnector**.

The following is a sample directory poller configuration with poller location and Web service information:

```

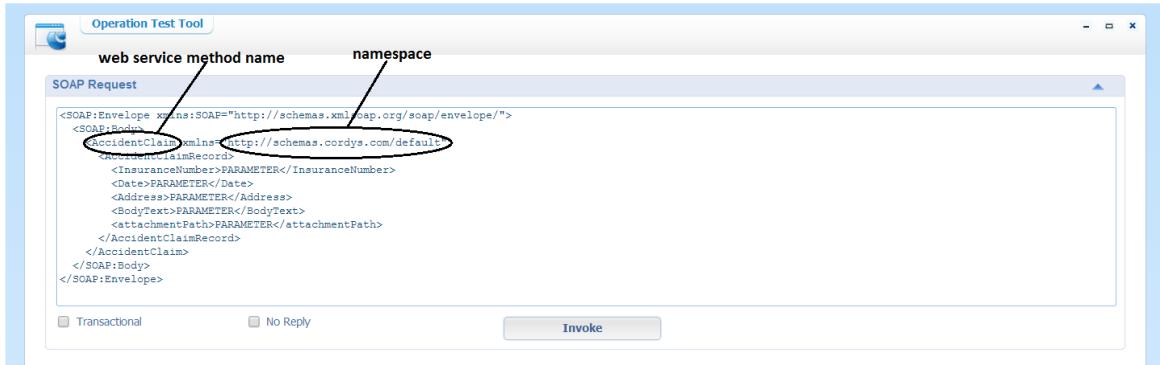
<configuration xmlns:FCDP="http://schemas.cordys.com/coe/FileConnector/Poller/1.0">
  <folder>
    <name>AccidentClaim</name>
    <location>E:\fileconnector\FolderToScan</location> → Location to be Polled
    <track-time>10</track-time>
    <trigger>
      <method>AccidentClaim</method> → Web service method Name
      <namespace>http://schemas.cordys.com/default</namespace>
      <organization>o=system,cn=cordys,cn=soaWip30,o=mydomain.com</organization>
      <user>cn=test,cn=organizational_users,o=system,cn=cordys</user>
      <parameters FCDP:element-data="content-xml"/> → Trigger Parameters
    </trigger>
  </folder>
</configuration>

```

The Exchange folder to be polled by the directory poller, and the corresponding Web service to be triggered are available in the configuration file of the directory poller.

For more information on the Web service method name, namespace, and a sample SOAP request for the generated Web service, see *Testing web service operations* in the *AppWorks Platform Advanced Development Guide*.

The following screenshot highlights the Web service method name and namespace in the Operation Test Tool:



You can place the XML content of the polled file in the SOAP request by adding the attribute `FCDP:element-data="content-xml"` in the trigger parameters or its descendant nodes. All the nodes with this attribute are replaced with the content of the input file and the attribute is removed. When the SOAP request is formed, the content of the SOAP request method node is replaced by the content of the trigger parameters node.

The following is a sample trigger configuration and the SOAP request formed by the Directory Poller corresponding to the input file:

### Trigger configuration

```

<trigger> <method>AccidentClaim</method>
<namespace>http://schemas.cordys.com/default</namespace>
<organization>o=system,cn=cordys,cn=soaWip30,o=mydomain.com</organization>
<user>cn=test,cn=organizational
users,o=system,cn=cordys,o=mydomain.com</user> <parameters>
<type>AccidentClaimRequest</type> <data FCDP:element-data="content-xml" />
</parameters> </trigger>

```

### Input file content

```
<AccidentClaimRecord> <InsuranceNumber>962M09843256PRIVAT</InsuranceNumber>
<Date>09.06.2014</Date> <Address>Address</Address> <BodyText>Claim Request
Info</BodyText> <attachmentPath>C:\OCC\Exchange4Cordys\AccidentReport\II, _
none__0000.pdf</attachmentPath> </AccidentClaimRecord>
```

### SOAP request

```
<AccidentClaimRequest xmlns="http://schemas.cordys.com/default">
<type>AccidentClaimRequest</type> <data> <AccidentClaimRecord>
<InsuranceNumber>962M09843256PRIVAT</InsuranceNumber>
<Date>09.06.2014</Date> <Address>Address</Address> <BodyText>Claim Request
Info</BodyText> <attachmentPath>C:\OCC\Exchange4Cordys\AccidentReport\II, _
none__0000.pdf</attachmentPath> </AccidentClaimRecord> </data>
</AccidentClaimRequest>
```

**Note:**

- You can also change the XML content of the input file before placing it in the SOAP request by adding another attribute `FCDP:xslPath="<xml_store_path_to_an_xsl_file>"`, with the `FCDP:element-data` attribute. The XSL stored in the XML store is used to transform the input file.
- The directory poller can also be configured to poll multiple folders corresponding to each of the OCC document types. The [AppWorks Developer community](#) provides additional resources that you may find helpful.

**Note:** You cannot share the folder to be scanned with multiple FileConnector instances. Therefore, the fail-over and load balancing cannot be configured for FileConnector.

## Providing log configuration details

You must configure the log categories for the security levels after they are registered. See Log Configuration in the *AppWorks Platform API Reference Guide* for more details. You can also specify the log consumers to which log messages can be published.

**To configure the severity level and select log consumers:**

1. Create a service container.
2. In the Provide Log Configuration Details screen, the severity levels for the registered log categories are displayed. Choose one of these options:

Enable System Policy	Enable Logging
Disables all the log categories and severities, since the system-level configuration is applied. On enabling the System Policy option,	Enable the log categories and severities. <ol style="list-style-type: none"> <li>1. Select <b>Root Severity</b> to turn the log option on for all the severity levels, including those that are not listed on the</li> </ol>

Enable System Policy	Enable Logging
<p>Log4jConfiguration.xml is applied to a service container.</p>	<p>screen. The selected severity levels apply to all categories.</p> <p><b>Note:</b> Enabling the lowest severity level automatically enables the remaining severity levels. Similarly, enabling or selecting the highest severity level enables only that severity. For example, selecting the Debug option enables all the options (Debug, Info, Warn, Error, and Fatal). Disabling Fatal disables the remaining severity levels. Selecting severity Fatal does not enable other Severities.</p> <ol style="list-style-type: none"> <li>2. In <b>Category-wise Severities</b>, select the severity for a specific category. <ul style="list-style-type: none"> <li>▪ <b>Root Severity</b> automatically turns the category-wise severity option on for all the categories.</li> <li>▪ After enabling Root Severity, you can clear the severities for any category. This takes precedence over root-level severity.</li> <li>▪ Enabling Root Severity and clearing all options for category-wise severities enables the severities for custom categories that are not listed.</li> </ul> </li> </ol> <p><b>Note:</b> If a severity option is selected in both Root Severities and Category-wise Severities, the latter takes precedence.</p> <ol style="list-style-type: none"> <li>3. Select the log consumer in the Logger Consumers pane. For more information, see <a href="#">Log consumer</a>.</li> </ol> <p><b>Note:</b> If the details are not specified as the above LDAP-based configuration, an advanced user can directly modify and save the configuration details in the AppWorks Platform logger configuration file (&lt;AppWorks Platform_installdir&gt;/config /Log4jConfiguration.xml).</p>

Enable System Policy	Enable Logging
	<p><b>Note:</b> Log4jConfiguration.xml is the standard log file specified through the system policy to log messages when starting the service container. If the service container is configured for logging as specified, during the startup process logging activity switches to the Processor-specific log file.</p> <p><b>Note:</b> Use System Policy for logger configuration when a service container is configured to an OS Process. The log messages for that Service Container are logged to an OS Process-specific log file, that is, &lt;OS Process Name.xml&gt;.</p>

## Enabling logging for gateway

Modify the logger configuration file to enable logging for gateway.

**Note:** If the log configuration details are provided during service container configuration, those are automatically implemented.

The logger configuration takes care of the logging at the gateway level. Modifying configuration details in the configuration file automatically takes care of the changes to the logger at the Gateway level.

### To enable logging for gateway:

1. The default code is:

```
<!DOCTYPE log4j:configuration SYSTEM 'log4j.dtd'> <log4j:configuration
xmlns:log4j="http://jakarta.apache.org/log4j/"> <renderer
renderedClass="com.eibus.util.logger.internal.LocalizableLogMessage"
renderingClass="com.eibus.util.logger.internal.TextRenderer"/> <renderer
renderedClass="com.eibus.util.logger.internal.LogMessage"
renderingClass="com.eibus.util.logger.internal.TextRenderer"/> <appender
class="com.eibus.util.logger.appender.ProcessNamedDailyRollingFileAppender"
name="file"> <param name="File" value="%LN.xml"/> <param name="DatePattern"
value="&apos;.&apos;yyyy-MM-dd"/> <layout
class="org.apache.log4j.xml.XMLLayout"/> </appender> <category
name="Cordys.com.eibus.management.AlertSystem"> <priority value="info"/>
</category> <root> <priority value="error"/> <appender-ref ref="file"/> </root>
</log4j:configuration>
```

2. Add the following lines:

```
<category name='com.eibus.web'> <priority value ='info'/'> </category>
```

This enables the logging for gateway and gateway applications. However, depending on the specific gateway application for which logging must be enabled, you must provide the precise package or category details for that gateway application. For example, to enable logging for wsdl gateway, the category name must be com.eibus.web.tools.wsdl.

**Note:** The priority value can be set to debug, info, warn, error, or fatal.

#### Modified sample code:

```
<!DOCTYPE log4j:configuration SYSTEM 'log4j.dtd'> <log4j:configuration
xmlns:log4j="http://jakarta.apache.org/log4j/"> <renderer
renderedClass="com.eibus.util.logger.internal.LocalizableLogMessage"
renderingClass="com.eibus.util.logger.internal.TextRenderer"/> <renderer
renderedClass="com.eibus.util.logger.internal.LogMessage"
renderingClass="com.eibus.util.logger.internal.TextRenderer"/> <appender
class="com.eibus.util.logger.appenders.ProcessNamedDailyRollingFileAppender"
name="file"> <param name="File" value="%LN.xml"/> <param name="DatePattern"
value="&apos;..&apos;yyyy-MM-dd"/> <layout
class="org.apache.log4j.xml.XMLLayout"/> </appender> <category
name="Cordys.com.eibus.management.AlertSystem"> <priority value="info"/> </category>
<category name="com.eibus.web"> <priority value="info"/> </category> <root>
<priority value="error"/> <appender-ref ref="file"/> </root> </log4j:configuration>
```

## Customizing logging

Custom logs can be added for categories and for classes.

### Customizing categories

These custom categories can be added to the logger configuration:

- **Enabling logging for a package:** The package name must be specified in the category name for which the logging is to be enabled.

```
<category name='packageName'> <priority value='info'/> <appender-ref
ref='appenderReference'/'> </category>
```

- **Enabling logging for a class:** The class name must be specified in the category name for which the logging is to be enabled.

```
<category name='ClassName'> <priority value='warn'/> <appender-ref
ref='appenderReference'/'> </category>
```

### Customizing appenders

Logging for different appenders can be customized. For details, see [Filters for Alert System](#) and [Appenders for an alert system](#).

**Note:** The logger category must be specified as the category name.

## Chapter 43

# Business Activity Monitoring (How Tos)

You can easily perform specific tasks while working with BAM. The examples used in the following topics help you in performing a task.

- [Developing a dashboard](#)
  - [Deciding monitoring content](#)
  - [Identification of data sources](#)
  - [Process monitoring object for attributes to be monitored](#)
  - [Business measures for graphs and KPIs](#)
  - [Drilldown on dashboards](#)
  - [Email action aspects](#)
- [Developing a Business Event Response](#)
- [Developing a KPI](#)
- [Developing an Enterprise Data Object in BAM](#)

## Developing a dashboard

The purpose of a dashboard is to equip you with actionable business information in a format that is intuitive and insightful. Develop a dashboard using the already available Business Measures, KPIs, and the Standard Views to provide a consolidated view on the process performance.

### To develop a dashboard:

1. Decide the content to be monitored. See [Deciding monitoring content](#).
2. Identify the data sources for the content to be monitored. See [Identification of data sources](#).
3. Create the Process Monitoring Object for attributes to be monitored. See [Process monitoring object for attributes to be monitored](#).

**Note:** This step is not relevant for views built on EDO (Enterprise Data Object).

4. Create Business Measures for Graphs and KPIs. See [Business measures for graphs and KPIs](#).
5. Create KPI Monitor for trend analysis. For getting trend graph on KPIs, we can define a KPI Monitor using KPI Editor. KPI Monitor can be scheduled to be run on scheduled time, for example, every month and collect data on rolling basis for each month. From data security perspective, each KPI Monitor is available as separate web service. See [Developing a KPI](#).
6. Create dashboard with drilldown. See [Drilldown on dashboards](#).

A dashboard is developed.

## Deciding monitoring content

### Visualization Views

You can use multiple views to visualize the dashboards. Decide the content to be visualized. Choose from:

- **View 1:** Graph of 'Order count for each customer in the last one month'.
- **View 2:** From View 1, drilldown to the graph 'Order count for different products by the customer in the last one month'.
- **View 3:** From View 2, drilldown to see order instance details like quantity, amount, and status in a tabular view.
- **View 4:** For order instances where the status is **Hold**, see the business processing path graphically to know why the status is **Hold**.
- **View 5:** Key performance indicator (KPI), depicting whether orders with status **Hold** in the last month are in the **Normal** or **Above Normal** range.
- The **Trend Graph** for KPIs for the past year with the KPI in View 5, measured for monthly data.

## Identification of data sources

Data source for the content to be monitored can be a business process, or Web services or enterprise data objects.

Once the high-level data source is identified, for example, **Order Processing** business process, it is useful to find which 'attributes' from that data source can provide data required to be monitored and build views for the dashboard.

In case of business processes, it will be identifying what activities and what attributes from them can return the requisite data.

For instance, to build View 1; graph of **Orders count per customer in last one month**, you may need **Count of order Grouped by customer**, which means **OrderID** and **Customer** attributes must be selected from relevant activities in the **Order Processing** business process.

Similarly, in View 3; viewing Order instances details like quantity, amount, status and so on, in form of a tabular view, only few attributes which are related to process instance is important and not all the attributes of business process.

**Note:** Here term **Attribute** is used for the leaf element nodes available in Message map XML of business process. In case of views developed over enterprise data objects, identify the MDM entities, which are deployed on the BAM hub.

## Process monitoring object for attributes to be monitored

Based on all the attributes identified under Step 2 in the creation of a dashboard topic, you have to create Process Monitoring Object, that is an attributes container object, using Process Monitoring Object Modeler. See Composing dashboards and Creating a process monitoring object in the *AppWorks Platform Advanced Development Guide*.

Process Monitoring Object can contain attributes from a single business process and multiple non-process web services. So, the starting context for designing a Process Monitoring Object is always a single business process and on the process events, attribute values can be retrieved from other AppWorks Platform Web services also.

If the dashboard contains views from multiple business processes, multiple Process Monitoring Objects need to be built. For the large business processes, Process Monitoring Object can also be used for logically grouping different attributes and, thus, multiple Process Monitoring Objects can be built on same business process.

**Note:** There is an example of building a Process Monitoring Object named **OrderProcessingMO** for reference in the following topics.

## Business measures for graphs and KPIs

Each graph and KPI on a dashboard is a business measure Web service developed using the Business Measure designer. Separate Web service per business measure enables better data security by having separate ACL on each Web service. Business measure implementation is a SQL query, built using the Query editor in the Business Measure designer, on a monitored object or Enterprise Data Object.

**Note:** The Business Measure editor does not support nested sub-queries.

For graph view, the SQL query:

```
Select Count(OrderID) CountOrderID from OrderProcessingMO where fromTime >=
:startTime and toTime < :endTime groupby Customer
```

**Note:** Similarly, business measure Web services for graph view and KPI view can be built.

On publish of business measure and KPI, respective composite controls are created for them. These composite controls can be dragged and dropped on the Dashboard designer (XForms). Properties for each composite control can be set and then they can be visualized in the Dashboard (runtime).

## Drilldown on dashboards

Earlier to creating dashboards, you have created Business Measure services for Graphs or KPI views and KPI Monitoring service for trends. You have not defined explicit content for View 3 and View 4 because these views are available by default as **Process Monitoring Object View** and **Process Instance View** composite controls and to define them as drill down for View 1 you have to map them.

**Assigning time frame:** While creating dashboards through User Interface, you can add multiple views to a single dashboard. You can define the time period for the data you want to fetch on the dashboard, per view level. You can define time frame at dashboard level for which you have to wire a single time frame composite control to each view in the user interface. If you want different time frame for each view then you have to drag time frame composite control for each view and map them separately to each view. You can then change the time frame while viewing the dashboard in the runtime.

For example: You can drag and drop View 1, View 2, and View 5 to the Dashboard Designer (XForms) and map only one time frame composite control to each view or separately. Based on this, the data is fetched on the dashboard at runtime.

**Note:** For the first time, data in the dashboard is fetched based on the time frame specified in properties of each composite control. In the dashboard, you can change the time frame through the wired time frame composite control.

**Drill down between business measures:** You can design drill down from a view (graphs) to multiple views (graphs or KPIs). Select business measures composite controls and KPI Monitor composite control and drag them one by one to the Dashboard Designer (XForms). While designing dashboard, you can choose the kind of visualization to view on dashboard for graph or KPIs. You can do this through properties page for each view. For example, for type Graph, you can choose visualization as a bar chart or line chart. You can define different ranges for KPIs.

You can then map two views (Business Measure and KPI) by choosing configure drill down from context menu of Business Measure composite control and then map output parameter of Business Measure to Input parameter of the target view ( KPI). For example, View 2 can be a Target View for View 1. When drill down happens on click of bar or line in graph, 'X-axis' value of graph is passed to Target View and must be mapped to one of the available parameters in Target View. For example, in above drill down scenario, Customer name is passed on from View 1 to View 2.

You can use User Interface's flexibility in arranging the views whichever you want.

**Drill down to Process Monitoring Object Instances:** For wiring the Process Monitoring Object Instances View 3 you first need to open the context menu of Monitoring Object Instances Composite Control and choose **Configure Drill Down** and select the view Business Measure View 1 or KPI View 5 you would like to wire it to. Then you open the context menu of chosen view View 1 or View 5. You right-click a view and choose **Configure Drill Down** option to define a drill down where in you have to map the output parameter of the source view to the input parameter of the target view.

In the dashboard when you click a access point in View 1, you can drill down to Monitoring Object Instances (View 3).

## Email action aspects

When you use email as an 'Action' on Business Event Response or KPI, while exporting or importing BAM contents, consider the following aspects:

- Notification model containing the email template and email method set must be selected explicitly while creating the application(ISVP) as they are not implicitly part of the BAM artifacts.
- For deploying Business Event Response or KPI in multiple organizations, the application containing the email template model can be deployed in the System organization and the email template Web service interface must be explicitly attached to the notification processor existing in the System organization. The HTML format of email can be customized per organization. Business Event Response attributes must not be changed per organization as this is not supported at message template infrastructure level.
- The email template Web service must have already been selected before deploying notification model application, so when Business Event Response or KPI are published per organization then the BAM service does not attempt to attach this method set to the System organization Notification service container.
- If Business Event Response or KPI have to be deployed only for one organization, then for organizational isolation reasons, the notification processor can be configured in the same organization. In this case, email template method set can be selected or changed during publishing of the Business Event Response or KPI and need to be attached to that particular organization notification processor.
- In any deployment scenario, the notification processor must point to the same database as Business Process Management processor because some tables of notification are also synchronized from Business Process Management database to the PIM database. It is assumed that notification and Business Process Management persist data in the same database.
- Packing and deployment works effectively when source and destination are two separate AppWorks Platform system. In the same system, issues can surface when we try to deploy because notification database may be the same system as we use for designing, and primary key violation may occur.

## Developing a Business Event Response

Business Event Response can be defined as a set of conditions on a particular Business process instance and when that happens, actions need to be triggered to intimate or involve relevant stakeholders.

1. Select a business process, on whose instances Business Event Response has to be built. All the published business processes are available for usage in BAM.

2. Decide on the conditions, in the context of business process instance, when 'Corrective or preventive actions' needs to be initiated.

Conditions for the Business Event Response can be either data conditions or temporal (time-based) conditions. Temporal condition can be built on Process or Activity Lead Time. On a Business Event Response, BAM supports building of condition based on multiple data attributes and a single temporal attribute (LEADTIME). As data attributes, elements from Web services can also be used, if these Web services can be invoked in the process instance context, that is, by passing parameters from message map of the process instance.

**For example:**

```
Condition: If (OrderAmount > '100000' && Customer_Type='Gold' && Approval.LEADTIME > 2 day).
```

For temporal conditions on LEADTIME attribute, some pre-defined condition templates are shipped by BAM which help in defining temporal conditions with different unit of measures like hour, day, and so on.

3. Select process events on which attributes needed to build conditions are available. In a business process, different message map elements are available only when corresponding activities are invoked as part of business process execution. Also, all parameters to be passed to invoke enterprise application Web services may not be available at start of business process.  
To make available attributes decided in Step 2, select corresponding process or activity events.
  4. Optional. Select and map Web services, which can fetch attributes needed to build conditions.  
For condition defined under Step 2, Customer\_Type attribute is not available as part of business process message map and need to be fetched from a CRM Web service, which would require Customer\_ID as input parameter. Customer\_ID is available at Process Start and, thus, this Web service can be mapped to invoke at after Process Start event itself.
  5. Optional. Model Email template for actions of type 'email'.  
If email has to be sent as an action for the Business Event Response, then you have to model the email template to serve that purpose. The Email Template Editor is available as part of Business Event Response Editor.
  6. Configure Condition rules. Configure condition rules as decided under Step 2.  
Condition rules can be configured in the Decision Table Editor, which is available as part of Business Event Response Editor.
- Note:** For more details on configuring rules in the Decision Table Editor, see Building a decision table in the *AppWorks Platform Advanced Development Guide*.
7. Configure corrective and preventive actions for Business Event Response. When the Business Event Response conditions are met then multiple actions can be initiated either to notify some people or initiate some corrective or preventive processes.

A Business Event Response is developed.

## Developing a KPI

A KPI helps organizations measure process performance against some pre-defined and custom metrics. A KPI can monitor KPI at scheduled intervals and can trigger actions when KPI crosses certain metric thresholds. Data generated by a KPI can also be used to observe trend on KPI.

1. Identify data sources for the content to be monitored. See [Identification of data sources](#).
2. Create a Process Monitoring Object for attributes to be monitored. This step is not relevant for views built on EDO. See [Process monitoring object for attributes to be monitored](#).
3. Create Business Measures for KPIs. See [Business measures for graphs and KPIs](#).
4. Create a KPI for closed loop monitoring and trend analysis. See Defining a KPI in the *AppWorks Platform Advanced Development Guide*.

**Note:** For closed-loop monitoring, based on business objective defined for the KPI conditions, some actions like Email, Call Web Service, Invoke Business process and so on can be defined. If the specified conditions are met based on schedule run of the KPI, then specified actions get triggered.

The KPI is developed.

### After you complete this task

For getting trend graph on KPIs, define a KPI using KPI Editor. KPI can be scheduled to run every month and collect data on rolling basis for last month. From data security perspective, each KPI is available as separate Web service.

## Developing an Enterprise Data Object in BAM

An Enterprise Data Object (EDO) holds the business data of an enterprise. In AppWorks Platform BAM you can plug-in the EDO and visualize the following:

- Business data in the form of a graph or KPI
- KPI built on the EDO to keep track of any exceptions that happens in the organization and address the same by triggering actions automatically
- Trend on the business data
- Drill-down from the enterprise data to the process level data

The MDM data is synchronized as:

- SPOKE1 (PIM) <--> HUB (MDM) <--> SPOKE2 (BAM)
- In BAM, the MDM synchronization is optimized as SPOKE1 (PIM) -> HUB (MDM + BAM). The HUB and SPOKE2 is combined into one group, HUB (MDM + BAM).
- For EDO, the following model is considered: SPOKE (EDO datastore) -> HUB (MDM + BAM).

### To develop an Enterprise Data Object in BAM:

1. Identify the enterprise application tables which have to be monitored. The enterprise application tables are the source application tables, which MDM detects to retrieve the application data. For example, the 'ORDERS' table of Northwind Database is an enterprise application table.
2. The EDO table that you create in the HUB (BAM) should have the following fields:

Field	Datatype	Value
[RCORGGUID]	[nvarchar](50)	NOT NULL
[RCORDELETED]	[tinyint]	NOT NULL
[RCOROWNER]	[nvarchar](1000)	NOT NULL
[RCORLASTMODIFIED]	[datetime]	NOT NULL
[RCOREXPIRY]	[datetime]	NOT NULL
[RCORLASTMODIFIEDBY]	[nvarchar](50)	NOT NULL
[RCOROPERATION]	[numeric](18, 0)	NOT NULL DEFAULT ((1))

- a. Create the target EDO table schema along with the above specified attributes in HUB (MDM + BAM).
- b. Create a spoke service container of the type MDM Publisher to extract the data from SPOKE (EDO datastore) to HUB (MDM + BAM).  
For more information on creating a service container, see [Creating a service container](#), and for configuring the MDM Publisher Connector Interface, see Fields for Configuring MDM Service Groups in the *AppWorks Platform Advanced Development Guide*.
3. Create a model for the EDO by following the procedure mentioned in Creating the components of an MDM model in the *AppWorks Platform Advanced Development Guide*.
4. Use the created EDO as a data-source in building a Business Measure of BAM and visualize the business data in the form of a Graph or KPI. See Selecting a Data-source in the *AppWorks Platform Advanced Development Guide* for more information on selecting a EDO as data-source in building a Business Measure.

The EDO is developed in BAM.

# Chapter 44

## Extended ECM (How Tos)

Extended Enterprise Content Management (xECM) provides a business context to unstructured data, which is stored in content repositories or Data Management System (DMS), by connecting the structured process and data of enterprise applications and business processes with unstructured content. xECM is a module that must be deployed and configured on Content Server. AppWorks Platform, as of now, uses Enterprise Content Management (ECM) systems as a repository for storing attachments or documents related to business applications. See Extended ECM in the *AppWorks Platform Overview Guide*.

### Business case

Consider a scenario where a customer places an order for a product in the vendor portal.

In this scenario, the business objects are:

- Customer
- Product
- Order

In the following sections, these business objects will be configured in Content Server and AppWorks Platform and the structured data in AppWorks Platform will be meaningfully related to the unstructured content in Content Server.

### Roles involved

Consider the same person is an Administrator for Content Server and AppWorks Platform and both the systems are configured with OTDS.

The following roles have been identified for above scenario:

- Administrator - Performs necessary configuration in Content Server and AppWorks Platform.
- Application developer - Designs the application.
- User - Creates entity instances and uses the application.

## Scenarios

- [Scenarios for business workspace creation](#)
  - Automatic: See [Creating a business workspace automatically](#).
  - Manual: See [Creating a business workspace manually](#).
  - Early: See [Early creation of business workspace](#).
  - Late: See [Late creation of business workspace](#).
- [Defining relations between business objects](#)
  - Parent to Child: See [Parent to Child relation](#).
  - Peer: See [Peer relation](#).
  - Reflexive: See [Reflexive relation](#).
- [Cross application business workspace for multiple business objects](#)
  - Shared: See [Shared business workspace](#).
  - Related: See [Related business workspace](#).
- [Configuring a link to an existing business workspace](#).

## Extended ECM overview

Enterprises and government organizations invest in business applications such as ERP to automate their business processes using AppWorks Platform. These applications excel at managing highly structured transactional information or numerical data, that is generally stored in databases. However, each one of these business applications also generate, process, and store unstructured content such as email, photos, contract documents, invoices, and product specifications. The unstructured data is generally managed by Enterprise Content Management (ECM) systems.

In most of the scenarios, the structured business processes and unstructured content exist in isolation and there is no business context attached to the stored content in the ECM systems. Organizations face a major challenge of linking their business processes with the unstructured content, and gaining a comprehensive overview of the content existing for a business object, such as customer, vendor, or contract. In case of applications developed in AppWorks Platform using classic Business Process Modeling, Case Modeling, or other entities, the content repository stores the files attached to the processes. The files stored in these repositories do not make any business sense for an external user.

Extended ECM (xECM) provides a business context to unstructured data stored in content repositories or Data Management System (DMS). This is done by connecting the structured process, data of enterprise applications, and business processes with unstructured content. This adds business value for the customer and provides a consistent view to a customer to access the business application or DMS. xECM thus enables the management of unstructured content in the context of processes, transactions, and business objects.

## Key concepts of xECM

The following are the key concepts of xECM for AppWorks Platform.

### ***Business object***

A business object represents a meaningful real life entity that is being processed in the application, automatically or manually by users. The examples for business objects are bills of material, orders, or assets in Manufacturing Management. In the Human Resource Management System (HRMS), business objects can be defined for employees, reimbursement request, or applicants. Each module has its own business objects. Business objects consist of structured data, that is data stored in a database. Often, business processes also require unstructured data, such as documents, emails, images, and drawings. These types of information are stored in the Content Server with its document management and archiving, workflow and collaboration capabilities.

### ***Business workspaces***

One of the key concepts of xECM is called business workspace, which contains all the content relevant to a specific business process or a business object such as a customer, vendor, material or product. From the Content Server side, authorized xECM users can view the data of the business object, even if they are not AppWorks Platform users. From the AppWorks Platform application side, users can access Content Server items in the business workspace without leaving the AppWorks Platform application. A business workspace provides a 360° view of the content related to an entity or business object in AppWorks Platform, and provides a full range of ECM capabilities to manage the content.

### ***Business relationships***

xECM also provides a concept called business relationships to reuse the business logic. Using business relationships, you can define the relationship between business objects and make them available inside ECM as relationships between business workspaces. Relationships have a hierarchical structure, known as parents and child.

### ***Business attachments***

Business attachments are links between an item in Content Server, like folder or document, to an entity instance. This means that any existing content available in Content server can be linked to the business object.

## Features of xECM integration with AppWorks Platform

The following features are enabled when xECM is integrated with AppWorks Platform.

### ***Single sign on***

xCM and AppWorks Platform support OTDS-based authentication. You can authenticate against AppWorks Platform and work with xECM without any further authentication.

## **Different ways of workspace creation**

xECM integration with AppWorks Platform supports multiple ways of workspace creation such as synchronized, early, or late.

- Synchronized workspace creation: When a business application creates an entity instance, it automatically creates the workspace instance in Content Server. Based on the configuration, you can disable automatic creation of workspace and enable creation of workspace based on your action.
- Early workspace creation: You can create a workspace in the Content Server before the business object or entity instance is created in the application. When the business object or entity instance is created in the application, you can link the workspace with the created entity instance.
- Late workspace creation: You can create a workspace in the Content Server and link it with an existing entity instance.

## **Represent entity relations as business workspace relations**

All the various types of entity relations defined in the application supported by Entity Modeling are represented as relations between workspaces.

## **Document management**

Document management is the core of xECM for AppWorks Platform applications. It enables you to manage content of all types and formats across the systems. It provides check in and check out, version control, audit trails, comprehensive search and access control. It also includes metadata categorization to enrich content by structured data to create custom properties, control document status, and support content search and retrieval.

## **Cross application support**

Applications developed and deployed using AppWorks Platform can join an existing xECM environment with another leading business application like Salesforce, SAP, or SuccessFactors. In such scenarios, AppWorks Platform becomes a supporting application where it shares an existing workspace created by the leading application. It also can pull data from the workspace created by leading application and store the data in entity instance based on configuration.

## **Content access**

You can access business relevant content using multiple interfaces such as AppWorks Platform UI, ECM UI, email client, and Windows Explorer. You must be authorized to access the content.

## **Business attachment**

This feature enables application users to link existing content, like documents , available in the Content Server with an entity instance. This is useful when documents are stored in a central location and needs to be used by multiple entity instances.

## **Link workspace**

AppWorks Platform can link to an existing workspace created in a content server independently without any business object and, optionally, pull data from this workspace to the entity instance.

## **View documents using Brava viewer**

Documents stored in the workspace can be viewed using Brava! Viewer configured in AppWorks Platform.

## **Configuration report**

xECM integration with AppWorks Platform requires configuration in Content Server and AppWorks Platform. Configuration report feature of xECM integration provides a single view of all configurations related to xECM to administrators, both in AppWorks Platform and in Content Server.

**Note:** To know more about the various scenarios in xECM, see [Extended ECM \(How Tos\)](#).

# **Scenarios for business workspace creation**

Business workspaces can be created through the following scenarios:

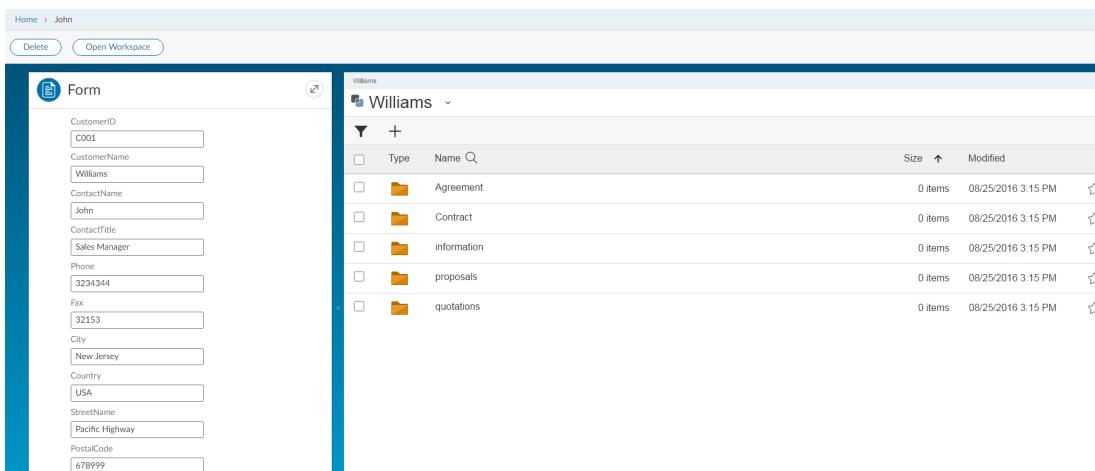
- Automatic creation - The workspace creation in Content Server occurs automatically from AppWorks Platform. See [Creating a business workspace automatically](#).
- Manual creation - The workspace is manually created in Content Server or AppWorks Platform by a user. See [Creating a business workspace manually](#)
- Early creation - The creation of a workspace is enabled even if a related entity is not available in AppWorks Platform. The business workspace is created based on a template; the reference to the entity instance can be added later. This scenario can be used when a business workspace is required to file documents but an entity instance is not yet created. See [Early creation of business workspace](#).
- Late creation - The workspace creation takes place when an entity instance is created first and later during the creation of business workspace in Content Server, the entity instance is attached to it. See [Late creation of business workspace](#).

## **Creating a business workspace automatically**

This scenario automatically creates a business workspace for each entity instance. Each entity instance in AppWorks Platform is related to a business workspace in Content Server, which contains documents and other unstructured information such as images related to the entity instance. A business workspace can have an initial folder structure based on the document template configured for the workspace type.

## To create a business workspace automatically from AppWorks Platform:

1. Develop an application in AppWorks Platform. See [Developing application in AppWorks Platform](#).
  2. Configure AppWorks Platform in Content Server. See [Configuring AppWorks Platform in Content Server](#)
  3. Create and Modify Document Store Repository to configure the Document Store application connector for Content Server in AppWorks Platform. See Creating and modifying Document Store repository in the *AppWorks Platform Advanced Development Guide*.
- Note:** Ensure that the **Synchronize Workspaces Manually** option is not selected in the Document Store application connector configuration page.
4. To create a business workspace automatically:
    - a. Create a customer instance by accessing your application -  
`http://<hostname>/home/<org>/app/start`
    - b. Click **+** and select **Customer** entity.
    - c. Type customer details and create a customer. When the entity instance is created in AppWorks Platform, the business workspace is automatically created in Content Server.
    - d. Open the entity instance.



**Note:** If the automatic creation of business workspace fails, the **Synchronize Workspace Manually** option is enabled for manual synchronization.

- e. You must now be able to perform all the document operations in the right pane which is an instance of the document template in Content Server.
- f. Sign in to Content Server.  
The business workspace must have been created along with the attached documents.
- g. On the Action Bar, click **Open Workspace**.  
The corresponding business workspace opens.

This enables you to access, navigate, and perform all the document management operations in the business workspace, similar to the Content Server user.

## Creating a business workspace manually

### To manually create a business workspace from AppWorks Platform:

1. Develop an application in AppWorks Platform. See [Developing application in AppWorks Platform](#).
2. Configure AppWorks Platform in Content Server. See [Configuring AppWorks Platform in Content Server](#).
3. Create and Modify Document Store Repository to configure the Document Store application connector for Content Server in AppWorks Platform. See Creating and modifying Document Store repository in the *AppWorks Platform Advanced Development Guide*.

**Note:** Ensure that the **Synchronize Workspaces Manually** option is selected in the Document Store application connector configuration page.

4. To create a business workspace manually:
  - a. Create a customer instance by accessing your application -  
`http://<hostname>/home/<org>/app/start`.
  - b. Click **+** and select the **Customer** entity.
  - c. Type customer details and create a customer. The business workspace must not be created automatically in Content Server.
  - d. Open the entity instance.

The **Synchronize Workspace** option must be available.

- e. Click **Synchronize Workspace**.

The business workspace is created in Content Server.

- f. Refresh the page.

The right pane, which is an instance of the document template, is displayed. You can now perform document management operations.

g. Sign in to Content Server to confirm that the business workspace is created with the attached documents.

h. On the Action Bar, click **Open Workspace**.

The corresponding business workspace opens.

Use this to access, navigate, and perform all the document management operations in the business workspace, similar to the Content Server.

## Early creation of business workspace

### Before you begin

1. Develop an application in AppWorks Platform. See [Developing application in AppWorks Platform](#).
2. Deploy the application on to a server.

### Configurations in Content Server

**Note:** You must have Admin privileges in Content Server.

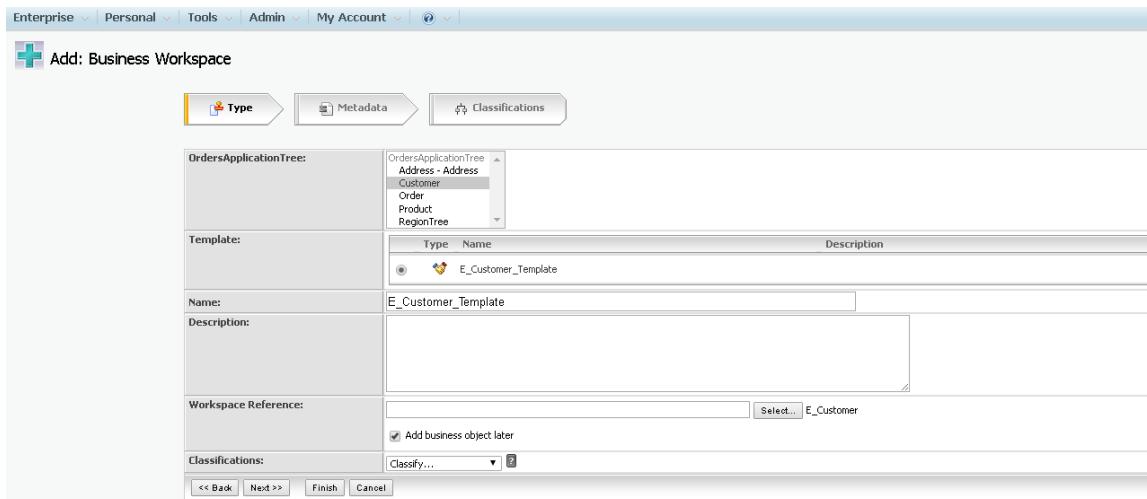
#### To configure Content Server:

1. Configure AppWorks Platform in Content Server. See [Configuring AppWorks Platform in Content Server](#).
2. On the Content Server Administration page, click **Admin > Document Templates Administration >Configure Content Server Document Templates**.
3. Click **Browse** and select the related classification tree for document types.
4. Click **Save Changes**.
5. On the Content Server Administration page, click **Extended ECM > Configure Business Object Types**.
6. Click the business object type created earlier.  
The Configure Business Object Type page opens.
7. Select the following options:
  - **Is Default Display for Workspace Type**
  - **Is Default Search for Workspace Type**
  - **Can be Added as Business Object**
  - **Enable Metadata Mapping from the External System to OpenText Content Server**
8. In **Business Object Name Pattern**, enter a pattern that will be used for the business object name when it is added to a Content Server item. You can use business properties and plain text, for example Document: [OBJTYPE] [DESCRIPT] ([OBJKEY]).
9. Click **Save Changes**.

## **Creating business workspace in Content Server**

### **To create business workspace:**

1. In the Content Server homepage, click **Enterprise > Workspace**.
2. Click **Add Item** and select the **Business Workspace** option.



3. Select a classification from the list of classifications.
4. Type name and description.
5. Select the **Add business object later** check box against **Workspace Reference**.
6. Click **Next** and click **Finish**.  
The following message is displayed: The Business Workspace 'Business Workspace Name' was successfully created.

## **Configuring Document Store application connector in AppWorks Platform**

### **Note:**

- You must have Admin privileges in AppWorks Platform.
- Ensure that the **Synchronize Workspaces Manually** option is selected in the Document Store application connector configuration page.

See Modifying document repository in the *AppWorks Platform Advanced Development Guide* to configure the Document Store application connector for Content Server in AppWorks Platform.

## **Creating entity instances in AppWorks Platform**

### **To create entity instances in AppWorks Platform:**

1. Access your application - `http://<hostname>/home/<org>/app/start`.
2. Click **+** and select a **Customer** entity.
3. Type the customer details and create an entity instance.

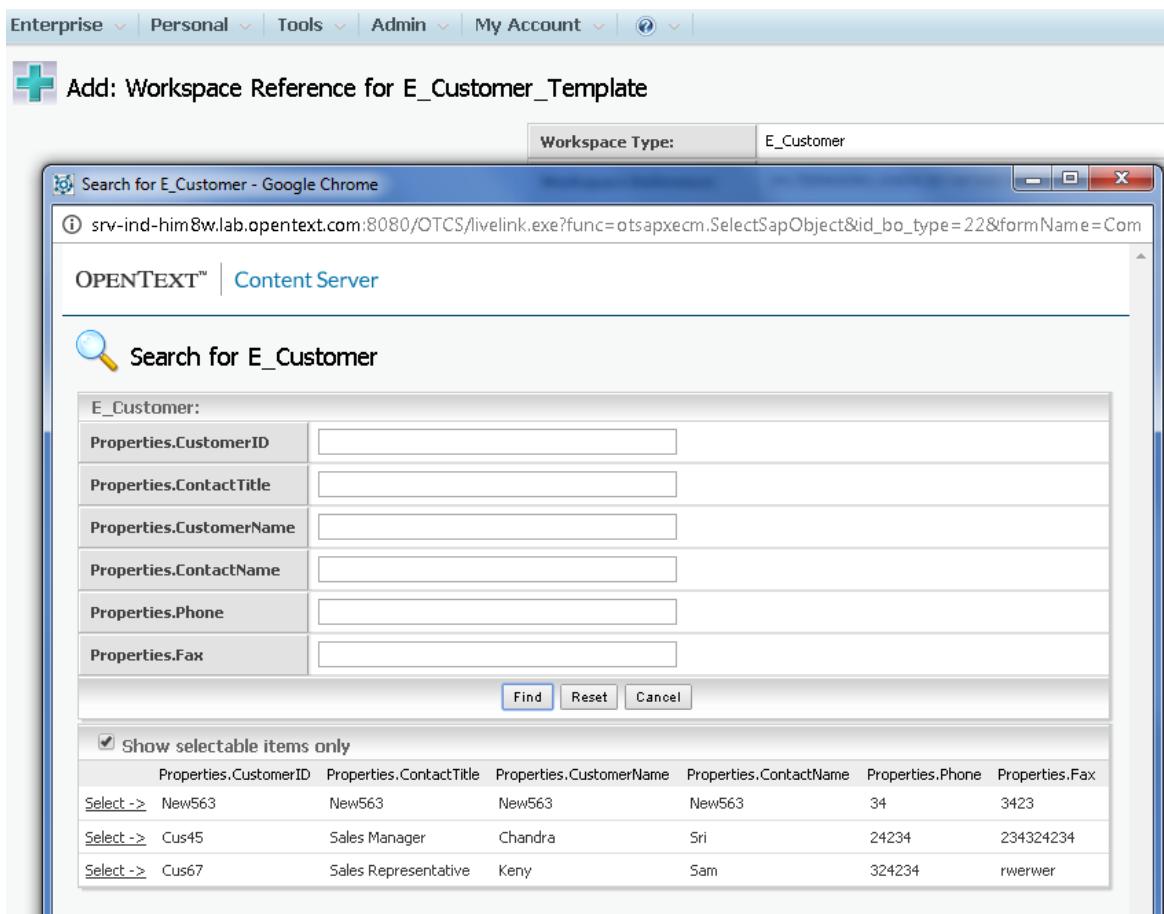
## Attaching entity instances to a business workspace

To attach the entity instance created earlier to a business workspace:

1. In the Content Server homepage, click **Enterprise > Workspace**.
2. Select the business workspace created above from the business workspace folder.
3. Open properties of the business workspace and click **Complete Reference** under **General** tab.

General		Type:
Name:	Cus_89	Size:
Description:		Display:
Created:	10/20/2016 05:18 PM	Modified:
Created By:	Admin	Owned By:
Business Object	Complete Reference	Workspace Type:
Best Bets Value:	<a href="#">Manage Best Bets Items</a>	Best Bets Expiry:
Nickname:	331510	Short Links:

4. Click **Select** against **Workspace Reference**.
5. Click **Find**. The already created entity instances but not yet linked with a business workspace instance are displayed as follows:



6. Select the desired entity instance and click **Add**.
7. Click **Update** to attach the entity instance with the business workspace.
8. Open the application and open the entity instance that was selected earlier. You should be able to view the business workspace that was attached from Content Server.

## Late creation of business workspace

In this scenario, the entity instance is first created in AppWorks Platform and later during the time of business workspace creation in Content Server, the entity instance is attached to it.

### Before you begin

1. Develop an application in AppWorks Platform. See [Developing application in AppWorks Platform](#).
2. Deploy the application on to a server.

### Configurations in Content Server

**Note:** You must have Admin privileges in Content Server.

### To configure Content Server:

1. Configure AppWorks Platform in Content Server. See [Configuring AppWorks Platform in Content Server](#).
2. On the Content Server Administration page, click **Connected Workspaces > Configure Document Templates**.
3. Click **Browse** and select the related classification tree for document types.
4. Click **Save Changes**.
5. On the Content Server Administration page, click **Extended ECM > Configure Business Object Types**.
6. Click the business object type created earlier. The Configure Business Object Type page opens.
7. Select the following options:
  - **Is Default Display for Workspace Type**
  - **Is Default Search for Workspace Type**
  - **Can be Added as Business Object**
  - **Enable Metadata Mapping from the External System to OpenText Content Server**
8. In **Business Object Name Pattern**, enter a pattern that will be used for the business object name when it is added to a Content Server item. You can use business properties and plain text, for example Document: [OBJTYPE] [DESCRIPT] ([OBJKEY]).
9. Click **Save Changes**.

### ***Configuring Document Store application connector in AppWorks Platform***

#### **Note:**

- You must have Admin privileges in AppWorks Platform.
- Ensure that the Synchronize Workspaces Manually option is selected in the Document Store application connector configuration page.

See Modifying document repository in the *AppWorks Platform Advanced Development Guide* to configure the Document Store application connector for Content Server in AppWorks Platform.

### ***Creating entity instances in AppWorks Platform***

#### **To create entity instances in AppWorks Platform:**

1. Access your application - `http://<hostname>/home/<org>/app/start`.
2. Click **+** and select the **Customer** entity. Type the customer details and create an entity instance.

## **Creating business workspace in Content Server and attaching entity instances**

**To create business workspace and attach the above created entity instance:**

1. In the Content Server homepage, click **Enterprise > Workspace**.
2. Click **Add Item** and select the **Business Workspace** option.

3. Select a classification from the list of classifications.
4. Type name and description.
5. Click **Select** against **Workspace Reference**. The mapped business object properties are displayed.
6. Click **Find**. The already created entity instances are displayed.

Properties.CustomerID	Properties.ContactTitle	Properties.CustomerName	Properties.ContactName	Properties.Phone	Properties.Fax
New663	New663	New663	New663	34	3423
Cus87	Sales Representative	Keny	Sam	324234	newerer
Cus87	Sales Manager	Mile	Joy	45435	345345

7. Select the desired entity instance, click **Next** and click **Finish**. The following message is displayed: The Business Workspace 'Business Workspace Name' was successfully created.

The workspace instance is created in the location configured with workspace type.

8. Open the application and open the newly created entity. You should be able to view the business workspace that was attached from Content Server.

## Developing application in AppWorks Platform

### Before you begin

- Install AppWorks Platform16.6.
- You must have a Developer role and have knowledge about CWS.

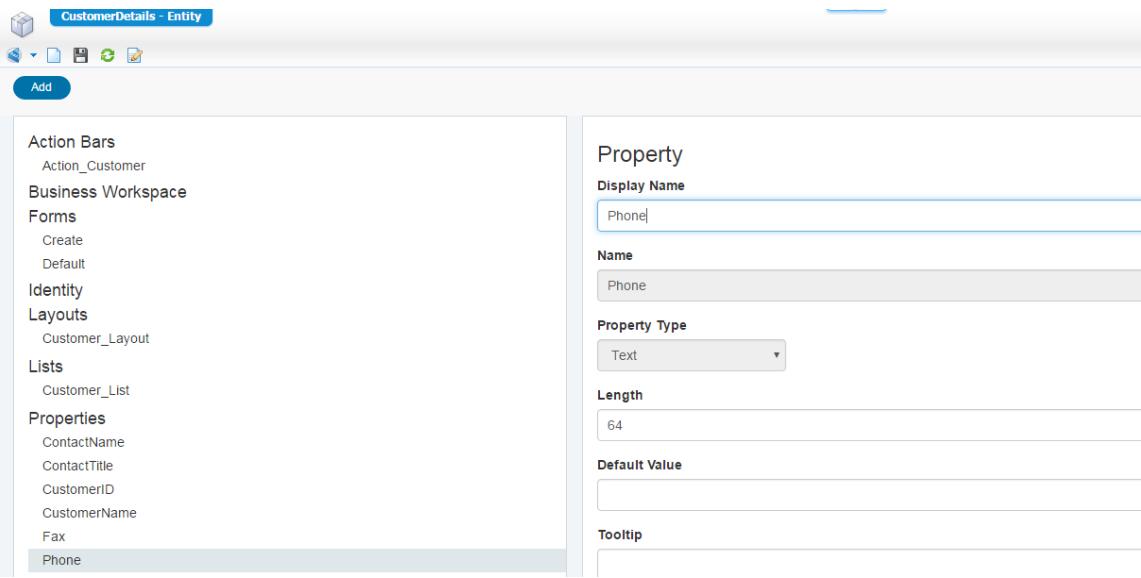
Consider a business case where a customer places orders in the vendor portal for buying products. In this scenario, the following are the business objects:

- Customer
- Product
- Order

Now, create the Customer, Product, and Order entities in AppWorks Platform.

### To configure business objects in AppWorks Platform as entities:

1. Access the AppWorks Platform URL. On the Welcome page click, **Workspace Documents** which is a design time artifact.
2. Create workspace and project.
3. On the CWS explorer, right-click **project > New > Other**.  
The New Document window opens.
4. Search and open the required entity.
5. Click **Add** and select **Property** from **Add building blocks**.
6. Type name, description, and select appropriate **Property Type**. Create properties for Customer as - CustomerID, CustomerName, ContactName, ContactTitle, Phone, Fax



7. Similarly, create Product and Order entities and add the following properties:

Order - OrderID, OrderDate, RequiredDate, ModeOfPayment, ShipName, ShipAddress, ShipCity, ShipRegion, ShipPostalCode, ShipCountry

Product - ProductID, ProductName, SupplierName, Category, UnitPrice, QuantityPerUnit, UnitsInStock, Discontinued

8. Click **Add** and select **Business Workspace** from **Add building blocks**.

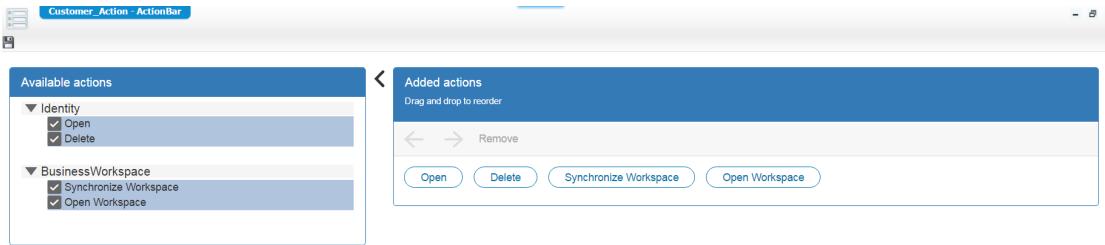
9. Click **Configure** and select the properties for which you want to view metadata in Content Server.

The screenshot shows the 'BusinessWorkspace - Business Workspace' interface. At the top, there's a search icon and a save icon. Below the title bar, the main area is titled 'Available Properties'. It contains three expandable sections: 'Identity' (unchecked), 'Properties' (checked), and 'BusinessWorkspace' (unchecked). Under 'Properties', several checkboxes are checked: ContactName, ContactTitle, CustomerID, CustomerName, Fax, and Phone.

10. Similarly, configure **Business Workspace** building block for the Product and Order entities.
11. Click **Add** and select **Form** from **Add building blocks**.
12. Create the following forms:
  - **Create** - To enter data in runtime.
  - **Default** - To view data in runtime.
13. Click **Configure** and drag properties on to the canvas for which you want to enter values and view data during runtime.
14. Similarly, add forms for the Product and Order entities.
15. Click **Add** and select **Action bars** from **Add building blocks**.
16. Type name, description, and click **Configure**.

17. Select the required actions:

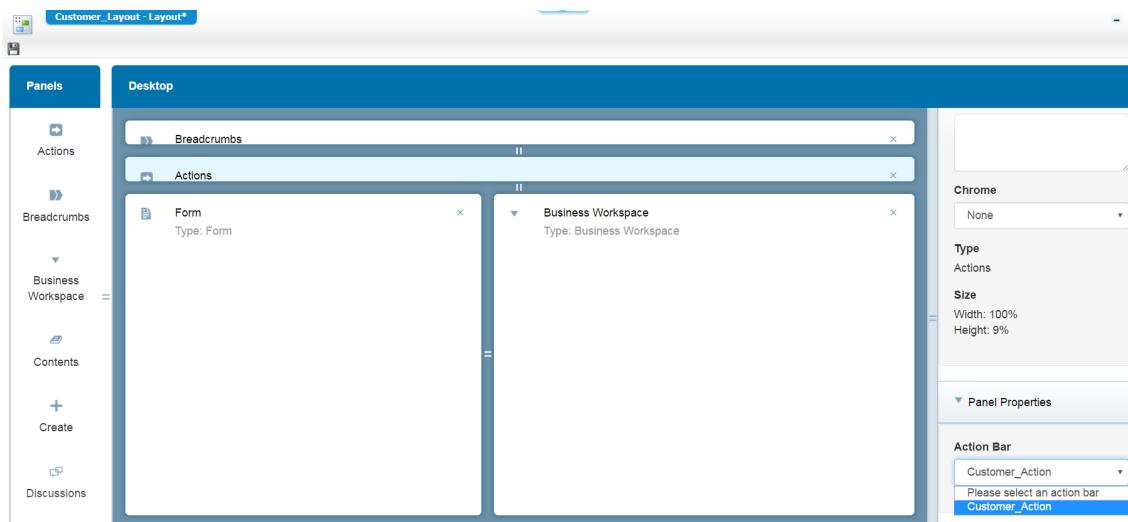
- **Synchronize Workspace** - To manually create or update business workspace during runtime. Configure Action Bar for Customer, Product, and Order.
- **Open Workspace** - To open business workspace from the application.



18. Click **Add** and select **Layouts** from **Add building blocks**.

19. Type name, description, and select **Preview Layout** and **Full Layout**.

20. Click **Configure** and arrange the panels. To perform document management operations, the Business Workspace panel renders the **Folder Browser Widget** of Content Server during runtime.



21. Configure **Layout** for Customer, Product, and Order.

22. Click **Add** and select **Lists** from **Add building blocks**.

23. Type name, click **Configure**, and select the properties that you want to view in List runtime.

24. Configure **List** for Customer, Product, and Order.

25. Save and publish all the entities to view in runtime and to be able to create instances.

**Note:** Configure the security building block for role based access of **Synchronize Workspace** and **Open Workspace** actions in the entities.

26. To work with entity in runtime, you must have an Entity role in the AppWorks Administration page. Access the AppWorks Administration URL - `http://<server>:<port> /home/<organization>/app/admin/web/config#`
  - a. In **Workspaces**, select the project in which the entities are configured.
  - b. In **Configurable Elements > Solution Security > Use Solution** > select the **Entity Runtime User of OpenText Entity Runtime** in Roles.
27. Now access the application URL - `http://<server>:<port> /home/<organization>/app/start`. You should be able to view all the published entities when you click **+**.

## Configuring AppWorks Platform in Content Server

This topic describes the process to configure AppWorks Platform in Content Server.

### Before you begin:

- OTCS is installed on a server. Both OTCS 10.5 and OTCS 16 are supported.
- Ensure that the xECM module is deployed in Content Server. The following configurations are explained using OTCS 16.

## Configuration in AppWorks Platform TomEE

### To access AppWorks Platform as an external system in Content Server:

1. Access `<TomEE_installdir>\conf\tomcat-users.xml`, assign a role to a user, and save it. The user must be an AppWorks Platform user with a developer role.

For example, `<user username="tomee" password="tomee" roles="manager-gui" />`

2. Edit the `<TomEE_installdir>\webapps\home#app#xecm\WEB-INF\web.xml` file.

3. Search for the text `Enter TomEE User Role` and replace with the role specified.

```
<auth-constraint>
<role-name>manager-gui</role-name>
</auth-constraint>
```

4. Restart TomEE.

## Configuration in Content Server

Access the Content Server URL. Click the **Admin** tab > **xECM** module. The following tabs are available:

**Note:** From Content Server 16.2, the navigation path has changed for the configuration components that are under Connected Workspaces.

- **Connected Workspaces**
- **Extended ECM**

### ***Configurations in Connected Workspaces***

On the Content Server Administration page, click **Connected Workspaces**. The configuration steps are available on the right side. Configuration in Connected Workspaces has the following stages:

- [Creating a category](#)
- [Creating a classification](#)
- [Creating a workspace type](#)
- [Configuring document templates](#)
- [Opening a document template](#)

### ***Configurations in xECM***

On the Content Server Administration page, click **xECM**. The configuration steps are displayed on the right side. The configuration in xECM has the following stages:

- [Configuring connections to external systems](#)
- [Configuring business object types](#)

### ***Security parameter configurations***

You must perform a few administrative tasks in Content Server to open the business workspace in the application.

1. On the Content Server Administration page, click **Server Configuration**.  
The configuration steps are available on the right side.
2. Click **Configure Security Parameters**.
3. Add the application server details:

Field	Description
Trusted Referring Websites	<code>http://&lt;machine name:portnumber &gt;</code>
Trusted Cross Domains	<code>http://&lt;machine name:portnumber &gt;</code>
Frame Embedding	Clear the <b>Prevent request handlers from being embedded in external frames</b> option.

**Note:** Ensure that the user is present both in AppWorks Platform and Content Server to be able to open the workspace from the application.

## ***System object volume configurations***

For a cross application to work in AppWorks Platform, configure Content Server as follows:

1. On the Content Server Administration page, navigate to **search Administration > open the System Object Volume > Enterprise Data Source Folder > Enterprise Search Manager > properties > Regions.**
2. In the Regions section, set the properties.
  - a. Select **Enable Queryable** for **XECMWkspLinkRefTypeID**.
  - b. Select **Displayable** for **XECMWkspLinkSAPObjectKey**.
3. Click **Update**.

## ***Configurations in Content Server to link business workspace support in AppWorks Platform***

1. On the Content Server page, navigate to **Tools > Facets volume > Workspace columns > Workspace name en\_us > Function menu > Properties > Workspaces.**
2. Select **Used for Sorting and Filtering**.
3. Click **Update** to save the configuration.

## ***Configurations in Content Server 10.5***

The following configurations are different in Content Server 10.5 when compared to Content Server 16.

**Note:** Except for the following, all other steps explained for Content Server 16 are the same for Content Server 10.5.

In Content Server 10.5:

- All the required configurations can be made under a menu named **ECMLink Administration**.
- While configuring the Business Object Type, you must manually enter the **business object name pattern** which can be obtained from the Workspace type corresponding to the business Object type.
- Only the Classic View is supported.

## ***Creating a category***

Create categories in Content Server bundle attributes and define their type and order. By creating categories, you can add relevant metadata to the business workspaces or documents.

Categories and attributes can be used for the following configurations:

## **Workspace type configuration**

- Define the location of business workspace
- Display information in widgets

## **Business object type configuration**

- Map business properties from the business application to the category attributes.

### **To create a category:**

1. On the Content Server Administration page, click **Connected Workspaces > Open the Categories Volume**.

The Content Server Categories page opens.

**Note:** From Content Server 16.2, navigation path has changed. On the Content Server page, click **Enterprise > Connected Workspaces > Categories**.

The Content Server Categories page opens.

2. On the right side of the page, click **Add Item > Category Folder**. Type name of the new category folder and click **Add**. Use the category folder to group all the categories under one folder
3. Open the created category folder.
4. On the right side of the page, click **Add Item > Category**.
5. Type a name of the new category and click **Add**. For this scenario, create the categories Customer, Order, and Product in the created folder.
6. Open each category and add attributes to it.

- **Customer:** CustomerID, Customer Name, Contact Name, Contact Title, Phone Number, and Fax.

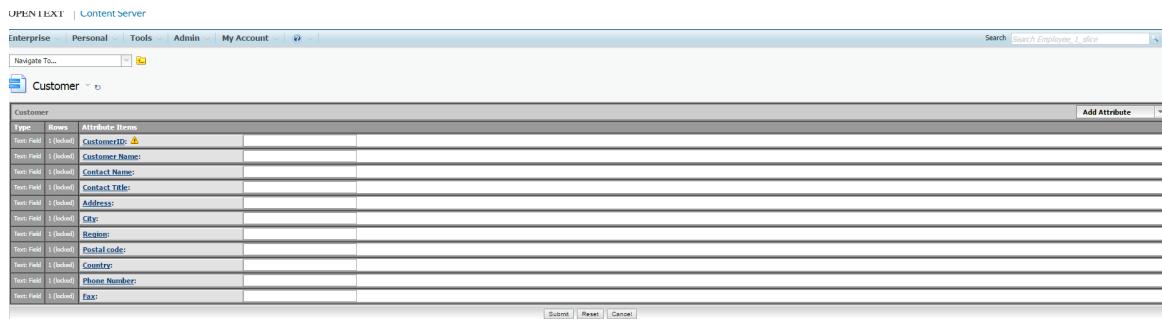
**Note:** All fields are type text.

- **Order:** OrderID, OrderDate (date), RequiredDate (date), ModeOfPayment, ShipName, ShipAddress, ShipCity, ShipRegion, ShipPostalCode, ShipCountry, and DeliveryDate.

**Note:** All fields are type text except date fields.

- **Product:** ProductID, ProductName, SupplierName, Category, Quantity Per Unit, Unit Price, Units In Stock, and Discontinued.

**Note:** All fields are type text.



The supported datatype mapping between AppWorks Platform and Content Server:

<b>AppWorks Platform</b>	<b>Content Server</b>
Boolean Check Box	Flag: Check box
Date	Date: Field
DateTime	Date: Field <b>Note:</b> Select <b>Include Time Field</b> during attribute creation.
Decimal	Text: Field
Duration	Text: Field or Text: MultiLine
Integer	Integer: Field
Float	Text: Field
Enumerated Integer	Integer: Popup
Long Text	Text: MultiLine
Text	Text: Field
Enumerated Text	Text: Popup

## Creating a classification

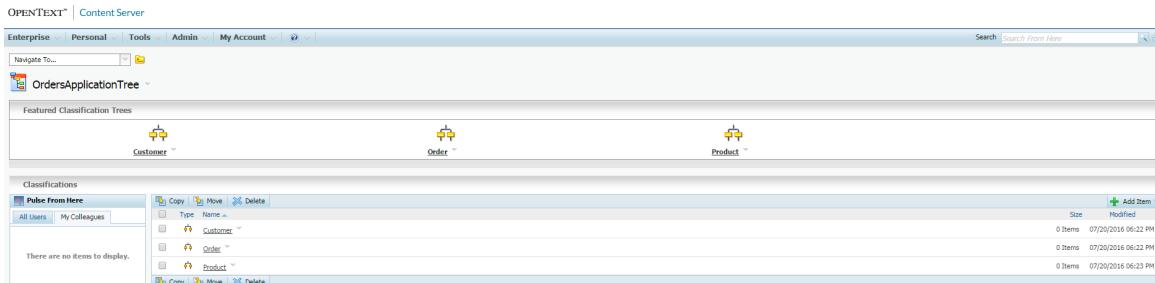
To create business workspaces, you must have a classification tree and classifications used in the document template in the location of business workspaces.

### To create a classification:

1. On the Content Server Administration page, click **Connected Workspaces > Open the Classification Volume**.  
The Classification page opens.

**Note:** The navigation path has changed from Content Server 16.2. On the Content Server page, click **Enterprise > Connected Workspaces > Classifications**.  
The classifications page opens.

2. Open an existing classification tree or create a new one. You can name the classification tree, for example, **OrdersApplicationTree**.
3. Click **Add Item > Classification** and define the new classification according to your requirements.
4. Create a classification for each workspace type. For this scenario, create the Customer, Order, and Product classifications in the classification tree.



## Creating a workspace type

A workspace type provides the framework for the creation of business workspaces. It defines the appearance of business workspaces.

### What you configure in a workspace type

- Location of the business workspace.
- Indexing and search settings.
- Name of the business workspace. It is available in multiple languages.
- Access policies.
- **For Smart View**
  - Name of the workspace type in several languages. The name of the workspace type can be displayed in the header tile of a business workspace.
  - Default icon for business workspaces of this type. Icons can be changed for each individual business workspace.
  - **Perspective Manager:** Configure a business workspace perspective for the workspace type. Perspective Manager is a separate tool.
- **For Classic View**
  - Icon for business workspaces.
  - Population of the business workspace sidebar with sidebar widgets. Sidebar widgets enhance the standard user interface with additional information related to the respective business workspace.

A workspace type is connected to the following:

- One or more business object types that provide the metadata.
- A folder in Content Server where workspaces can be created.

- A classification.

### To create a new workspace type:

1. On the Content Server Administration page, click **Connected Workspaces > Configure Workspace Types**.

The Configure Workspace Types page opens.

**Note:** The navigation path has changed from Content Server 16.2. On the Content Server page, click **Enterprise > Connected Workspaces > Workspace Types**. The Workspace Types page opens.

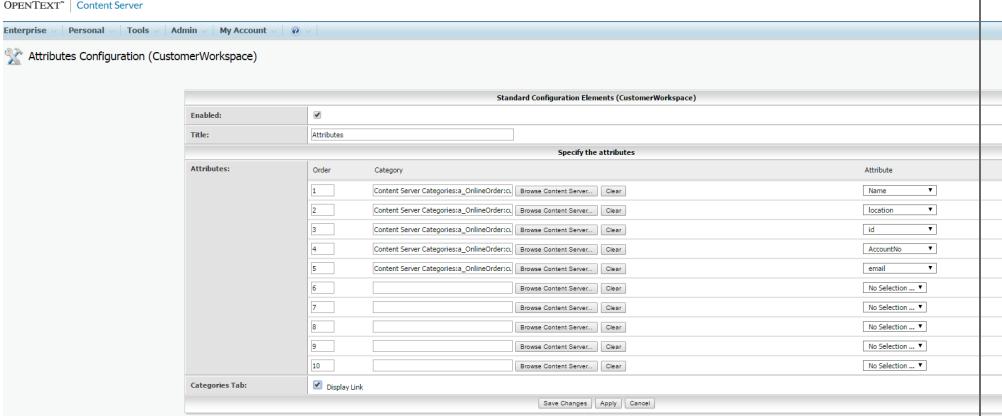
2. Click **+** (**New Workspace Type**).

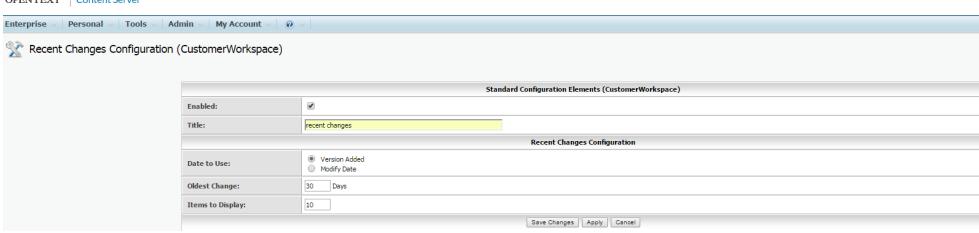
The Create New Workspace Type page opens.

The screenshot shows the 'Change Workspace Type' dialog box with the 'General' tab selected. The workspace name is set to 'E\_Customer'. In the 'Side Bar Widgets' section, several widgets are listed under 'Enabled' and 'Widget' columns, with their 'Title' and 'Vertical' or 'Horizontal' orientation. The 'Vertical' orientation is selected for most widgets. The 'RecentChanges' and 'RelatedOrders' widgets are also present. The 'Workspace Creation Settings' section includes fields for 'Location' (Content Server Folder) and 'Classification' (None). The 'Business Workspace Names' section shows a language selection for 'en\_US (Default)' and a 'Name Pattern' field containing '[100331]Customer Name'. The 'External Document Storage' section includes 'Sub Location Path' and 'RM Classification' fields, both set to 'None'. At the bottom, there are sections for 'Perspective Manager' and 'Widget Configuration for Smart View'.

3. Define the new workspace types for Customer, Order, and Product:

Field	Description
Name	Name of the workspace.
Policies Enabled (SAP integration only)	NA for xECM for AppWorks Platform.
Workspace Icon (for Classic View)	<p>Specify an icon, which is displayed in business workspaces. Click <b>Select Icon</b> to browse the available icons.</p> <p><b>Note:</b> The icon is visible to users in the Classic View on business workspaces and their root folder. For Smart View, use the Widget icon.</p>

Field	Description										
Indexing Settings	Select if the category attributes of the business workspace must become supplementary indexed metadata for child documents. With this option, you can use category attributes in the advanced search to find documents in addition to business workspaces.										
Search Settings	Configure how the search behaves when a user searches from a business workspace that has a related workspace: <ul style="list-style-type: none"> <li>■ Always search in related workspaces.</li> <li>■ Let the users decide if they want to search in related workspaces.</li> <li>■ Disable the search in related workspaces.</li> </ul>										
Side Bar Widgets (for Classic View)	Configure sidebar widgets to display metadata for the business workspace in Classic View. Select a sidebar widget type and configure it. Each sidebar widget type requires different configuration parameters. You can configure each sidebar widget individually or use the same sidebar widget type several times with a different configuration. The following sidebar widget types have been used in this application: <b>Attributes sidebar widget:</b> Displays attributes of the business workspace.  <table border="1"> <thead> <tr> <th>Field</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>Enabled</td> <td>Select this option.</td> </tr> <tr> <td>Widget</td> <td>Select <b>Attributes</b> from the list.</td> </tr> <tr> <td>Title</td> <td>Type a title.</td> </tr> <tr> <td>Actions</td> <td>Click <b>Detailed Configuration</b>. The Attribute Configuration window opens.</td> </tr> </tbody> </table>	Field	Description	Enabled	Select this option.	Widget	Select <b>Attributes</b> from the list.	Title	Type a title.	Actions	Click <b>Detailed Configuration</b> . The Attribute Configuration window opens.
Field	Description										
Enabled	Select this option.										
Widget	Select <b>Attributes</b> from the list.										
Title	Type a title.										
Actions	Click <b>Detailed Configuration</b> . The Attribute Configuration window opens.										

Field	Description						
	<table border="1"> <thead> <tr> <th>Field</th><th>Description</th></tr> </thead> <tbody> <tr> <td>Attributes</td><td>Defines which attributes are displayed. Click <b>Browse Content Server</b> to select the category and select the attribute. The attributes are available based on the selected category.</td></tr> <tr> <td>Categories Tab</td><td>Defines if a link is displayed in the sidebar widget that opens the <b>Categories Tab</b>. All categories and attributes of the business workspace are displayed on the <b>Categories Tab</b>.</td></tr> </tbody> </table>	Field	Description	Attributes	Defines which attributes are displayed. Click <b>Browse Content Server</b> to select the category and select the attribute. The attributes are available based on the selected category.	Categories Tab	Defines if a link is displayed in the sidebar widget that opens the <b>Categories Tab</b> . All categories and attributes of the business workspace are displayed on the <b>Categories Tab</b> .
Field	Description						
Attributes	Defines which attributes are displayed. Click <b>Browse Content Server</b> to select the category and select the attribute. The attributes are available based on the selected category.						
Categories Tab	Defines if a link is displayed in the sidebar widget that opens the <b>Categories Tab</b> . All categories and attributes of the business workspace are displayed on the <b>Categories Tab</b> .						
	<p><b>Recent Changes sidebar widget:</b> Displays a list of items that have been changed.</p> 						
Field	Description						
Enabled	Select this option.						
Widget	Select <b>Recent Changes</b> from the list.						
Title	Type a title.						
Actions	Click <b>Detailed Configuration</b> . The Recent Changes Configuration window opens.						
Date to Use	Select an option. Based on the selected option, items are displayed in the list. You can either use the date when the latest version was added ( <b>Version Added</b> ) or the date when the last modification ( <b>Modify Date</b> ) was done.						
Oldest Change	Modifications older than the defined number of days are not displayed. If you do not enter a						

Field	Description								
	<table border="1"> <thead> <tr> <th data-bbox="458 291 752 333">Field</th><th data-bbox="752 291 1411 333">Description</th></tr> </thead> <tbody> <tr> <td data-bbox="458 333 752 380"></td><td data-bbox="752 333 1411 380">number, all changes are displayed.</td></tr> <tr> <td data-bbox="458 380 752 551">Items to Display</td><td data-bbox="752 380 1411 551">Number of items you want to display in the list. If the number of recent changes is higher than what you have defined here, only the latest changes are displayed.</td></tr> </tbody> </table>	Field	Description		number, all changes are displayed.	Items to Display	Number of items you want to display in the list. If the number of recent changes is higher than what you have defined here, only the latest changes are displayed.		
Field	Description								
	number, all changes are displayed.								
Items to Display	Number of items you want to display in the list. If the number of recent changes is higher than what you have defined here, only the latest changes are displayed.								
	Configure <b>Attributes</b> and <b>Recent Changes</b> widget for customer, Order, and Product.								
Workspace Creation Settings	<p>Define the location where business workspaces are stored, and select the classification that is used for this new business workspace.</p> <p><b>Creating a location for the business workspaces</b></p> <p>In Content Server, create one or more folders where the business workspaces can be created.</p> <p><b>To create and configure the folder:</b></p> <ol style="list-style-type: none"> <li>In Content Server, go to the location where you want to create the root folder for your business workspaces. Click <b>Enterprise &gt; Workspace &gt; Add Item &gt; Folder</b>. The Folder window opens.</li> <li>Create a folder by providing values for the name and description. Create the subfolders: Customer, Order, and Product under the main folder.</li> <li>Enter values for the subfolders:</li> </ol> <table border="1"> <thead> <tr> <th data-bbox="502 1262 959 1305">Field</th><th data-bbox="959 1262 1411 1305">Description</th></tr> </thead> <tbody> <tr> <td data-bbox="502 1305 959 1351">Name</td><td data-bbox="959 1305 1411 1351">Folder name.</td></tr> <tr> <td data-bbox="502 1351 959 1398">Description</td><td data-bbox="959 1351 1411 1398">Description of the folder.</td></tr> <tr> <td data-bbox="502 1398 959 1537">Classifications</td><td data-bbox="959 1398 1411 1537">Corresponding classifications created for Customer, Order, and Product.</td></tr> </tbody> </table> <p><b>Note:</b> Select these classifications while configuring document templates.</p> <ol style="list-style-type: none"> <li>Click <b>Add</b>. The folder is added.</li> </ol> <p><b>Defining the location and classification</b></p>	Field	Description	Name	Folder name.	Description	Description of the folder.	Classifications	Corresponding classifications created for Customer, Order, and Product.
Field	Description								
Name	Folder name.								
Description	Description of the folder.								
Classifications	Corresponding classifications created for Customer, Order, and Product.								

Field	Description
	<ul style="list-style-type: none"> <li>▪ <b>Workspace Creation Settings:</b> Select these created folders, for example, select the customer folder for customer workspace. The documents uploaded for customer from the application are available under the customer folder.</li> <li>▪ <b>Use also for manual creation:</b> Select this option if you want to use the location settings for manual creation of business workspaces on Content Server. Business workspaces are created only in the specified location, regardless of the folder where the user started the creation. After the business workspace is created, the user is directed to the newly created business workspace.</li> </ul>
Classification	Select the classification that was created earlier.
Business Workspace Names	Select an attribute, with which the business workspace must be created. For example, if you select <code>CustomerName</code> attribute, while entity instance is created in AppWorks Platform, business workspaces are created with <code>CustomerName</code> in Content Server. You can also select a combination of attributes for the business workspace names. Configure this field for Customer, Order, and Product.
Widget Configuration for Smart View	<p>For Smart View, you can display an icon that is specific to a workspace type. You can also localize the workspace type name. While the workspace name defines the name of the single business workspace, the workspace type name can be displayed in the header tile of a business workspace. The name is then displayed according to your preferred metadata language.</p> <ul style="list-style-type: none"> <li>▪ <b>Perspective Manager:</b> Start the Perspective Manager by clicking the link <b>Manage Perspectives for this workspace type</b>. It opens with a reduced set of features, which are essential for business workspaces. You can edit an existing perspective or create a new one.</li> <li><b>Note:</b> To configure Perspective Manager, see <i>OpenText Extended ECM Platform - Integration Guide</i>.</li> <li>▪ <b>Workspace Type Name:</b> Add a workspace type name for each language in the workspace type configuration.</li> <li><b>Note:</b> For business workspaces without business object types, you can specify any name. Irrespective of the settings specified in this field, the workspace types always contain the name that is entered during the creation. You can change your preferred metadata language in Content Server by navigating through <b>My Account &gt; Settings &gt; Metadata Language</b>.</li> </ul>

Field	Description
	<ul style="list-style-type: none"><li>■ <b>Widget Icon:</b> The widget icon is displayed in the header tile of business workspaces of this type. To add a workspace type icon, click <b>Browse</b> and select the icon. The recommended format is <code>svg</code>, because pictures of this format are scalable.</li></ul>

## Configuring document templates

This topic describes the procedure to configure document templates in the Content Server.

### To configure document templates:

1. On the Content Server Administration page, click **Connected Workspaces > Configure Document Templates**.  
The Configure Content Server Document Templates page opens.  
**Note:** The navigation path has changed from Content Server 16.2. On the Content Server page, click **Document Templates Administration > Configure Content Server Document Templates**.  
The Configure Content Server Document Templates page opens.
2. In the **Managed object types** section, click **Configure** and select atleast the **Business Workspace (subtype 848)** item.
3. For the **Classification tree for document types**, select an existing classification tree

for business workspaces.

The screenshot shows the 'Configure Content Server Document Templates' interface. At the top, there's a navigation bar with links for Enterprise, Personal, Tools, Admin, and My Account. Below the navigation is a title bar with a gear icon and the text 'Configure Content Server Document Templates'. The main area is divided into two sections: 'Document Types Settings' and 'Wizard Settings'. In the 'Document Types Settings' section, under 'Classification tree for document types', the 'OrdersApplicationTree' is selected. Under 'Default document type', a dropdown menu shows 'RegionTree' as the current selection. The 'Wizard Settings' section contains various configuration options like 'Enable wizard', 'Document template selection', 'Category step', etc. At the bottom, there are 'Action' buttons for Save Changes and Reset.

## Opening a document template

This topic describes the procedure to open document templates in Content Server.

### To open a document template:

1. On the Content Server Administration page, click **open the Document Templates Volume**.

The Content Server Document Templates page opens.

**Note:** From Content Server 16.2, the navigation path has changed. On the Content Server page, click **Document Templates Administration > Open the Content Server Document Templates Volume**. The Content Server Document Templates page opens.

2. Click **Add Item > Select Business Workspace**.
3. Type name, select **Workspace Type**, and **Classifications** created earlier.

4. Add a template for Customer, Product, and Order.
5. Under each template, add the folders in which you want to upload documents. For example, for customer template, upload documents such as agreement, contracts, mails.

## Configuring connections to external systems

This topic describes the procedure to configure connections to external systems in Content Server.

### To configure connections to external systems:

1. On the Content Server Administration page, click **Extended ECM**. The configuration steps become available.
2. Click **Configure Connections to External Systems > Add External Systems**.

3. Enter the application server details.

Field	Description
Logical External System Name	Type a name. This name is used in the <b>External System Id</b> , while configuring document store connector for AppWorks Platform.
Connection Type	Displays the connection type.
Enabled	Enable the configuration.
Comment	Provide more information.
Base URL	Common URL for accessing applications through a Web browser. You can use this base URL when configuring business object types on Content Server. For example: <code>http://&lt;server&gt;:&lt;port&gt;</code>
Application Server Endpoint	URL that is called to obtain business object information. For example, for AppWorks Platform the URL is: <code>http://&lt;server&gt;:&lt;port&gt;/home/&lt;organization&gt;/app/xecm/ECMLinkService?WSDL</code>
Schema Version	Select the interface version.
User Name	User that is used to access business object type information from the external system. <b>Note:</b> The user must be present in AppWorks Platform and TomEE with required roles for accessing the entities. For more information, see <a href="#">Configuring AppWorks Platform in Content Server</a> and the section on security in the <i>Entity Modeling Guide</i> .
Password	Password of the defined user.
Test Connection	Click <b>Test</b> to perform a connection check to the specified external system. After a successful check, the message <code>Connection test to &lt;Logical External System Name&gt; was successful</code> is displayed. The system ID is also retrieved and displayed.

## Configuring business object types

A business object represents a meaningful real life entity that is being processed in the application, automatically or manually by a user. The entities in a business application, which are enabled for the xECM functionality, must be configured and mapped to the business object in xECM.

You must configure a business object type to use in Content Server. Select the business object type from the business application, map properties of the business object to categories and attributes in Content Server, and select a workspace template. You can also enable the business object type for the use of business attachments.

### To configure business object types:

1. On the Content Server Administration page, click **Extended ECM Administration > Configure Business Object Types**.
2. Click **+** to create a New Business Object Type.  
The Configure Business Object Type page opens.
3. Provide the business object type details:

Field	Description
Name	Enter a name for the new business object type.
External System	Select the external system that was created in the previous step.
Business Object Type	<p>Click <b>Select from External System</b>. A window opens displaying the list of entities of the organization configured in the selected external system. Select the appropriate entity from the list.</p> <p><b>Note:</b> For a customized entity, manually enter the element ID of the required customized entity to link to the business object type. You can obtain the element ID from the item URL property, when you add a layout building block. For more information, see the section about item layouts in the designing the presentation topic in the <i>AppWorks Platform Entity Modeling Guide</i>.</p> <p>Content Server may report a failure in validation, but this failure report can be ignored.</p>
Workspace Type	Select the workspace type created for this business object type. For example, for customer select the Customer workspace type.
Is Default Display for Workspace Type	If you have more than one business object type associated with the same workspace type, enable this option to make this business object type the default type to be displayed.
Is Default Search for Workspace Type	Enable this option so that users search the business object type when they create a business workspace manually in Content Server. Whenever you select this option for a business object type, it is removed from other business object types that are linked to the same workspace type.
Display URL	Continue with the default values.

4. Map the properties in the **Property Mapping** section:

Field	Description
Business Properties	<p>Based on the entity selected in the <b>Business Object Type</b>, the field entries in <b>Business Properties</b> are available to retrieve the metadata from AppWorks Platform. Select a business property and select <b>Category Attribute</b> as the <b>Mapping Method</b>. Then click <b>Select</b> against <b>Category and Attribute</b> and select the corresponding category for the entity. Once you select the category, the corresponding attribute is available in the list. Select the corresponding attribute that has to be mapped to the entity property. Hence, AppWorks Platform entity properties gets mapped with the Content Server category attributes.</p> <p><b>Note:</b></p> <ul style="list-style-type: none"> <li>■ Only <b>Category Attribute</b> as the <b>Mapping Method</b> is supported.</li> <li>■ For customised entity specified in the <b>Business Object Type</b>, you must manually enter the field entries in <b>Business Properties</b>. The business properties must be prefixed with building block name delimited by a dot character. For example, <code>Properties.CustomerID</code>.</li> </ul> <p>You can obtain these property names in the desired format when you configure a rule expression on the entity with these properties.</p> <p><b>To obtain a business property:</b></p> <ol style="list-style-type: none"> <li>a. Design a filter in the <b>Basic</b> mode. For more information on using rule expression, see the section about Rule in the Adding Business logic topic in the <i>AppWorks Platform Entity Modeling Guide</i>.</li> <li>b. Switch to the <b>Advanced</b> mode.</li> <li>c. Remove the '<code>item.</code>' prefix from the property in the displayed rule expression. For example, <code>item.Properties.CustomerID</code>.</li> </ol> <p>You then obtain the business property,</p>

Field	Description
	Properties.CustomerID.
Business Property Groups	Currently not supported.
Workspace Template	Select the document template that you was created in the connected workspaces. For example, for customer business object select the Customer workspace template.

The screenshot shows the 'Business Object Type' configuration dialog. Key sections include:

- Name:** CustomerBusinessObject
- External System:** DC\_Customer (HTTP)
- Business Object Type:** 3417EB96909011E6E9C8D7AE56BD5DA9 (selected from a dropdown)
- Workspace Type:** E\_Customer
- Is Default Display for Workspace Type:** (checkbox checked)
- Is Default Search for Workspace Type:** (checkbox checked)
- Display URL:** \$BaseUrl\$/sap/bc/gui/sap/its/webgui?>logingroup=SPACES->transact=
- Adding of Business Objects** section: Can be Added as Business Object: Yes
- Automatic Adding of Business Objects** section: Trigger Automatic Creation by: (dropdown menu)
- Retrieve Business Object Key from:** Key Fields of 3417EB96909011E6E9C8D7AE56BD5DA9: Category (Identity.ItemId) and Attribute (Identity.ItemId)
- Property Mapping** section: Mappings for Business Properties (CustomerID, CustomerName, ContactTitle, Identity.Id, Fax, ContactName) and Business Property Groups.
- Workspace Template** section: Workspace Template: Content Server Document Templates:E\_Customer\_Template

- Save the Business Object Type.
- Configure Business Object Types for the Customer, Order, and Product business objects. The [AppWorks Developer community](#) provides additional resources that you may find helpful.

## Defining relations between business objects

You can map AppWorks Platform entity relations with the business workspace in Content Server.

## Entity relations

The relations that can be defined in AppWorks Platform:

- [Parent to Child relation](#)
  - One to Many
- [Peer relation](#)
  - One to One
  - One to Many
  - Many to Many
- [Reflexive relation](#)

## Parent to Child relation

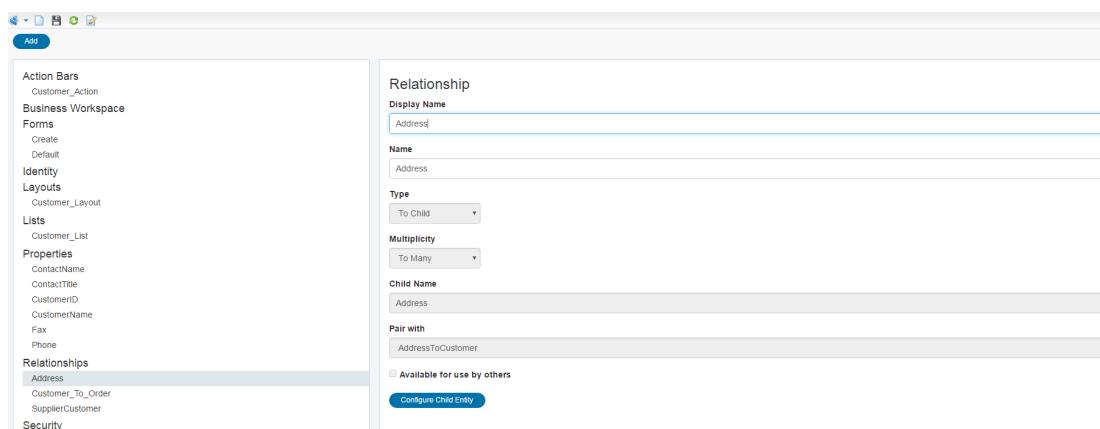
### Before you begin:

1. Develop an application in AppWorks Platform. See [Developing application in AppWorks Platform](#)
2. Extend the application to define relations by performing the steps mentioned in the following sections.

Consider Customer as a parent entity and Address as a child entity. A single customer can have multiple addresses.

### To establish Parent to Child relation:

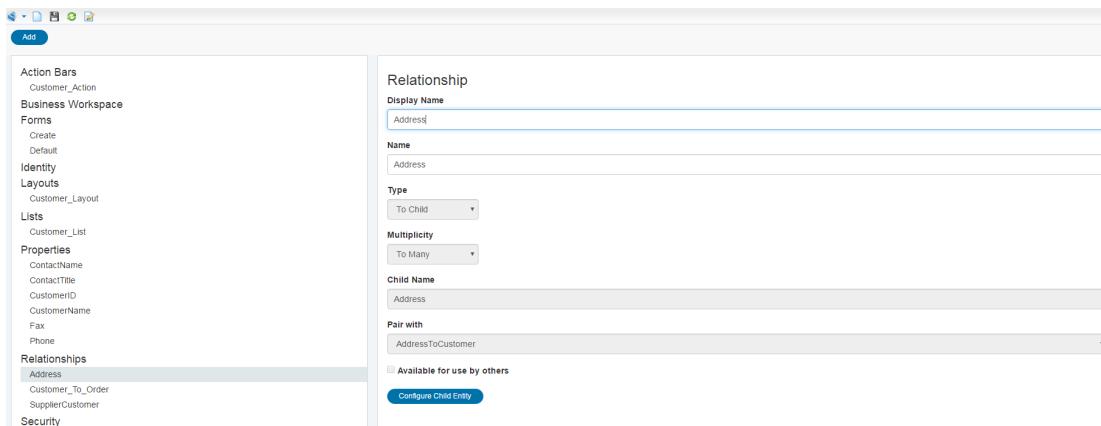
1. Define the relationship between parent and child entities:
  - a. Open the Customer entity in the CWS workspace.
  - b. Click **Add > Relationship**. The Relationship pane opens.



- c. Enter the following values:

Field	Description
Display Name	Type a display name.
Name	Type a name.
Type	Select To Child option.
Multiplicity	Select To Many option.
Child Name	Displays the value provided in the Name field.

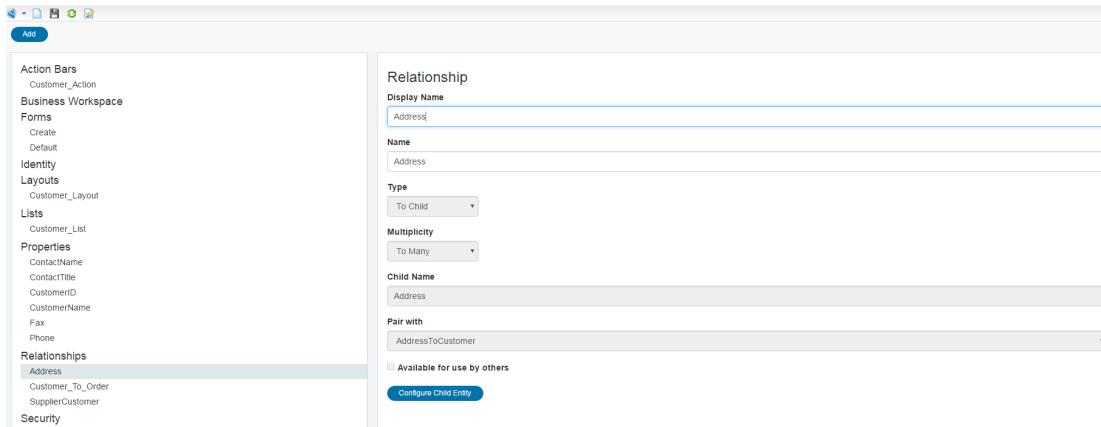
- d. Click **Save** to save the relationship.
2. Define the relationship between parent and child entities:
- Open the Customer entity in the CWS workspace.
  - Click **Add > Relationship**. The Relationship pane opens.



- c. Enter the values:

Field	Description
Display Name	Type a display name.
Name	Type a name.
Type	Select To Child option.
Multiplicity	Select To Many option.
Child Name	Displays the value provided in the Name field.

- d. Click **Save** to save the relationship.
3. Define the relationship between parent and child entities:
- Open the Customer entity in the CWS workspace.
  - Click **Add > Relationship**. The Relationship pane opens.



c. Enter the values:

Field	Description
Display Name	Type a display name.
Name	Type a name.
Type	Select To Child option.
Multiplicity	Select To Many option.
Child Name	Displays the value provided in the Name field.

d. Click to save the relationship.

4. Click **Configure Child Entity**.
5. Click **Add** and select **Property** from **Add building blocks**. Define the following properties in the child entity: City, Country, StreetName, PostalCode, and State.
6. Click **Add** and select **Form** from **Add building blocks**.
7. Click **Add** and select **Business Workspace** from **Add building blocks**.
8. Click **Configure** and select the properties for which you want to view metadata in Content Server.
9. Create the following forms:
  - a. **Create**: To enter data in runtime.
  - b. **Default**: To view data in runtime.
10. Create the following forms:
  - a. **Create** - To enter data in runtime.
  - b. **Default** - To view data in runtime.
11. Create the following forms:
  - **Create** - To enter data in runtime.
  - **Default** - To view data in runtime.

12. Click **Configure** and drag properties on to the canvas for which you want to enter values and view data during runtime.
13. Click **Add** and select **Layouts** from **Add building blocks**.
14. Type name, description, and select **Preview Layout** and **Full Layout**.
15. Click **Configure** and arrange the panels.  
The Business Workspace panel, renders the **Folder Browser Widget** of Content Server during runtime to perform document management operations.
16. Define **Relationships**:

Field	Description
Display Name	Type a display name.
Name	Type a name.
Type	Select <b>To Parent</b> option.
Multiplicity	Select <b>To One</b> option.
Target Entity	Select the Parent entity.

17. Click **Save** to save the relationship.  
The parent child relation is established.
18. Click **Save** to save the relationship. The parent child relation is established.
19. Click  to save the relationship. The parent child relation is established.
20. In **Pair with**, select the child relationship created earlier.
21. On the Customer entity, click **Create form > Configure**, drag the child relationship to the presentation area. Perform the same action for the **Default form**. You can edit the value.
22. Click  to save the entity.
23. Publish the parent entity.  
In the application, you can view and open the embedded child entity inside the parent entity.

## Configuring in Content Server

### To configure Content Server:

1. Configure Address child entity in Content Server. For more details, see [Configuring AppWorks Platform in Content Server](#).
2. In workspace type,
  - For classic view, configure **Side Bar Widgets**. Configure **Related Items** as follows:

Field	Description
Enabled	Select this check box.
Widget	Select <b>Related Items</b> from the list.
Title	Type a title.
Action	<ol style="list-style-type: none"> <li>Click <b>Detailed Configuration</b>. The <b>Related Items Configuration</b> window opens.</li> <li>Select <b>Show Parent Relationships</b> check box.</li> <li>Click <b>Save Changes</b>.</li> </ol>

- For smart view, configure **Perspective Manager**. To configure Perspective Manager, see *OpenText Extended ECM Platform - Integration Guide*.
3. In **Workspace Creation Settings**, click **Select** and select the parent location.
  4. In **Business Workspace Names**, click **Insert Attribute** and select a child attribute, for example, City.
  5. Save the changes.

### **Configuring Document Store application connector in AppWorks Platform**

**Note:** You must have admin privileges in AppWorks Platform.

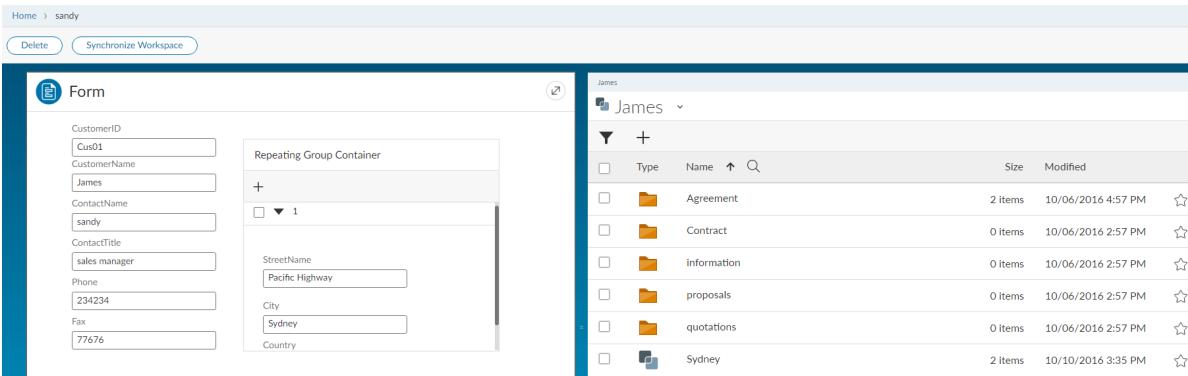
#### **To configure document store application connector in AppWorks Platform:**

1. Create Document Store application connector. See [Creating a service group](#).
2. Configure the Document Store application connector for Content Server in AppWorks Platform. See [Document Store Configuration interface](#).

### **Creating business workspace from the application**

1. Create a Customer instance by accessing the application URL  
`http://<server>:<port>/home/<organization>/app/`
2. Click **+** and select the **Customer** entity.
3. Enter the entity details and click **Create**.
4. Click the entity instance to open it.
5. Enter values for the child entity and click **Synchronize Workspace**. The child entity **City** is created.

The following screenshot shows the child entity **Sydney** created in the parent business workspace **James**.



## Peer relation

### Before you begin:

1. Develop an application in AppWorks Platform. See [Developing application in AppWorks Platform](#)
2. Extend the application to define relations by performing the steps mentioned in the following sections.

### **Peer - One to One and One to Many relations**

Consider Customer and Order, one customer can place multiple orders and one order can be placed by one customer. From Customer to Order, it is One to Many relation. From Order to Customer, it is One to One relation.

#### To establish One to One and One to Many relations:

1. Open the Customer entity in the CWS workspace.
2. Click **Add > Relationship**.  
The Relationship pane opens.
3. Enter the values:

Field	Description
Display Name	Type a display name.
Name	Type a name.
Type	Select <b>To Peer</b> option.
Multiplicity	Select <b>To Many</b> option.
Target Entity	Select the Order entity.
Pair with	Automatically filled with the relationship defined in the Order entity.

4. Click to save the entity.

5. Open the Order entity in the CWS workspace.
6. Click **Add > Relationship**.  
The Relationship pane opens.
7. Enter the values:

Field	Description
Display Name	Type a display name.
Name	Type a name.
Type	Select <b>To Peer</b> option.
Multiplicity	Select <b>To One</b> option.
Target Entity	Select the Customer entity.
Pair with	Select the relationship defined in Customer.

8. On the Order entity, click **Create form > Configure**, drag the above defined relationship to the presentation area. Perform the same action for the **Default form**. You can edit the value.
9. Click  to save the entity.
10. Publish the Customer and Order entities.  
In the application, you can search for the existing customers in the Order entity for placing an order.

### Configurations in Content Server

1. On the Content Server Administration page, click **Connected Workspaces > Configure Workspace Types**.
2. In workspace type,
  - For classic view, configure **Side Bar Widgets**. Configure **Related Items** as follows:

Field	Description
Enabled	Select this check box.
Widget	Select <b>Related Items</b> from the list.
Title	Type a title.
Action	<ol style="list-style-type: none"> <li>a. Click <b>Detailed Configuration</b>. The <b>Related Items Configuration</b> window opens.</li> <li>b. Select <b>Show Child Relationships</b> check box.</li> </ol>

Field	Description
	c. Click <b>Save Changes</b> .

- For smart view, configure **Perspective Manager**. To configure Perspective Manager, see *OpenText Extended ECM Platform - Integration Guide*.

### Configuring Document Store application connector in AppWorks Platform

**Note:** You must have admin privileges in AppWorks Platform.

#### To configure document store application connector in AppWorks Platform:

- Create Document Store application connector. See [Creating a service group](#).
- Configure the Document Store application connector for Content Server in AppWorks Platform. See [Document Store Configuration interface](#)

### Creating business workspace from the application

- Create a business workspace from the application.
- Create a customer instance by accessing the application URL -  
`http://<server>:<port>/home/<organization>/app/`
- Click **+** and select the Order entity.
- Enter the entity details, search for the customer and select the relevant customer.
- Click **Create**, an order instance gets created.
- Click the entity instance to open it and click **Synchronize Workspace**.

In the Content Server you should be able to view the relations, for example, which customer has placed what orders.

### Peer - Many to Many relation

Consider Customer and Product. Many customers can buy many products.

#### To establish Many to Many relation:

- Open the Customer entity in the CWS workspace.
- Click **Add > Relationship**.  
The Relationship pane opens.
- Enter the values:

Field	Description
Display Name	Type a display name.
Name	Type a name.

Field	Description
Type	Select <b>To Peer</b> option.
Multiplicity	Select <b>To Many</b> option.
Target Entity	Select the Product Entity.
Pair with	Automatically populated with the relationship defined in the Product entity.

4. On the Customer entity, click **Create form > Configure**, drag the above defined relationship to the presentation area. Perform the same action for the **Default form**. You can edit the value.
5. Click **Save** to save the Customer entity.
6. Open the Product entity. Click **Add > Relationship**. The Relationship pane opens.
7. Enter the values:

Field	Description
Display Name	Type a display name.
Name	Type a name.
Type	Select <b>To Peer</b> option.
Multiplicity	Select <b>To Many</b> option.
Target Entity	Select the Customer Entity.
Pair with	Select the relationship defined in Customer.

8. On the Product entity, click **Create form > Configure**, drag the above defined relationship to the presentation area. Perform the same action for the **Default form**. You can edit the value.
9. Click **Save** to save the Product entity.
10. Publish the Customer and Product entities.

## Configuration in Content Server

### To configure Content Server:

1. On the Content Server Administration page, click **Connected Workspaces > Configure Workspace Types**.
2. In workspace type,
  - For classic view, configure **Side Bar Widgets**. Configure **Related Items** as follows:

Field	Description
Enabled	Select this check box.
Widget	Select <b>Related Items</b> from the list.
Title	Type a title.
Action	<p>a. Click <b>Detailed Configuration</b>. The Related Items Configuration window opens.</p> <p>b. Select the <b>Show Child Relationships</b> check box.</p> <p>c. Select <b>Tree</b> in Display Style.</p> <p>d. Click <b>Save Changes</b>.</p> <p>e. Click Detailed Configuration. The Related Items Configuration window opens.</p> <p>f. Select Show Child Relationships check box.</p> <p>g. Select Tree in Display Style.</p> <p>h. Click Save Changes.</p> <p>i. Click <b>Detailed Configuration</b>. The Related Items Configuration window opens.</p> <p>j. Select <b>Show Child Relationships</b> check box.</p> <p>k. In <b>Display Style</b>, select <b>Tree</b>.</p> <p>l. Click <b>Save Changes</b>.</p>

- For smart view, configure **Perspective Manager**. To configure Perspective Manager, see *OpenText Extended ECM Platform - Integration Guide*.

### Configuring Document Store application connector in AppWorks Platform

**Note:** You must have admin privileges in AppWorks Platform.

#### To configure document store application connector in AppWorks Platform:

- Create Document Store application connector. See [Creating a service group](#).
- Configure the Document Store application connector for Content Server in AppWorks Platform. See [Document Store Configuration interface](#)

### Creating business workspace from the application

- Create business workspaces from the application.
- Create a customer instance by accessing the application URL  
`http://<server>:<port>/home/<organization>/app/`

3. Click **+** and select the **Customer** entity.
4. Enter the entity details and click **Create**.
5. Click the entity instance to open it and click **Synchronize Workspace**.  
A business workspace is created in Content Server. Similarly, create few more customers.
6. Click **+** and select the **Product** entity. Enter the entity details and click **Create**.
7. Click the entity instance to open it.  
You can select multiple customers and link.  
Similarly, in customer instance you can select multiple products and link.

## Reflexive relation

This topic describes the procedure to define Reflexive relation between business objects.

### Before you begin:

1. Develop an application in AppWorks Platform. See [Developing application in AppWorks Platform](#).
2. Extend the application to define relations by performing the steps in this topic.

This is a special case of peer relation, where relationship is defined with the same entity. Consider a Customer who is also a Supplier.

### To establish a reflexive relation:

1. Open the Customer entity in the CWS workspace.
2. Click **Add > Relationship**.  
The Relationship pane opens.
3. Enter the values:

Field	Description
Display Name	Type a display name.
Name	Type a name.
Type	Select <b>To Peer</b> option.
Multiplicity	Select <b>To One</b> option.
Target Entity	Select the <b>Customer</b> entity.

4. Click  (Save) to save the entity.
5. On the Customer entity, click **Default form > Configure**, move the relationship created to the presentation area.
6. Click  (Save) and publish the entity.

In the application, you can search for the customer and select a relevant supplier.

## **Configurations in Content Server**

### **To configure Content Server:**

1. On the Content Server Administration page, click **Connected Workspaces > Configure Workspace Types**.
2. In workspace type:
  - For the classic view, configure **Side Bar Widgets**. Configure **Related Items** as follows:

Field	Description
Enabled	Select this check box.
Widget	Select <b>Related Items</b> from the list.
Title	Type a title.
Action	a. Click <b>Detailed Configuration</b> . The Related Items Configuration window opens. b. Select the <b>Show Child Relationships</b> check box. c. Click <b>Save Changes</b> .

- For the smart view, configure **Perspective Manager**. To configure Perspective Manager, see *OpenText Extended ECM Platform - Integration Guide*.

## **Configuring Document Store application connector in AppWorks Platform**

**Note:** You must have Administrator privileges in AppWorks Platform.

### **To configure document store application connector in AppWorks Platform:**

1. Create Document Store application connector. See [Creating a service group](#).
2. Configure the Document Store application connector for Content Server in AppWorks Platform. See [Document Store Configuration interface](#).

## **Creating business workspaces from the application**

1. Create a customer instance by accessing the application URL  
`http://<server>:<port>/home/<organization>/app/`
  2. Click  and select the **Customer** entity.
  3. Type the entity details and click **Create**.
  4. Click the entity instance to open it and select a relevant customer as a Customer.
  5. Click **Synchronize Workspace**.
- A business workspace is created in Content Server.

# Cross application business workspace for multiple business objects

In the cross application scenario, the business workspace is already created in Content Server by a leading system. From AppWorks Platform, the created workspace is shared or related. To share or relate from AppWorks Platform, search for an existing business workspace and link it to an entity instance in Content Server.

## Cross application business workspace scenarios

To share or relate the existing business workspace of an application using the AppWorks Platform application, configurations must be done both in AppWorks Platform and Content Server.

- **Shared business workspace:** If you have semantically similar business object types in different applications, such as a customer in Contract Management and a business partner in Customer Management, you can create one cross application business workspace for two or more business objects of different types, and from different applications. In this case, one is a leading application and the other application is sharing a business workspace with it.
- **Related business workspace:** The business workspaces of different applications are related to each other.

## Shared business workspace

### *Configuring shared business workspace*

#### To configure and use cross application business workspace for shared business workspace:

1. Develop a customer application in any xECM supported system such as Success Factor, SAP, Oracle, and treat it as a leading system.
2. Develop an application in AppWorks Platform that shares the business workspace of the leading system.
  - a. Create a Customer entity in AppWorks Platform.  
See [Developing application in AppWorks Platform](#) for more information.
  - b. Mandatory. Open the customer entity. Create an entity property with the corresponding property defined in the leading system.
    - i. To get the value from leading system, on the Content Server Administration page, click **Connected Workspaces > Configure Workspace Types**.
    - ii. Click the leading system's **Workspace Types**.
    - iii. Verify the name pattern provided in **Business Workspace Names**.

**Note:** Consider the values provided in **Business Workspace Names** as common attributes for leading system and AppWorks Platform.

## Configurations in AppWorks Platform

Do the following configurations in AppWorks Platform.

In the AppWorks Administration page, enter the leading system's details.

1. In the AppWorks Administration page, in **Workspaces**, select the project where the entity is configured.
2. In **Configurable Elements > Entities**, select the **Customer** entity.
3. In **Extended ECM**, provide the following configurable parameters:
  - a. In **Business workspace configuration**, select **Cross-application shared workspace**.
  - b. In **External System ID**, type the External System ID of the leading system configured in Content Server.  
To get this value, navigate to **Content Server > Admin tab > Content Server Administration > Extended ECM > Configure Connections to External Systems**, and then select the **Logical External System Name** that is configured to the leading system.
  - c. In **Workspace type ID**, type the workspace ID of the leading system configured in Content Server.  
To get this value, navigate to **Content Server > Enterprise > Connected Workspaces > Workspace Types**. Click and open the leading system's Workspace Type. The URL in the browser's address field shows a string that contains the parameter `ID_CFG`, for example, `ReferenceTypeEdit&ID_CFG=24`. Make a note of the value; in this example, it is 24.
  - d. In **Business Object Type**, type the business object type of the leading system configured in Content Server.  
To get this value, navigate to **Content Server > Admin tab > Content Server Administration > Extended ECM > Configure Business Object Types**, click the leading system's **Business Object Type**, and then select the **Business Object Type** from the Configure Business Object Type page.
  - e. In **Search expression**, type the search expression to use to search the business workspace in the leading system.  
To get this value, go to **Content Server > Enterprise > Connected Workspaces > Workspace Types**, click the leading system's **Workspace Type**, and then read the pattern from Business Workspace Names.  
For example, if in Content Server the pattern is `[100331:Attribute1]-[100331:Attribute2]`, the search expression should be:  
`item.Properties.EntityProperty1+ "-" +  
item.Properties.EntityProperty2.`  
Using this search expression, available business workspaces are listed in the application to link to an entity instance.
  - f. Optional. If the application needs to pull data from the shared workspace to the entity instance, in **Map entity properties to Content Server categories**, do the following steps to map the entity properties with the Content Server category

attributes of the leading system for which you want to view Content Server data in AppWorks Platform. This step is applicable for cross application shared business workspaces.

- i. In **Select an entity property**, select an entity property.
- ii. Click the search icon beside **Select category attribute**.  
The Select a Category window opens.
- iii. Select the leading system category, and then click **Select**.
- iv. Click **Select category attribute**.  
The attributes of the selected category are displayed.
- v. Select a required attribute, and then click **Add** to define the mapping.

## Configurations in Content Server

Do the following configurations in Content Server.

1. Define categories in Content Server for the AppWorks Platform Entity. See [Creating a category](#).

**Note:** It is not required to create Classification, Workspace Type, and document template, as these are shared from the leading system.

2. Configure connections to External Systems. See [Configuring connections to external systems](#).
3. Configure business object types. See [Configuring business object types](#).

**Note:** While creating the business object type, in **Workspace Type**, select the leading system's workspace type.

4. In **Business Properties**, select the leading system's common attributes along with the AppWorks Platform's attributes.
5. In **Workspace Template**, select the leading system's template, and then click **Save Changes** to save the business object type.

## Configuring Document Store application connector in AppWorks Platform

### Note:

- You must have Admin privileges in AppWorks Platform.
- To access cross application business workspace in the application, see the section about system object volume configurations in [Configuring AppWorks Platform in Content Server](#).

To configure the Document Store application connector for Content Server in AppWorks Platform, see Modifying document repository in the *AppWorks Platform Advanced Development Guide*.

## ***Creating entity instances in AppWorks Platform***

### **To search for an existing business workspace for an entity instance:**

1. Access your application - `http://<hostname>/home/<org>/app/start`.
2. Click **+** and select a customer entity.
3. Type the customer details and create an entity instance.
4. In the **Results** panel, select the above created instance and click **Synchronize Workspace**.  
The cross application workspace search window opens.  
All the matched results are displayed based on the search criteria defined on the Admin page.
5. Select a business object to share the business workspace, and then click **OK**.
6. In the **Results** panel, click the instance.  
The instance opens in a new window. You can perform all the document management operations in the **Folder Browser Widget**.

**Note:** Do not edit common field values from AppWorks Platform, else the leading system's values are overwritten.

7. Click **Synchronize workspace** to view the leading system's updated data from Content Server after the business workspace is shared.

**Note:** For an automatic synchronization, **Synchronize Workspace** is disabled. Business workspace sharing is enabled automatically. If more than one search result is found, an error is displayed and **Synchronize Workspace** is enabled to link the required business workspace manually.

## **Related business workspace**

This topic describes the procedure to configure and use cross application business workspace for related business workspace.

### ***Configuring related business workspace***

### **To configure and use cross application business workspace for related business workspace:**

1. Develop a customer application in any xECM supported system such as Success Factor, SAP, Oracle, and treat it as a leading system.
2. Develop an application in AppWorks Platform which shares the business workspace of the leading system.
  - a. Create a Customer entity in AppWorks Platform. See [Developing application in AppWorks Platform](#) for more information.
  - b. Open the customer entity. Create an entity property with the corresponding property defined in the leading system.

- To get the value from leading system, on the Content Server Administration page, click **Connected Workspaces** > **Configure Workspace Types**.
- Click the leading system's **Workspace Types**.
- Verify the name pattern provided in **Business Workspace Names**.

**Note:** Consider the values provided in Business Workspace Names as common attributes for leading system and AppWorks Platform.

## Configurations in AppWorks Platform

In the AppWorks Administration page, enter the details of the leading system.

1. In the AppWorks Administration page, in **Workspaces**, select the project where the entity is configured.
2. In **Configurable Elements** > **Entities**, select the **Customer** entity.
3. In **Extended ECM**, provide the following configurable parameters:
  - a. In **Business workspace configuration**, select **Cross-application related workspace**.
  - b. In **External System ID**, type the External System ID of the leading system configured in Content Server.  
To get this value, navigate to **Content Server** > **Admin** tab > **Content Server Administration** > **Extended ECM** > **Configure Connections to External Systems**, and then select the **Logical External System Name** that is configured to the leading system.
  - c. In **Workspace type ID**, type the workspace ID of the leading system configured in Content Server.  
To get this value, navigate to **Content Server** > **Enterprise** > **Connected Workspaces** > **Workspace Types**. Click and open the Workspace Type of the leading system. The URL in the address field of the browser displays a string that contains the parameter `ID_CFG`, for example, `ReferenceTypeEdit&ID_CFG=24`. Make a note of the value. In this example, it is 24.
  - d. In **Business Object Type**, type the business object type of the leading system configured in Content Server.  
To get this value, navigate to **Content Server** > **Admin** tab > **Content Server Administration** > **Extended ECM** > **Configure Business Object Types**, click the leading system's **Business Object Type**, and then select the **Business Object Type** from the Configure Business Object Type page.
  - e. In **Search expression**, type the search expression to use to search the business workspace in the leading system.  
To get this value, go to **Content Server** > **Enterprise** > **Connected Workspaces** > **Workspace Types**, click the Workspace Type of the leading system, and then read the pattern from Business Workspace Names. For example, if in Content Server the pattern is `[100331:Attribute1]-[100331:Attribute2]`, the search expression should be: `item.Properties.EntityProperty1+ "-" +`

item.Properties.EntityProperty2. Using this search expression, available business workspaces are listed in the application to link to an entity instance.

## Configurations in Content Server

Do the following configurations in Content Server:

1. Define the categories for AppWorks Platform.
  - a. Create category and attributes in Content Server.
  - b. Add attributes of leading system with which you want to search in AppWorks Platform for sharing the business workspace. For example, if the search criteria is \$CustomerID\$ \$CustomerName\$ then create the attributes as CustomerID and CustomerName for AppWorks Platform categories.
2. Create a classification. See [Creating a classification](#).
3. Create a workspace type. See [Creating a workspace type](#).
4. Configure document templates. See [Configuring document templates](#).
5. Open document templates. See [Opening a document template](#).
6. Configure connections to External Systems. See [Configuring connections to external systems](#).
7. Configure business object types. See [Configuring business object types](#).
  - In Business Properties, select the common attributes of the leading system with the attributes of AppWorks Platform.

## Configuring Document Store application connector in AppWorks Platform

### Note:

- You must have Admin privileges in AppWorks Platform.
- To access cross application business workspace in the application, see the section about system object volume configurations in [Configuring AppWorks Platform in Content Server](#).

To configure the Document Store application connector for Content Server in AppWorks Platform, see Modifying document repository in the *AppWorks Platform Advanced Development Guide*.

## Creating entity instances in AppWorks Platform

### To search for an existing business workspace for an entity instance:

1. Access your application - <http://<hostname>/home/<org>/app/start>.
2. Click  and select the required entity. Type the customer details and create an entity instance.

3. In the Results panel, select the above created instance and click **Synchronize Workspace**.  
The cross application workspace search window opens.  
All the matched results are displayed based on the search criteria defined on the Admin page.
4. Select a business object to relate the business workspace, and then click **OK**.
5. In the **Results** panel, click the instance.  
The instance opens in a new window. You can perform all the document management operations in the Folder Browser Widget.

**Note:** For an automatic synchronization, **Synchronize Workspace** is disabled. Business workspace sharing is enabled automatically. If more than one search result is found, then an error is displayed and **Synchronize Workspace** is enabled to link the required business workspace manually.

## Configuring a link to an existing business workspace

When Content Server is the user-facing application, and users manage workspaces and documents directly using it without any other applications, they create workspaces and provide metadata using this server. Other business applications must reuse this existing workspace. This configuration enables the application user to link to the Content Server business workspaces from entity items and get the metadata stored in workspace as part of the entity data .

**Note:** The option to link an existing business workspace is available only if AppWorks Platform is configured with Content Server 16.2 or later.

Example: There is an order record management system in Content Server with the following fields: OrderID, OrderDate, RequiredDate, ModeOfPayment, ShipName, ShipAddress, ShipCity, ShipRegion, ShipPostalCode, ShipCountry, and DeliveryDate.

The Content Server user enters order details like OrderID, OrderDate, RequiredDate, and ModeOfPayment, and creates a business workspace in Content Server. A AppWorks Platform user then uses this business workspace and fills the shipping details like ShipName, ShipAddress, ShipCity, ShipRegion, ShipPostalCode, ShipCountry, and DeliveryDate, and links with the Content Server business workspace. This enables the AppWorks Platform user to use and update the Content Server data and vice versa.

### To link Content Server and AppWorks Platform:

1. **Design and develop an application in AppWorks Platform.**  
In the application, model an Order entity which has Business workspace building block.  
For more information, see [Developing application in AppWorks Platform](#). Publish or deploy the application.
2. **Configure Extended ECM in Content Server.**

**Prerequisites:**

- a. [Creating a category](#).
- b. [Creating a classification](#).
- c. [Creating a workspace type](#).

In the Workspace Type window, in **Business Workspace Names**, click **Insert Attribute**, select **Order:ShipCity** attribute, and then click **Save Changes**.

- d. [Configuring document templates](#).
- e. [Opening a document template](#).
- f. [Configuring connections to external systems](#).
- g. [Configuring business object types](#).

In the Configure Business Object Type window, in the **Property Mapping** section, in **Business Properties**, select **Properties.ShipCity**. You can also select other mappings **Properties.ShipName**, **Properties.ShipAddress**, **Properties.ShipRegion**, **Properties.ShipPostalCode**, **Properties.ShipCountry** and **Properties.DeliveryDate**. Values from AppWorks Platform entity instances are updated in Content Server for the defined property mappings. Search using the field with which the business workspace is created.

**3. Create a Business Workspace in Content Server.**

- a. Navigate to **Content Server > Enterprise > Workspace**.
- b. Click the folder configured for creating the order business workspace.
- c. Click **Add Item > Business Workspace**, and provide the name and description of the business workspace.
- d. Select the **Add business object later** check box and click **Next**.
- e. On the next page, type the metadata values for **OrderId**, **OrderDate**, **RequiredDate**, **ModeOfPayment**.
- f. Click **Finish**.

A business workspace with the given name is created.

**4. Configure the Document Store application connector for Content Server in AppWorks Platform.**

For more information, see Modifying document repository in the *AppWorks Platform Advanced Development Guide*.

**5. Configure a link to an existing business workspace in AppWorks Platform.**

In AppWorks Platform application, you can link the workspace created in Content Server, and update the metadata values from AppWorks Platform to Content Server and vice versa.

- a. In the AppWorks Administration page, in **Workspaces**, select the project where the entity is configured.
- b. In **Configurable Elements > Entities**, select the **Order** entity to link to an existing business workspace.

- c. In **Extended ECM**, provide the following configurable parameters:
- In **Business workspace configuration**, select **Link existing business workspace**.
  - In **Workspace type ID**, type the workspace ID of the entity configured in Content Server.  
To get this value, navigate to **Content Server > Enterprise > Connected Workspaces > Workspace Types**, click and open the configured system's Workspace Type. The URL in the browser's address field shows a string that contains the parameter `ID_CFG`, for example, `ReferenceTypeEdit&ID_CFG=24`. Make a note of the value; in this example, it is 24.
  - In **Search expression**, type the search expression to use to search the existing business workspace in Content Server.  
To get this value, navigate to **Content Server > Enterprise > Connected Workspaces > Workspace Types**, click the configured Workspace Type, and then read the pattern from Business Workspace Names. In this case, it is **item.Properties.ShipCity**. Using this search expression, available business workspaces are listed in the application to link to an entity instance.
  - In **Map entity properties to Content Server categories**, to map the entity properties with the Content Server category attributes for which you want to pull data from Content Server:
    - In **Select an entity property**, select a property.
    - Click the search icon beside **Select category attribute**.  
The Select a Category window opens.
    - Select the order category, and then click **Select**.
    - Click **Select category attribute**.  
The attributes of the Order category are displayed.
    - Click **Add**.
    - Similarly, map the order entity properties with the Content Server category attributes for which you want to pull data from Content Server. In this case, map for OrderID, OrderDate, RequiredDate, and ModeOfPayment.

**Note:** Ensure that the property mapping done in Content Server in the business object type does not overlap with the mapping done in the AppWorks Administration. If the same property mapping is done at both ends, data is overwritten with the latest update.

- Create an Order entity instance in AppWorks Platform to link it to an existing business workspace.**
  - Access your application - `http://<hostname>/home/<org>/app/start`.
  - Click **+** and select the **Order** entity.  
The Create Order window opens.

- c. Type the order details for **ShipName**, **ShipAddress**, **ShipCity**, **ShipRegion**, **ShipPostalCode**, **ShipCountry**, and **DeliveryDate**.
  - d. Click **Create**.  
An order instance is created.
  - e. Click and open the order instance, and then click **Synchronize workspace**.  
The Create or complete workspace window opens. Matched results are displayed according to the search criteria defined on the AppWorks Administration page.

**Note:** The Synchronize workspace option is available only when manual synchronization is enabled in the document store application connector configuration. For automatic synchronization, business workspace is linked automatically when the entity instance is created, based on the search expression. If more than one search result is found, an error is logged and the Synchronize Workspace option is enabled to link the required business workspace manually.
  - f. Select a business workspace to link the entity instance and click **Complete**. On successful linking, based on the mapping configuration, you can view the Content Server attribute values in AppWorks Platform and vice versa.
  - g. If no matches are found, click **Create new**, provide the order details, and then click **Create**.  
A business workspace in Content Server is created.
7. After the business workspace is linked, to view the updated data from the Content Server, click **Synchronize workspace**.

**Note:** Once a link is established, it cannot be unlinked. The AppWorks Platform metadata values are available in Content Server.

# Part VI

# Reference



# Chapter 45

## Reference

This section contains topics that provide information about the concepts, features, and interfaces of AppWorks Platform Administration.

---

# Administration

Administration refers to maintenance and management of activities performed by an Administrator. Reference topics listed in this section will help you understand the non-interface based features and functionalities of AppWorks Platform. This section contains the following topics:

- **AppWorks Platform User Privileges:** All the AppWorks Platform services begin from the AppWorks Platform User context. This is to prevent security hacks and granting lesser privilege so as to avoid services from disturbing the system environment. This topic provides information on types of users and their permissions on AppWorks Platform components.
- **Platform Properties:** Platform Properties are used by the platform to start its resources. The wcp.properties file is the corresponding file for these properties. This section covers the properties for LDAP, Web gateway, XML, NOM, Management Console, UDDI components.
- **Managing Alert System:** Alert System in AppWorks Platform has been designed to monitor the health of software. It is a mechanism through which an operator or an administrator of a system is informed of undesirable situations such as exceptions, errors, overload on a part of the system and so on. This section includes information on alerts, filters, appenders and alert repository.
- **Managing Logging Framework:** The logging framework is a key feature of AppWorks Platform that records certain critical activities during runtime. The information gathered from the log files is used to analyze the error state and fix support calls. This section provides extensive information about logging framework process, log message formats, consumers, categories and severity levels.
- **Starting a batch of Service Containers:** This topic provides an overview about starting a set of Service Containers in batches.
- **Creating a Configuration Profile for FTP Server:** This topic describes the procedure to create a profile for FTP server configuration details. You can use this profile while invoking the FTP SOAP requests.
- **Configuring File System Permissions of CWS Folders on a Linux Environment:** This topic describes how to configure the file system permissions of CWS folders on a Linux environment.

## Alert repository for AppWorks Platform

This table describes the alerts available in AppWorks Platform:

<b>Alert ID and text</b>	<b>Level and problem state related</b>	<b>Description and instructions for the admin or operator to correct the situation</b>
Cordys.BAM.DebugMessages.monitoringEngineStopped  BAM Monitoring engine stopped.	Informational No	This alert is issued when the running BAM Monitoring engine is stopped.  Check the log files to find the cause for this failure. Restart the BAM service container.
Cordys.BAM.ErrorMessages.ruleEngineInitializationFailure  The embedded monitoring rule engine in BAM service container is not initialized.	Error No	This alert is issued when the embedded monitoring rule engine initialization has failed.  Verify the BAM service container log for the errors and correct them.  The error can be due to an invalid configuration in the BAM service container, or when a database configuration with invalid database details is selected for the embedded monitoring rule engine in the BAM service container.
Cordys.BAM.ErrorMessages.scheduleEngineInitializationFailure  The embedded monitoring schedule engine in BAM service container is not initialized.	Error No	This alert is issued when the embedded monitoring schedule engine initialization has failed.

<b>Alert ID and text</b>	<b>Level and problem state related</b>	<b>Description and instructions for the admin or operator to correct the situation</b>
		Verify the BAM service container log for the errors and correct them. The error can be due to an invalid configuration in the BAM service container, or when a database configuration with invalid database details is selected for the embedded monitoring schedule engine in the BAM service container.
Cordys.BAM.ErrorMessages.stateSynchUpFailure The State SynchUp (SSU) initialization has failed in BAM service container. SSU is needed for maintaining the cache coherence across the BAM service containers.	Error No	This alert is issued when the State SynchUp configuration encounters issues. <ol style="list-style-type: none"><li>1. Connect to CARS using CMC and verify the monitor syncup information entry.</li><li>2. Verify the machine name and port.</li><li>3. Restart the BAM service container.</li></ol>
Cordys.BAM.ErrorMessages.threadDispatcherFailure	Error	This alert is issued

<b>Alert ID and text</b>	<b>Level and problem state related</b>	<b>Description and instructions for the admin or operator to correct the situation</b>
An unexpected error occurred during the thread execution of the dispatcher "{0}".	No	when an unexpected error occurred during the thread execution of the dispatcher "{0}". Verify the BAM service container log for the errors and correct them. The error may be due to an invalid Process Monitoring Object (PMO) or Business Event Response (BER).
Cordys.BPM.Messages.archiveFileZippingFailure The archive file was not created successfully. Create a zip folder at "{0}" with the same folder name.	Error No	This alert is issued when the archive file is not created successfully due to a failure in zipping of the files. Verify the log for the error and correct it. This issue may occur due to the following reasons: <ul style="list-style-type: none"> <li>■ Insufficient space</li> <li>■ No permissions in the folder where the archive is being created.</li> </ul>
Cordys.BPM.Messages.binaryParserInitializationFailure The embedded binary parser did not initialize. Verify the	Error No	This alert is issued when the embedded binary parser does

<b>Alert ID and text</b>	<b>Level and problem state related</b>	<b>Description and instructions for the admin or operator to correct the situation</b>
service container configuration properties.		not initialize. Verify the log file for the error and correct it. There may be an issue in the configuration of the binary parser embedded in the business process engine. Verify the configuration properties of the corresponding BPM service container.
Cordys.BPM.Messages.dbConnectionFailure The database server may not be functioning or the network may be encountering server connectivity issues.	Error No	This alert is issued when the processor cannot be started due to a database connectivity issue. Verify the log for the error and correct it. The issues may be due to the following reasons: <ul style="list-style-type: none"><li>■ The number of connections to the database may have exceeded the specified limit.</li><li>■ The connection thread to the service container and database may be closed.</li></ul>

<b>Alert ID and text</b>	<b>Level and problem state related</b>	<b>Description and instructions for the admin or operator to correct the situation</b>
		<ul style="list-style-type: none"> <li>■ The database server or service may be down.</li> </ul>
<p>Cordys.BPM.Messages.notificationSOAPProcessorFailure</p> <p>The system was unable to deliver the task or notification from the process engine. The Notification service container is not functioning properly or has not yet started.</p>	Error No	<p>This alert is issued when the Notification service container fails to function.</p> <p>Verify the log for the error. Identify the issue in the Notification service container and correct it.</p>
<p>Cordys.BPM.Messages.reachedMaxAllowedInstanceCount</p> <p>The rate of creation of new process instances exceeded the threshold limit set by the administrator. Try after some time or contact the administrator.</p>	Warning No	<p>The current rate of creation of new process instances "{0}" for the business process model "{1}" deployed to the model-space "{2}" exceeded the threshold limit of "{3}" process-instances per "{4}" second(s) set by the administrator. The request is sent from the organization "{5}" by the user "{6}". Try after some time or contact the administrator.</p> <p>Verify the</p>

<b>Alert ID and text</b>	<b>Level and problem state related</b>	<b>Description and instructions for the admin or operator to correct the situation</b>
		requirement for having the specified number of instances created in the specified time interval. If the number of instances is valid, update the threshold limit.
Cordys.BPM.Messages.reachedMaxIterationCount The loop activity crossed the maximum number of iterations.	Warning No	The business process instance with the instance ID "{0}" of the business process model "{1}" is suspended by the system as it has reached the threshold limit "{7}" set by the administrator. The details of the process instance are provided below: {2} - DN of the organization in whose context the process instance is instantiated {3} - DN of the user who created the instance {4} - Current iteration count of the activity {5} - Name of the activity in which the iteration count exceeded the threshold limit {6} -

<b>Alert ID and text</b>	<b>Level and problem state related</b>	<b>Description and instructions for the admin or operator to correct the situation</b>
		<p>Type of the activity</p> <ol style="list-style-type: none"> <li>1. Verify the Business Process Model that is creating infinite loops.</li> <li>2. Verify whether it is required to execute that many iterations.</li> <li>3. Correct the Business Process Model.</li> </ol>
<p>Cordys.BPM.Messages.ruleEngineInitializationFailure</p> <p>The embedded rule engine did not initialize. Check the BPM service container configuration properties.</p>	Error No	<p>This alert is issued when the embedded rule engine does not initialize.</p> <p>Verify the BPM service container log for the error and correct it. It may be due to configuration issues or invalid database details provided for the rule engine embedded in the BPM service container configuration.</p>
<p>Cordys.BPM.Messages.stateSyncUpFailure</p> <p>The State SyncUp (SSU) is not configured. SSU is required for the synchronization between multiple BPM engines that are configured for load balancing.</p>	Error No	<p>This alert is issued when the State SyncUp configuration encounters issues.</p> <ol style="list-style-type: none"> <li>1. Connect to CARS using CMC and</li> </ol>

<b>Alert ID and text</b>	<b>Level and problem state related</b>	<b>Description and instructions for the admin or operator to correct the situation</b>
		verify the monitor syncup information entry. 2. Verify the machine name and port.
Cordys.BPM.Messages.unableToResolveDso The process engine was unable to resolve the data source object from the specified information in the business process model; transactional process "{0}" for instance ID "{1}".	Error No	This alert is issued when the database configuration information required to initialize the transaction used in the business process cannot be retrieved. 1. Verify whether the service container and the data source object are defined properly to execute the specified WS-AppServer. 2. Verify the log for the error and correct it.
Cordys.Case.Messages.AddingProcessToQueueFailed An error occurred while adding a process instance to the queue for execution. The error is: {0}.	Informational No	An error occurred while adding a process instance to the queue for execution. The error is: {0}. Not Applicable.
Cordys.Coboc.Messages.dbConnectionFailure	Error	This alert will be

<b>Alert ID and text</b>	<b>Level and problem state related</b>	<b>Description and instructions for the admin or operator to correct the situation</b>
The database server may not be functioning or the network may be encountering server connectivity issues.	No	issued when CoBOC fails to establish a connection with the database server. Ensure you are able to connect to the database, and verify if the database is up and running.
Cordys.Coboc.Messages.stateSyncUpFailure  State SyncUp is not configured. It is required to enable the orchestrator cache synchronization.	Error No	This alert is issued when the State SyncUp configuration encounters issues. <ol style="list-style-type: none"><li>1. Go to the Cordys system database and verify the syncup_configuration in the Coboc_syncup_info table.</li><li>2. Verify the Machine IP address and port number.</li><li>3. Verify if the port number defined is free and available to form a ring. If it is not free, change the port number and restart the service container.</li></ol>

<b>Alert ID and text</b>	<b>Level and problem state related</b>	<b>Description and instructions for the admin or operator to correct the situation</b>
Cordys.DBConnection.Messages.databaseConnectionEstablished Connection to the database server has been established.	Informational Leave	This alert is issued when the connection to the database server is established again. Information. No action required.
Cordys.DBConnection.Messages.databaseConnectionFailure  The database server is not available. The database server may not be functioning, or the network may be encountering server connectivity problems. Ensure that the database server is functioning, and the network is communicating properly with the server.	Error Enter	This alert is issued when connection to the database server is lost.  Ensure the database server is running and the database is accessible.
Cordys.Email.Messages.mailServerConnectionEstablished  The connection to the mail server has been established.	Informational Leave	This alert is issued when the connection to the mail server is successful.  Information. No action required.
Cordys.Email.Messages.mailServerConnectionFailure  The client cannot connect to the mail server as the mail server may be unavailable, the port may be busy, or there may be other network issues.	Error Enter	This alert is issued when the connection to the mail server fails.  Verify the availability of the mail server.
Cordys.ESBClient.Messages.directoryConnectionEstablished  The connection to the directory server has been established.	Informational Leave	This alert is issued when the connection to the directory server is reestablished.  Information. No action required.

<b>Alert ID and text</b>	<b>Level and problem state related</b>	<b>Description and instructions for the admin or operator to correct the situation</b>
Cordys.ESBClient.Messages.directoryConnectionFailure  The directory server is unavailable. The directory server may not be functioning, or the network may be encountering server connectivity issues. Ensure that the directory server is functioning, and the network is communicating properly with the server.	Error Enter	This alert is issued when the connection to the directory server is lost.  1. Verify the status of CARS. 2. Restart CARS and the AppWorks Platform (<instance name>) service.
Cordys.ESBClient.Messages.ldapReconnectionSuccessful  The LDAP server connection was lost but after reconnection, it succeeded.	Informational No	This alert is issued when the connection to the LDAP server fails but immediate reconnection succeeds.  Information. No action required.
Cordys.ESBClient.Messages.newSyncupGroupSuccess  The members {0} have formed a new group.	Informational No	This alert is issued when a new member joins the group.  Information. No action required.
Cordys.ESBClient.Messages.syncupMembersRemoved  The members '{0}' are no longer a part of the group.	Warning No	This alert is issued when a member is removed from the SyncUp group.  Verify the log information in the Monitor.xml file. Based on the error information, connect to CARS. Verify the

<b>Alert ID and text</b>	<b>Level and problem state related</b>	<b>Description and instructions for the admin or operator to correct the situation</b>
		monitor SyncUp info entry and ensure that both the 'Port' and 'Machine Name' specified are correct.
Cordys.ESBServer.Messages.consolidatedReportToSendLicenseServiceFailure  The consolidated report cannot be sent to the license service '{0}'.	Error No	This alert is issued when the consolidated report cannot be sent to the license service.  Upload the License usage report through the License portal or contact Support.
Cordys.ESBServer.Messages.jvmOutOfMemoryError  The Java heap size has been exhausted for the process '{1}' residing on the host '{0}'. The process will be exited.	Error No	This alert is issued when a JVM hosting the Cordys service container exhausts its Java heap capacity.  Verify the log files for any cause or additional information, or contact Support.
Cordys.ESBServer.Messages.license_Valid  The license key is valid for {0} days.	Informational No	This alert is issued to provide information on the license validity.  Information. No action required.
Cordys.ESBServer.Messages.licenseExpired  The license expired {0} days ago. Use the License Manager to send the usage report and obtain a new	Error No	This alert is issued when the license validity expires.

<b>Alert ID and text</b>	<b>Level and problem state related</b>	<b>Description and instructions for the admin or operator to correct the situation</b>
license key.		License is expired and update the new license key from the License Manager.
Cordys.ESBServer.Messages.licenseKeyLoadFailure The key cannot be loaded on the {0}. Verify the status of the Web Server and Monitor Service on this computer.	Error No	This alert is issued when the license key cannot be loaded on a computer.  In LDAP, verify if the licinfo entry is available. If the licinfo is unavailable, contact Support to get the renewed license key and import the information in the form of .ldif format into CARS.
Cordys.ESBServer.Messages.licenseReportReceiveFailure The report cannot be obtained from the {0}. Verify the status of the Web Server and Monitor Service on this computer.	Error No	This alert is issued when the license service fails to receive a report from a computer.  Upload the License usage report through the License portal or contact Support.
Cordys.ESBServer.Messages.monitorCrash The Cordys monitor has terminated abnormally: host '{0}' processid '{1}'	Error No	This alert is issued when the AppWorks Platform (<instance name>) crashes.  1. Verify the log files for any

<b>Alert ID and text</b>	<b>Level and problem state related</b>	<b>Description and instructions for the admin or operator to correct the situation</b>
		<p>cause.</p> <ol style="list-style-type: none"> <li>2. After restarting the Monitor, verify the memory usage of the Monitor and service container.</li> <li>3. Keep the crashed log files and log files, and contact Support.</li> </ol>
Cordys.ESBServer.Messages.monitorStarted The Cordys monitor has started. {0}	Informational No	<p>This alert is issued when the AppWorks Platform (&lt;instance name&gt;) starts.</p> <p>Information. No action required.</p>
Cordys.ESBServer.Messages.monitorStartErrorBoots trapFailure  The Cordys monitor cannot be started as an error occurred while loading the bootstrap information.	Error No	<p>This alert is issued when the AppWorks Platform (&lt;instance name&gt;) cannot be started as there was an error while loading the bootstrap information.</p> <p>Check the log file of the Monitor for detailed error information. Think of issues like connectivity to CARS, non-existence of &lt;cordys_</p>

<b>Alert ID and text</b>	<b>Level and problem state related</b>	<b>Description and instructions for the admin or operator to correct the situation</b>
		installdir>/config/DMZ.xml.
Cordys.ESBServer.Messages.monitorStartErrorEntry NotInLdap  The Cordys monitor '{0}' has not been registered on the LDAP server '{1}', and therefore cannot be started.	Error No	This alert is issued when the AppWorks Platform (<instance name>) entry is not present in LDAP.  The baseline was not installed properly. Therefore, you must uninstall and install the baseline.
Cordys.ESBServer.Messages.monitorStartErrorExceptionOccured  A fatal error occurred while starting the Cordys monitor.	Error No	This alert is issued when the AppWorks Platform (<instance name>) cannot be started because of an exception.  Analyze the log files and the operating system log to find the exception causing the monitor to fail and resolve it.
Cordys.ESBServer.Messages.monitorStartErrorLdapFailure  The Cordys monitor cannot be started as accessing the LDAP server has failed. Ensure that the LDAP server is running. If so, the <machine> may be slow. Add the 'bus.ldaprequest.timeout=<timeout>' property in the wcp.properties file and provide the duration for the request timeout.	Error No	This alert is issued when the AppWorks Platform (<instance name>) cannot be started because of an LDAP connectivity issue.  1. Verify if the LDAP server is accessible.

<b>Alert ID and text</b>	<b>Level and problem state related</b>	<b>Description and instructions for the admin or operator to correct the situation</b>
		2. Verify the machine-specific Multicast configuration.
Cordys.ESBServer.Messages.monitorStartErrorLdapProcessorFailure  The Monitor could not be started due to failure to access the LDAP Server. Ensure that the LDAP server is running. If so, the <machine> could be slow. Add the 'bus.ldaprequest.timeout=<timeout>' property in the wcp.properties file and specify the duration for request time out.	Error No	This alert is issued when the monitor cannot be started because of LDAP connectivity problem.  The AppWorks Platform (<instance name>) cannot be started as contacting CARS has failed. Verify if CARS is accessible.
Cordys.ESBServer.Messages.monitorStartErrorLicenseInvalid  The Cordys monitor cannot be started as it has an invalid license key. Provide the valid license key.	Error No	This alert is issued when the AppWorks Platform (<instance name>) cannot be started due to an invalid license.  The license key is invalid. Update the license key through the License Manager.
Cordys.ESBServer.Messages.monitorStartErrorLicensekeyReadFailure  The Cordys monitor cannot be started as reading the license key has failed.	Error No	This alert is issued when the AppWorks Platform (<instance name>) cannot be started as the license key cannot be read.

<b>Alert ID and text</b>	<b>Level and problem state related</b>	<b>Description and instructions for the admin or operator to correct the situation</b>
		In LDAP, verify if the licinfo entry is available. If the licinfo entry is unavailable, contact Support to obtain the renewed license key and import the information in the form of the '.ldif' format into CARS.
Cordys.ESBServer.Messages.monitorStartErrorLicensekeyReadLdapFailure  The Cordys monitor cannot be started as reading the license key from LDAP has failed.	Error No	This alert is issued when the AppWorks Platform (<instance name>) cannot be started as reading the license key from LDAP has failed.  In LDAP, verify if the licinfo entry is available. If the licinfo entry is unavailable, contact Support to obtain the renewed license key and import the information in the form of the '.ldif' format into CARS.
Cordys.ESBServer.Messages.monitorStartErrorSyncupFailure  The Cordys monitor cannot be started as the SyncUp initialization has failed. Ensure that both the 'Port' and 'Machine Name' specified are correct.	Error No	This alert is issued when the AppWorks Platform (<instance name>) cannot be started on failure of the SyncUp intialization.

<b>Alert ID and text</b>	<b>Level and problem state related</b>	<b>Description and instructions for the admin or operator to correct the situation</b>
		<ol style="list-style-type: none"> <li>1. Connect to CARS using CMC.</li> <li>2. Verify the monitor SyncUp info entry.</li> <li>3. Verify the 'Machine Name' and 'Port' values.</li> </ol>
Cordys.ESBServer.Messages.processLaunch Launching the process '{0}'	Informational No	<p>This alert is issued when a process is in the starting mode.</p> <p>Information. No action required.</p>
Cordys.ESBServer.Messages.processLaunchError The process '{0}' cannot be launched.	Error No	<p>This alert is issued when launching of the process fails.</p> <p>Verify the AppWorks Platform (&lt;instance name&gt;) logs.</p>
Cordys.ESBServer.Messages.processorStartErrorExceptionOccured The Service Container {0} cannot be started.	Error No	<p>This alert is issued when the service container cannot be started because of an unexpected error.</p> <ol style="list-style-type: none"> <li>1. Verify the classpath settings of the service container.</li> <li>2. Verify the corresponding service</li> </ol>

<b>Alert ID and text</b>	<b>Level and problem state related</b>	<b>Description and instructions for the admin or operator to correct the situation</b>
		container log file.
Cordys.ESBServer.Messages.processorStartErrorLdapFailure  The Service Container '{0}' cannot be started as the service container entry was not found in LDAP.	Error No	This alert is issued when the service container cannot be started as the LDAP server connection fails.  Verify the LDAP service container status. If the LDAP service container is not started, then restart the AppWorks Platform (<instance name>).
Cordys.ESBServer.Messages.processorStartErrorLicenseInvalid  The Service Container cannot be started due to an invalid license key. Provide the correct license key.	Error No	This alert is issued when the Service Container cannot be started due to an invalid license.  The corresponding service container entry is not registered in LDAP. Therefore, restart the AppWorks Platform (<instance name>) and create the new service container in the System Resource Manager.
Cordys.ESBServer.Messages.processorStartErrorLicensekeyReadError  The Service Container cannot be started as reading the	Error No	This alert is issued when the Service Container cannot be

<b>Alert ID and text</b>	<b>Level and problem state related</b>	<b>Description and instructions for the admin or operator to correct the situation</b>
license key has failed due to an error.		started as reading the license key has failed. Verify the license server status and verify the licinfo entry in the LDAP server.
Cordys.ESBServer.Messages.processorStartErrorLicensekeyReadLdapError  The Service Container cannot be started as reading the license key from LDAP has failed.	Error No	This alert is issued when the Service Container cannot be started as reading the license key from LDAP has failed. Verify the license server status and verify the licinfo entry in the LDAP server.
Cordys.ESBServer.Messages.processorStartErrorSocketFailure  The Service Container '{0}' cannot be started as opening the socket configured for the connection point has failed.	Error No	This alert is issued when the service container cannot be started as there was an issue with the connection point. <ol style="list-style-type: none"><li>1. Verify the classpath settings of the service container.</li><li>2. Verify the corresponding service container log file.</li></ol>
Cordys.ESBServer.Messages.processStop	Information	This alert is issued

<b>Alert ID and text</b>	<b>Level and problem state related</b>	<b>Description and instructions for the admin or operator to correct the situation</b>
Stopping process {0}.	Informational No	when a process is in the stopping mode. Information. No action required.
Cordys.ESBServer.Messages.reportSentToLicense The usage report is sent to the license service '{0}' and the response is '{1}'.	Informational No	This alert is issued when the usage report is sent to the license service and the response is received. Information. No action required.
Cordys.ESBServer.Messages.reportSentToMaster The usage report is sent to the master gateway '{0}' and the response is '{1}'.	Informational No	This alert is issued when the usage report is sent to the master gateway and the response is received. Information. No action required.
Cordys.ESBServer.Messages.soapProcessorFailure The Service Container '{0}' has crashed. {1}	Error No	This alert is issued when the service container crashes. Analyze the log files to find any cause of this failure or contact Support.
Cordys.ESBServer.Messages.soapProcessorStarted The Service Container {0} has been started.	Informational No	This alert is issued when the service container starts successfully. Information. No action required.
Cordys.ESBServer.Messages.soapProcessorStopped The Service Container '{0}' has been stopped.	Informational	This alert is issued when the service

<b>Alert ID and text</b>	<b>Level and problem state related</b>	<b>Description and instructions for the admin or operator to correct the situation</b>
	No	container stops. Information. No action required.
Cordys.ESBServer.Messages.usageReportNotSendGatewayError  The usage report cannot be sent to the master gateway '{0}'.	Error No	This alert is issued when the usage report cannot be sent to the master gateway.  Upload the License report data through the License portal or contact Support.
Cordys.ESBServer.Messages.usageReportNotSendLicenseServiceFailure  The usage report cannot be sent to the license service '{0}'.	Error No	This alert is issued when the usage report cannot be sent to the license service.  Verify the license server status and verify the licinfo entry in the LDAP server.
Cordys.LDAP.Messages.ldapConnectionFailure  The LDAP Server is not available. There was a problem connecting to the server. Ensure that the server is running.	Error No	This alert is issued when connection to the LDAP server fails.  Restart the LDAP server.
Cordys.Notification.Messages.emailSOAPProcessorFailure  The Email SOAP processor is not functioning properly or has not yet been started.	Error No	The Email SOAP processor is not functioning properly or has not yet been started.  Verify if the Email SOAP processor is

<b>Alert ID and text</b>	<b>Level and problem state related</b>	<b>Description and instructions for the admin or operator to correct the situation</b>
		configured properly and functioning.
Cordys.Notification.Messages.eventSOAPProcessorFailure  The Event SOAP processor is not functioning properly or has not yet been started.	Informational No	The Event SOAP processor is not functioning properly or has not yet been started.  Verify if the Event SOAP processor is configured properly and is functioning.
Cordys.Scheduler.Messages.stateSyncUpFailure  The State SyncUp (SSU) is not configured. It is required to enable the Scheduler cache synchronization.	Error No	The State SyncUp (SSU) is not configured. It is required to enable the Scheduler cache synchronization.  Connect to CARS using Management Console and verify the monitor SyncUp info entry. Verify the machine name and port number.
Cordys.Search.Messages.indexingFailure  {0}. A problem is encountered in indexing the content.	Error No	This alert is issued in case of failure while building indexes for documents in product documentation.  This alert is not displayed for the administrator or users.
Cordys.Search.Messages.searchFailure	Error	This alert is issued

<b>Alert ID and text</b>	<b>Level and problem state related</b>	<b>Description and instructions for the admin or operator to correct the situation</b>
{0}. A problem is encountered while searching.	No	in case of failure while searching product documentation with a search keyword. Information. No action required.
Cordys.WebGateway.Messages.gatewayErrorLdapConnectionFailure  The gateway failed to process the request as establishing the connection with the LDAP service container has failed.	Error No	This alert is issued when the gateway fails to process the request as the LDAP server connection fails.  Restart the AppWorks Platform (<instance name>) as it is unable to send the request to the LDAP service container.
Cordys.WebGateway.Messages.gatewayErrorSocketFailure  The gateway failed to process the request as opening the socket connection has failed.	Error No	This alert is issued when the gateway fails to process the request because of a socket issue.  The inbound gateway socket cannot be created. Therefore, change the gateway port in LDAP and restart the AppWorks Platform (<instance name>) and the web application server.
Cordys.WebGateway.Messages.webApplicationFailure	Error	This alert is issued

<b>Alert ID and text</b>	<b>Level and problem state related</b>	<b>Description and instructions for the admin or operator to correct the situation</b>
The Web application cannot be executed. Verify the computer classpath settings and memory settings of the gateway.	No	<p>when execution of the web application fails.</p> <ol style="list-style-type: none"> <li>1. Verify if the ISAPI module or Apache module is registered.</li> <li>2. Restart the web application server.</li> </ol>
<p>Cordys.WSAppServer.Messages.LoadClassRegistryAborted</p> <p>Class registry loading was aborted.</p>	Error No	<p>Error in loading class registry</p> <p>This issue occurs in two scenarios:</p> <ul style="list-style-type: none"> <li>■ When loading the class registry at the Organization level fails</li> <li>■ When loading the class registry at the ISV level fails</li> </ul> <p>In both the above scenarios, to correct the issue, reset or restart the WS-AppServer service container or invoke the <code>reloadClassRegistry</code> API.</p>
Cordys.WSAppServer.Messages.MissingClassDefinition	Error No	This alert is issued when the class

<b>Alert ID and text</b>	<b>Level and problem state related</b>	<b>Description and instructions for the admin or operator to correct the situation</b>
Class {0} not found.		<p>definition is not found.</p> <p>Ensure that the required jar files are in the classpath of the Ws-AppServer service container.</p> <ul style="list-style-type: none"> <li>■ If the jar files are in the 'Ws-AppServer deploy folder', reset or restart the Ws-AppServer service container.</li> <li>■ If the Ws-AppServer component is embedded in any other component, the required jar files must be placed in the classpath of the corresponding service container. Then, restart the service container.</li> </ul>
<p><code>Cordys.WSAppServer.Messages.missingTagAndNamespaceInClass</code></p> <p>Ensure that the tag and namespace '{0}' for the required class is defined in the class registry.</p>	Error No	This alert is issued as the missing tag and namespace for the required class is not defined in the

<b>Alert ID and text</b>	<b>Level and problem state related</b>	<b>Description and instructions for the admin or operator to correct the situation</b>
		class registry. Reload the class registry by resetting or restarting the Ws-AppServer service container. If the Ws-AppServer component is embedded in any other component, the corresponding service container must be restarted too.

## ***Understanding the alert system***

An alert is a notable occurrence (possibly automated) that is of interest to an AppWorks Platform administrator or an operator. Alerts can be detected and described programmatically.

Examples of alerts that are used in AppWorks Platform are:

- Service has started, stopped, or crashed
- Unauthorized access to the system
- Connection with the database could not be established
- Service containers go into a problem state

Entering and leaving the problem state is indicated through alerts (error and informational respectively). See *Working With Problem Registry* in the *AppWorks Platform Advanced Development Guide*.

The Alert System in AppWorks Platform is designed to monitor the health of a software system. The Alert System informs an operator or an administrator of a system of undesirable situations such as exceptions, errors, and overload on a part of the system. With such a mechanism in place, the operators can proactively respond to deviations in performance and optimize the system usage, thereby preventing the error condition spreading to the other parts of the system. The Alert System provides the functionality to define an alert on a managed component and issue the alert when the need arises.

---

The Alert System issues, captures, processes, and forwards alerts to various consumers. A consumer is a program that processes the alerts appropriately. The Alert System in AppWorks Platform has the following features:

- Defines the standard structure of alerts
- Provides an interface using which the AppWorks Platform and AppWorks Platform-based applications issue alerts
- Possesses criteria to filter alerts
- Routes the alerts to consumers for specific purposes
- Issues alerts such as JMX notifications
- Manages the descriptive text within the messages
- Provides alerts in default culture-neutral language

An alert can be of the level: ERROR, WARNING, or INFORMATION, depending on the severity of the alert. All the alerts use the Log4J category `com.eibus.management.AlertSystem`.

The [Alert repository for AppWorks Platform](#) describes all the defined alerts and the following information:

Field	Description	Example
<b>Problem state related</b>	Whether this alert is related to a problem state. Possible values are: <ul style="list-style-type: none"><li>■ No: alert does not relate to the problem state.</li><li>■ Enter: alert is issued when entering the problem state.</li><li>■ Leave: alert is issued when leaving the problem state.</li></ul> See Working With Problem Registry in the <i>AppWorks Platform Advanced Development Guide</i> .	No
<b>ID</b>	Unique identification of the alert that is only exposed through JMX  For example: <code>Cordys.ESBServer.Messages.MONITOR_START_ERROR_EXCEPTION_OCCURRED</code>	<code>Cordys.ESBServer.Messages.MONITOR_START_ERROR_EXCEPTION_OCCURRED</code>
<b>Level</b>	Error, warning, or informational  For example: <code>The OpenText AppWorks Platform (&lt;instance name&gt;) could not start owing to a catastrophic exception</code>	Error

Field	Description	Example
	that occurred while starting the same.	
<b>Message text</b>	English alert message	The AppWorks Platform (<instance name>) could not start owing to a catastrophic exception that occurred while starting the same.
<b>Description</b>	Description of the alert that is exposed through JMX  For example:  This alert is issued when the OpenText AppWorks Platform (<instance name>) is unable to start because of an exception.	This alert is issued when the AppWorks Platform (<instance name>) is unable to start because of an exception.
<b>Instructions</b>	Instructions for the administrator or operator to correct the situation  For example:  Analyze the AppWorks Platform log files and the operating system log to find the exception causing the monitor to fail and resolve that.	Analyze the AppWorks Platform log files and the operating system log to find the exception causing the monitor to fail and resolve the issue.

This section covers the following topics:

- [Settings for alert configuration](#)
- [Filters for Alert System](#)
- [Appenders for an alert system](#)
- [Sending alerts](#)

### ***Settings for alert configuration***

This topic shows how an administrator can configure the alert system for filters and appenders by modifying the settings in the configuration file. See [Filters for Alert System](#) for filters and [Appenders for an alert system](#) for appenders. The configuration file contains settings for configuring filters, appenders, and priority (severity levels). Based on the values specified in this file, alert system initialization is performed. There is a common configuration file defined for both alert and logging, `Log4jConfiguration.xml` located in `<AppWorks Platform_installdir>/config`.

The `<category>` node, `com.eibus.management.AlertSystem` is mandatory to define alert configuration in `Log4jConfiguration.xml`. The configuration specified in `Log4jConfiguration.xml` defines the behavior of the alert system. Structure of an alert configuration:

```

<category name='com.eibus.management.AlertSystem' additivity='true'>
<priority value='info' /> <appender-ref ref=' alertfile' /> </category>
<appender
class='com.eibus.util.logger.appenders.ProcessNamedDailyRollingFileAppender'
name='alertfile'> <param name='File' value='%N.xml' /> <param
name='DatePattern' value=".yyyy-MM-dd" /> <layout
class='org.apache.log4j.xml.XMLLayout' /> </appender>

```

**Note:** By default, during installation the installer configures alert and logging with default values. Default configuration:

```

<category name='com.eibus.management.AlertSystem'> <priority
value='info' /> <appender-ref ref="OSBasedEventAppender"/> </category>
<appender name="OSBasedEventAppender"> <layout> <param name="LocationInfo"
value="true"/> </layout> </appender>

```

This means that the consumers or appenders for both logging and alerts are the same. All parameters required for an alert system must be defined under `<category>`. These parameters include the consumers or appenders, filters, and severity levels. These are the elements used in the configuration file.

Element	Description
<b>category</b>	Name attribute. Represents the fully qualified class name of the Alert System. The default value for <b>additivity</b> is <b>true</b> . If set to false for a logger, it prevents parents of that logger from using their appenders. For more details see <a href="http://logging.apache.org/log4j/1.2/manual.html">http://logging.apache.org/log4j/1.2/manual.html</a> .
<b>priority</b>	Type or level of message to issue. The value attribute is used to specify the type and its hierarchy. Possible values are: <ul style="list-style-type: none"> <li>■ DEBUG: Alerts defined on the system (DEBUG, INFO, WARN, ERROR, FATAL) are issued.</li> <li>■ INFO: Alerts defined on the system (INFO, WARN, ERROR, FATAL) are issued.</li> <li>■ WARN: Alerts defined on the system (WARN, ERROR, FATAL) are issued.</li> <li>■ ERROR: Alerts defined on the system (ERROR, FATAL) are issued.</li> <li>■ FATAL: Alerts defined on the system (FATAL) are issued.</li> <li>■ ALL: The ALL Level has the lowest possible rank and</li> </ul>

Element	Description
	<p>is intended to turn on all logging.</p> <ul style="list-style-type: none"> <li>■ OFF: The OFF Level has the highest possible rank and is intended to turn off all logging.</li> </ul>
<b>appender</b>	<p>Consumers to whom the alert messages must be sent. These can be specified using the class attribute, in which the fully qualified class name of the consumer must be defined, for example,</p> <pre data-bbox="678 608 1290 734">&lt;appender class='com.eibus.util.logger.appenders.ProcessNamedDailyRollingFileAppender' name='alertfile'&gt;</pre> <p>The alert system can be configured to various consumers such as File, E-Mail, Syslog, and Windows Event Log.</p>
<b>filter</b>	<p>Filters the alerts based on alert or message IDs and is specific to a particular appender. This must be specified inside the corresponding appender, for example,</p> <pre data-bbox="678 1020 1400 1083">&lt;filter class ='com.eibus.util.logger.filter.ResourceIDFilter'&gt;</pre>
<b>&lt;appender-ref&gt;</b>	<p>References another appender by its name. The ref attribute must match the name of appender declared elsewhere within an &lt;appender&gt; element.</p>

All alert and log messages are sent to a single file for processing. However, you can choose to publish alert messages to a different file. In such a case, the configuration must be done as follows:

```
<category name='com.eibus.management.AlertSystem' additivity='false'>
<priority value='info' /> <appender-ref ref=' alertfile' /> </category>
<appender
class='com.eibus.util.logger.appenders.ProcessNamedDailyRollingFileAppender'
name='alertfile'> <param name='File' value='Alert_%N.xml' /> <param
name='DatePattern' value=".yyyy-MM-dd" /> <layout
class='org.apache.log4j.xml.XMLLayout' /> </appender>
```

In this configuration, alerts get published to a file name with a prefix 'Alert\_'. Changes made in the configuration file are automatically updated to the alert system.

## **Filters for Alert System**

AppWorks Platform provides a set of filters or consumers that can be configured as specified below. Also, you can configure your own filters. Filters are set on alert messages to enrich or modify them. Alert System provides the following filters for AppWorks Platform:

- Message ID filter
- Severity filter

### **Message ID filter**

This filter ignores or allows alerts with IDs specified in the configuration file. For example, to allow only a set of alerts to pass through, the configuration must be as follows:

```
<category name="com.eibus.management.AlertSystem">

<priority value="info"/> <appender-ref ref="processNamefile"/>

</category>

<appender
class="com.eibus.util.logger.appender.ProcessNamedDailyRollingFileAppender"
name="processNamefile">

<param name="File" value="%N.xml"/>

<param name="DatePattern" value="&apos;.&apos;yyyy-MM-dd"/>

<layout class="org.apache.log4j.xml.XMLLayout"/>

<filter class="com.eibus.util.logger.ResourceIDFilter">

<param name="patterns" value="fully qualified message Bundle ID; fully qualified
message ID"/>

</filter>

</appender>
```

The alert messages for the Message IDs 'message Bundle ID1' and 'message ID1' are published. All other alert messages are ignored. An example of the fully qualified **messageID is:** Cordys.ESBServer.Messages.monitorStarted, where Cordys.ESBServer.Messages represents the fully qualified MessageBundleID and monitorStarted is for the actual MessageID. For more information on messageIDs see [Alert repository for AppWorks Platform](#).

However, you can also block (deny) a set of alert messages while allowing others to filter through. The configuration must be as follows:

```
<category name="com.eibus.management.AlertSystem"> <priority value="info"/> <appender-ref ref="processNamefile"/> <appender-ref ref="e-mail"/> </category> <appender class="com.eibus.util.logger.appenders.ProcessNamedDailyRollingFileAppender" name="processNamefile"> <param name="File" value="%N.xml"/> <param name="DatePattern" value="&apos; .&apos; yyyy-MM-dd"/> <layout class="org.apache.log4j.xml.XMLLayout"/> <filter class="com.eibus.util.logger.filter.ResourceIDFilter"> <param name="patterns" value="message ID1; message ID2"/> <param name="filterMode" value="DENY"/> </filter> </appender>
```

- In the `<param>` node , the value of the attribute `name` is either `filterMode` or `patterns`.
- The possible values for the `value` attribute in `filterMode` are `DENY`, `NEUTRAL`, or `ACCEPT`.
  - `DENY`: The log event is dropped immediately without consulting the remaining filters.
  - `NEUTRAL`: The next filter in the chain is consulted. If there are no more filters in the chain, the log event is logged. Therefore, in the presence of no filters, the default behavior is to log all logging events.
  - `ACCEPT`: The log event is logged without consulting the remaining filters.

## Severity filter

The purpose of the severity filter is to filter alerts that are less severe than the configured severity level. Each alert is assigned a severity level. These severity levels are predefined and cannot be changed by the user. Alerts can fall into one of the following levels of severity:

- Error
- Warn
- Info

The default severity level of an alert is `Info`, for example, `<priority value='info' />`.

## Sending alerts

SMTPAppender publishes alerts to the email specified in the configuration. See the following example.

1. Specify the following category and appender in `<AppWorks_Platform_installdir>/config/Log4jConfiguration.xml` and save it.

```

<category name="com.eibus.management.AlertSystem">
    <priority value="info" />
    <appender-ref ref="e-mail" />
</category>
<appender class="org.apache.log4j.net.SMTPAppender" name="e-mail">
    <param name="SMTPHost" value="srv-ind-dm19jw" />
    <param name="To" value="Admin@yourmail.com" />
    <param name="From" value="JMX@yourmail.com" />
    <param name="subject" value="AppWorks Platform Alert"/>
    <layout class="org.apache.log4j.HTMLLayout">
        <param name="LocationInfo" value="true" />
    </layout>
</appender>

```

2. Perform operations that raise alerts, for example, restarting the monitor, sending a license report, and restarting the database server.
3. Open the mailbox and check the alerts.

### Example of an error alert in email

The screenshot shows an email inbox with a single message from 'Cordys Alert'. The subject is 'tsesha@cordys.com'. The message content is a log entry starting at 'Log session start time Tue Jul 05 17:56:11 IST 2011'. The log table has columns: Time, Thread, Level, Category, File:Line, and Message. One entry shows:

Time	Thread	Level	Category	File:Line	Message
150687	DB_Connection_Monitor_Thread_1309868622164	ERROR	com.eibus.management.AlertSystem	LDAPWrapper.java:823	The directory server is not available. The directory server may not be functioning, or the network may be encountering server connectivity problems. Ensure that the directory server is functioning, and the network is communicating properly with the server.

### How to configure multiple appenders

To configure multiple appenders, the corresponding `<appender-ref>` element should be under `<category>`. See the following example:

```

<category name='com.eibus.management.AlertSystem'>
    <priority value='info' />
    <appender-ref ref='processNamefile' />
    <appender-ref ref='e-mail' />
</category>
<appender
class='com.eibus.util.logger.appenders.ProcessNamedDailyRollingFileAppender'
name='processNamefile'>
    <param name='File' value='%N.xml' />
    <param name='DatePattern' value=".yyyy-MM-dd" />
    <layout class='org.apache.log4j.xml.XMLLayout' />
</appender>
<appender class='org.apache.log4j.net.SMTPAppender' name='e-mail'>
    <param name='SMTPHost' value='ADDRESS-OF-SMTP-Host' />

```

```

<param name='To' value='DESTINATION@EMAIL' />
<param name='From' value='SENDER@EMAIL' />
<layout class='org.apache.log4j.xml.XMLLayout' />
</appender>

```

## How to send alerts to Windows Event Log

The NTEventLogAppender appender publishes alerts to the Windows event log. See the following example:

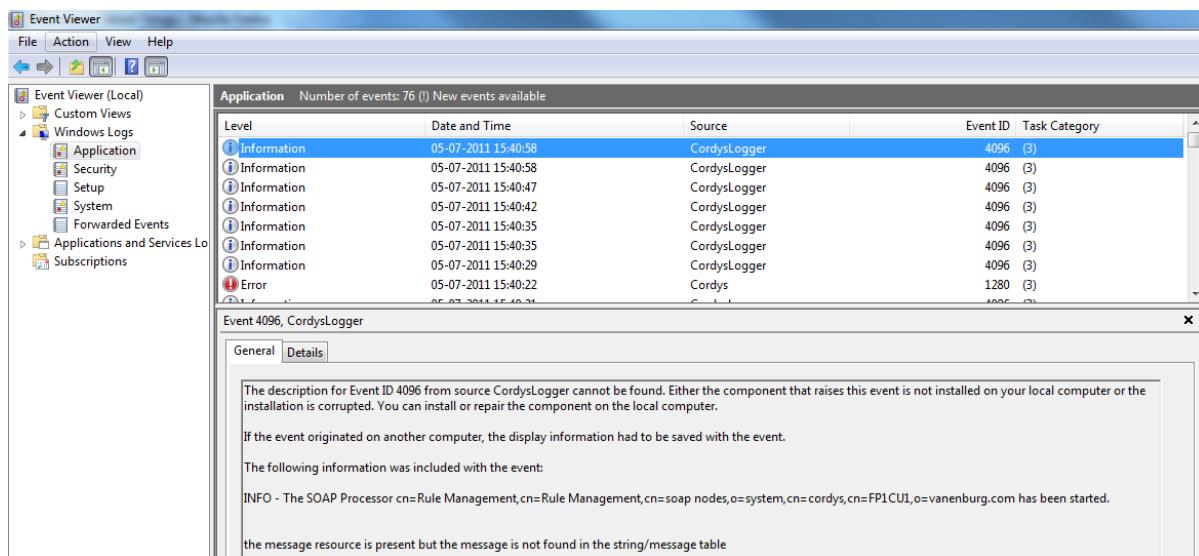
1. Specify the following category and appender in <AppWorks\_Platform\_installdir>/config/Log4jConfiguration.xml

```

<category name='com.eibus.management.AlertSystem'>
    <priority value='info' />
    <appender-ref ref='NTEventLogAppender' />
</category>
<appender xmlns='' class='org.apache.log4j.nt.NTEventLogAppender' name='NTEventLogAppender'>
    <layout class='org.apache.log4j.SimpleLayout' />
    <param name='source' value='CordysLogger' />
</appender>

```

2. Perform operations that raise alerts, for example, restarting the monitor, sending the license report, and restarting the database server.
3. Open **Windows Event Viewer** and check the alerts.



## How to send alerts to SysLog

The SyslogAppender appender publishes alerts to the Windows event log. See the following example:

- Specify the following category and appender in <AppWorks Platform\_installdir>/config/Log4jConfiguration.xml and save it.

```
<category name='com.eibus.management.AlertSystem'>
    <priority value='info' />
    <appender-ref ref='SyslogAppender' />
</category>
<appender xmlns='' class='org.apache.log4j.net.SyslogAppender' name='SyslogAppender'>
    <param name='syslogHost' value='<machinename>' />
    <param name='Facility' value='local6' />
    <layout class='org.apache.log4j.SimpleLayout' />
</appender>
```

- Perform operations that raise alerts, for example, restarting the monitor, sending the license report, and restarting the database server.
- Open **SysLog** and check alerts.

### Example of log messages in SysLog

The screenshot shows the 'Syslog Server' tool within the PacketTrap pt360 Tool Suite. The interface includes a toolbar with various monitoring and configuration icons, a menu bar with File, Edit, Tools, and Help, and a top navigation bar with social media links. The main window has tabs for Dashboard, Tools, Discovery, Admin, All Tools, and a selected 'Tools' tab. Below the tabs is a status bar showing 'Syslog Server (0)' and a message 'Follow us on Twitter'. The main content area is titled 'Syslog Server' with settings like port=514, maxdisplay=250, and logpath=C:\Users\tsesha\Documents\PacketTrap\Syslog. It features two tabs: 'Status' and 'Running Log'. The 'Status' tab shows a summary of running processes: 'Running.', '0 hours 3 mins 2 secs', and 'Targets (n/a)'. The 'Running Log' tab shows a detailed log of events starting at 16:13:44, listing numerous NFO-level messages about process launches and stops, such as 'NFO - Launching process cn=Combined\_OSPProcess2.cn=monitor@cn10660.cn=monitorsapnode@cn10660.cn...', 'NFO - The SOAP Processor cn=LDAP.cn=LDAP.cn=soap nodes.o=system.cn=cordys.cn=FPICU1.o=vanenburg.com...', and 'NFO - The members cn10660.34164.4ff have formed a new group.'

Status	Date/Time (detected)	Priority	Hostname (detected)	Message
Running.	Jul 05 04:16:04 PM	Local6.Info	10.192.69.88	NFO - Launching process cn=Combined_OSPProcess2.cn=monitor@cn10660.cn=monitorsapnode@cn10660.cn...
Running.	Jul 05 04:16:04 PM	Local6.Info	10.192.69.88	NFO - Launching process cn=User Management Service.cn=UserManagement.cn=soap nodes.o=system.cn=cordys...
Running.	Jul 05 04:16:04 PM	Local6.Info	10.192.69.88	NFO - Launching process cn=ExCL Application Connector.cn=EXCL Service.cn=soap nodes.o=system.cn=cordys.cr...
Running.	Jul 05 04:16:03 PM	Local6.Info	10.192.69.88	NFO - Launching process cn=Combined_OSPProcess1.cn=monitor@cn10660.cn=monitorsapnode@cn10660.cn...
Running.	Jul 05 04:16:03 PM	Local6.Info	10.192.69.88	NFO - Launching process cn=Email.cn=Email.cn=soap nodes.o=system.cn=cordys.cn=FPICU1.o=vanenburg.com...
Running.	Jul 05 04:15:52 PM	Local6.Info	10.192.69.88	NFO - The SOAP Processor cn=LDAP.cn=LDAP.cn=soap nodes.o=system.cn=cordys.cn=FPICU1.o=vanenburg.co...
Running.	Jul 05 04:15:50 PM	Local6.Info	10.192.69.88	NFO - Launching process cn=LDAP.cn=LDAP.cn=soap nodes.o=system.cn=cordys.cn=FPICU1.o=vanenburg.com...
Running.	Jul 05 04:15:48 PM	Local6.Info	10.192.69.88	NFO - The members cn10660.34164.4ff have formed a new group.
Running.	Jul 05 04:15:47 PM	Local6.Info	10.192.69.88	NFO - The SOAP Processor cn=LDAP.cn=LDAP.cn=soap nodes.o=system.cn=cordys.cn=FPICU1.o=vanenburg.co...
Running.	Jul 05 04:15:37 PM	Local6.Info	10.192.69.88	NFO - Stopping process cn=LDAP.cn=LDAP.cn=soap nodes.o=system.cn=cordys.cn=FPICU1.o=vanenburg.com...
Running.	Jul 05 04:15:37 PM	Local6.Info	10.192.69.88	NFO - The SOAP Processor cn=Combined_OSPProcess1.cn=monitor@cn10660.cn=monitorsapnode@cn10660.cn...
Running.	Jul 05 04:15:34 PM	Local6.Info	10.192.69.88	NFO - The SOAP Processor cn=Combined_OSPProcess1.cn=monitor@cn10660.cn=monitorsapnode@cn10660.cn...
Running.	Jul 05 04:15:34 PM	Local6.Info	10.192.69.88	NFO - Stopping process cn=Combined_OSPProcess1.cn=monitor@cn10660.cn=monitorsapnode@cn10660.cn...
Running.	Jul 05 04:15:33 PM	Local6.Info	10.192.69.88	NFO - The SOAP Processor cn=MAXDB.cn=MAXDBService.cn=soap nodes.o=system.cn=cordys.cn=FPICU1.o=v...
Running.	Jul 05 04:15:33 PM	Local6.Info	10.192.69.88	NFO - The SOAP Processor cn=ExCL Application Connector.cn=EXCL Service.cn=soap nodes.o=system.cn=cordys...
Running.	Jul 05 04:15:33 PM	Local6.Info	10.192.69.88	NFO - The SOAP Processor cn=SNMP Scan.cn=SNMP Walk.cn=Switch Port Mapper.cn=Syslog.cn=TFTP Server...

### How to send alerts to the file system

The File appender is the default consumer type built into the alert system. By default, this appender writes alerts to the log file at <AppWorks Platform\_installdir>\<instance>\Logs.

- ProcessNamedDaily RollingFileAppender - This appender creates a daily rolling file for each process. This is the default consumer supported by the alert system.

```
<category name='com.eibus.management.AlertSystem'>
    <priority value='info' />
```

```

<appender-ref ref='processNamefile' />
</category>
<appender
class='com.eibus.util.logger.appenders.ProcessNamedDailyRollingFileAppender'
name='processNamefile'>
    <param name='File' value='%N.xml' />
    <param name='DatePattern' value=".yyyy-MM-dd" />
    <layout class='org.apache.log4j.xml.XMLLayout' />
</appender>

```

**Note:** The naming convention for pattern is `%N.xml`. In the `<AppWorks Platform_installdir>\Logs` folder, a file is created for each process with the name as process name. This file contains both alert and log messages. If required, the alert messages can be published to a different file. The sample configuration is then as follows:

```

<category name='com.eibus.management.AlertSystem' additivity='false'>
    <priority value='info' />
    <appender-ref ref='alertfile' />
</category>
<appender
class='com.eibus.util.logger.appenders.ProcessNamedDailyRollingFileAppender'
name='alertfile'>
    <param name='File' value='Alert%N.xml' />
    <param name='DatePattern' value=".yyyy-MM-dd" />
    <layout class='org.apache.log4j.xml.XMLLayout' />
</appender>

```

- OSBasedEventAppender - This is a combination of NTEventLogAppender and SysLog Appender. More information about this appender can be found in the later sections of this topic. A sample configuration of this appender is as follows:

```

<appender class="com.eibus.util.logger.appenders.OSBasedEventAppender"
name="OSBasedEventAppender">
    <layout class="com.cordys.logger.internal.XMLLayout">
        <param name="LocationInfo" value="true" />
    </layout>
</appender>

```

- DB Appender - This appender writes alerts to a database specified in the configuration. The sample configuration of this appender is as follows:

```

<appender class="com.cordys.logger.appenders.CordysZeroConfJDBCAppender"
name="dbappender"/>
<category name="com.eibus.management.AlertSystem">
    <priority value="info" />

```

```

<appender-ref ref="OSBasedEventAppender" />
</category>
<category name="com.eibus.management.ManagedComponent">
    <priority value="info" />
</category>
<root>
    <priority value="error" />
    <appender-ref ref="file" />
    <appender-ref ref="dbappender" />
</root>

```

You can also configure few default appenders provided by Log4j, which are as follows:

- **File Appender** - This appender writes alerts to a file specified in the configuration. The sample configuration is as follows:

```

<category name='com.eibus.management.AlertSystem'>
<priority value='info' />
<appender-ref ref='FileAppender' />
</category>
<appender xmlns='' class='org.apache.log4j.FileAppender' name='FileAppender'>
<param name='File' value='<AppWorks Platform_installdir>\logs\Alerts.xml' />
<layout class='org.apache.log4j.xml.XMLLayout'>
<param name='locationInfo' value='true' />
</layout>
</appender>

```

- **Daily Rolling File Appender** - This appender creates a daily rolling file based on the date pattern. The sample configuration is as follows:

```

<category name='com.eibus.management.AlertSystem'>
<priority value='info' />
<appender-ref ref='FileAppender' />
</category><appender xmlns='' class='org.apache.log4j.FileAppender' name='FileAppender'>
<param name='File' value='<AppWorks Platform_installdir>\logs\Alerts.xml' />
<layout class='org.apache.log4j.xml.XMLLayout'>
<param name='locationInfo' value='true' />
</layout></appender>

```

- **Rolling File Appender** - This appender creates a rolling file based on the size. The sample configuration is as follows:

```

<category name='com.eibus.management.AlertSystem'>
<priority value='info' />
<appender-ref ref='RollingFileAppender' />
</category>
<appender xmlns='' class='org.apache.log4j.RollingFileAppender'
name='RollingFileAppender'>
<param name='File' value='<AppWorks Platform_installdir>\logs\Alerts.xml' />
<param name='MaxFileSize' value='10MB' />
<param name='MaxBackupIndex' value='5' />
<layout class='org.apache.log4j.xml.XMLLayout'>
<param name='locationInfo' value='true' />
</layout>
</appender>

```

## Categories and severity levels

### Categories

AppWorks Platform provides a classification of error messages that enables the developer to categorize the error messages and assign an appropriate security level to the same. AppWorks Platform ships with predefined categories that are used for internal purposes. Developers of application connectors can define their custom categories.

When developing custom categories, do the following:

- Mention the connector name in the category name
- For the Log4J categories, apply a naming scheme similar to Java classes and packages.

For monitoring purposes, all relevant alerts are documented. See [Alert repository for AppWorks Platform](#) and [Understanding the alert system](#).

AppWorks Platform provides the following default categories.

Category	Description	Package name
ACL	<p>Messages on events related to ACL evaluation are logged. These events include:</p> <ul style="list-style-type: none"> <li>■ reading access control information for a specific user</li> <li>■ evaluating whether the user has access to a</li> </ul>	com.eibus.security.acl

<b>Category</b>	<b>Description</b>	<b>Package name</b>
	specific resource	
License	Messages on events related to license validation are logged.	com.eibus.license
LDAP	Messages on events related to LDAP reads, updates, or search are logged.	com.eibus.directory
SOAP Framework	Messages on events related to processing of the SOAP request by the service are logged.	com.eibus.transport.SOAPMessage (Class Name)
SOAP Message	Messages on issues related to SOAP messages, which are sent to the service are logged.	com.eibus.soap
Transport	Messages on events related to the transport layer are logged. Any event that occurs during message delivery by transport providers such as Socket or JMS can be logged.	com.eibus.transport
Tools	Messages on events that are performed in the tools available with the AppWorks Platform installation, for example, Simple Client or Management Console are logged.	com.eibus.tools
Util	While developing applications on top of AppWorks Platform, developers can use some utility classes available with the AppWorks Platform installation. Messages that must be	com.eibus.util

<b>Category</b>	<b>Description</b>	<b>Package name</b>
	<p>logged in these utility classes are logged.</p> <p><b>Note:</b> OpenText recommends not logging messages in these utility classes. Instead, the classes developed using these utility classes must log messages.</p>	

## **Severity levels**

You can assign severity levels to the categories when configuring the service container. Each log category falls into a defined severity level. Severity levels help you filter the log message for each category. For example, the severity level for the `com.eibus.security.acl` category can be Fatal, while that for the `com.eibus.license` category can be Warn.

The following severity levels are defined for each category:

<b>Severity level</b>	<b>Description</b>
Fatal	Log messages for critical errors, service container being stopped, and application crashing.
Debug	Log messages for developers and support personnel for debugging the application and detecting the cause for the error.
Warn	Log messages for errors caused by incorrect user inputs and ignorable errors.
Error	Log messages for exceptions and errors.
Info	Log messages for the operators. For each application connector, try to log requests and responses with severity information.

### **Note:**

- The framework supports severity levels higher than what is configured. For example, if the severity level is configured to Warn, the logger logs all the severities that are higher than Warn.
- If the severity level of type Info is enabled for each application connector, the request and response is logged in that context.

## Appenders for an alert system

AppWorks Platform provides a set of appenders or consumers that can be configured for several purposes. Besides the ones provided in AppWorks Platform, you can also configure other appenders and consumers as needed.

Alerts are routed to various appenders through the `<appender>` element. The configuration for the following types of appenders is given below:

- [File](#)
- [Windows event log](#)
- [SysLog consumer](#)
- [Socket appender](#)
- [Email](#)

### File

File appender is the default consumer type built into the alert system. By default, this appender writes alerts in the log file `<AppWorks_Platform_installdir>\Logs`.

For each process, a log file with the alert and log messages is created. However, a file appender can be configured explicitly and the file name can be specified within the name and value attributes. For example, `system#xml_store_service#xml_store_processor.xml` is the name of the file in which spaces are replaced by underscore. This is the naming convention of the file.

Process	Name of the log
Service	<code>&lt;Organization Name&gt;#&lt;Service Group&gt;#&lt;Service Container&gt;.xml</code>
Monitor	<code>Monitor.xml</code>
Gateway	<code>Gateway.xml</code>

### Types of appenders

- `ProcessNamedDailyRollingFileAppender` is the default consumer supported by the Alert system that creates a daily rolling file for each process.

```
<category name='com.eibus.management.AlertSystem'>
    <priority value='info' />
    <appender-ref ref='processNamefile' />
</category>
<appender
    class='com.eibus.util.logger.appenders.ProcessNamedDailyRollingFileAppender'
    name='processNamefile'>
    <param name='File' value='%N.xml' />
    <param name='DatePattern' value="'.yyyy-MM-dd" />
    <layout class='org.apache.log4j.xml.XMLLayout' />
</appender>
```

---

The configuration is as follows:

```
<category name='com.eibus.management.AlertSystem' additivity='false'>
    <priority value='info' />
    <appender-ref ref='alertfile' />
</category>
<appender
class='com.eibus.util.logger.appenders.ProcessNamedDailyRollingFileAppender'
name='alertfile'>
    <param name='File' value='Alert%N.xml' />
    <param name='DatePattern' value="'.yyyy-MM-dd" />
    <layout class='org.apache.log4j.xml.XMLLayout' />
</appender>
```

- OSBasedEventAppender is a combination of NTEventLogAppender and SysLog Appender. The configuration of the appender is as follows:

```
<category name='com.eibus.management.AlertSystem'>
    <priority value='info' />
    <appender-ref ref='OSBasedEventAppender' />
</category>
<appender name="OSBasedEventAppender">
    <layout>
        <param name="LocationInfo" value="true" />
    </layout>
</appender>
```

- DB Appender writes alerts in the database specified in the configuration. The configuration of the appender is as follows:

```
<appender class="com.cordys.logger.appenders.CordysZeroConfJDBCAppender"
name="dbappender"/>
<category name="com.eibus.management.AlertSystem">
    <priority value="info" />
    <appender-ref ref="OSBasedEventAppender" />
</category>
<category name="com.eibus.management.ManagedComponent">
    <priority value="info" />
</category>
<root>
    <priority value="error" />
    <appender-ref ref="file" />
    <appender-ref ref="dbappender" />
</root>
```

The configuration of the default appenders in Log4j is as follows:

- FileAppender writes the alerts in the file specified in the configuration. The configuration is as follows:

```
<category name='com.eibus.management.AlertSystem'>
    <priority value='info' />
    <appender-ref ref='FileAppender' />
</category>
<appender xmlns='' class='org.apache.log4j.FileAppender' name='FileAppender'>
    <param name='File' value='<AppWorks_Platform_installdir>\logs\Alerts.xml' />
    <layout class='org.apache.log4j.xml.XMLLayout'>
```

---

```

        <param name='locationInfo' value='true' />
    </layout>
</appender>
```

- DailyRollingFileAppender creates a daily rolling file based on the date pattern. The configuration is as follows:

```

<category name='com.eibus.management.AlertSystem'>
    <priority value='info' />
    <appender-ref ref='DailyRollingFileAppender' />
</category>
<appender xmlns='' class='org.apache.log4j.DailyRollingFileAppender'
name='DailyRollingFileAppender'>
    <param name='File' value='<AppWorks Platform_installdir>\logs\Alerts.xml' />
    <param name='DatePattern' value=''.yyyy-MM-dd' />
    <layout class='org.apache.log4j.xml.XMLLayout'>
        <param name='locationInfo' value='true' />
    </layout>
</appender>
```

- RollingFileAppender creates a rolling file based on the size. The configuration is as follows:

```

<category name='com.eibus.management.AlertSystem'>
    <priority value='info' />
    <appender-ref ref='RollingFileAppender' />
</category>
<appender xmlns='' class='org.apache.log4j.RollingFileAppender'
name='RollingFileAppender'>
    <param name='File' value='<AppWorks Platform_installdir>\logs\Alerts.xml' />
    <param name='MaxFileSize' value='10MB' />
    <param name='MaxBackupIndex' value='5' />
    <layout class='org.apache.log4j.xml.XMLLayout'>
        <param name='locationInfo' value='true' />
    </layout>
</appender>
```

## **Windows event log**

NTEventLogAppender publishes alerts to Windows event log. The configuration details are as follows:

```

<category name='com.eibus.management.AlertSystem'>
    <priority value='info' />
    <appender-ref ref='NTEventLogAppender' />
</category>
<appender xmlns='' class='org.apache.log4j.nt.NTEventLogAppender'
name='NTEventLogAppender'>
    <layout class='org.apache.log4j.SimpleLayout' />
    <param name='source' value='CordysLogger' />
</appender>
```

For a detailed procedure, see [Sending alerts to the Windows Event Log](#).

## **SysLog consumer**

A SysLog appender is the default consumer configured for the Linux Operating System. It can also be used on Windows Operating System by modifying the setting <syslogserver> to the system in which the syslogd service is running. A SysLog appender can be configured by specifying the following settings in the configuration file:

```
<category name='com.eibus.management.AlertSystem'>
    <priority value='info' />
    <appender-ref ref='SyslogAppender' />
</category>
<appender xmlns='' class='org.apache.log4j.net.SyslogAppender'
name='SyslogAppender'>
    <param name='syslogHost' value='<machinename>' />
    <param name='Facility' value='local6' />
    <layout class='org.apache.log4j.SimpleLayout' />
</appender>
```

For a detailed procedure, see [Sending alerts to SysLog](#) A SysLog appender is the default consumer configured for the Linux Operating System. It can also be used on the Windows Operating System by modifying the <syslogserver> on the system that is running the syslogd service. You can set up SysLog to view log messages or alerts. The following example shows how to configure the SyslogAppender that publishes alerts to SysLog. Before making the changes to the Log4jConfiguration.xml file, make sure the content is valid XML. Any changes are applied immediately. To add a logging category and appender: Open Log4jConfiguration.xml from AppWorks\_Platform\_installdir/config. Add the following category. <category name="com.eibus.management.AlertSystem"> <priority value="info"/> <appender-ref ref="SyslogAppender"/> </category> Add the Appender for SyslogAppender. <appender xmlns="" class='org.apache.log4j.net.SyslogAppender' name='SyslogAppender'> <param name='syslogHost' value='<machinename>' /> <param name='Facility' value='local6' /> <layout class='org.apache.log4j.SimpleLayout' /> </appender> Save the configuration. To view log entries: Perform any operation that raises an alert. For example, restarting the Monitor, sending the license report, or restarting the database server. Open SysLog and check the alerts. The syslogd DaemonTo log messages to Syslog consumer: Start the syslogd Daemon with remote logging set to enabled. To stop the syslogd service: Type service syslog stop in the Linux prompt. To start with remote logging as enabled: Type syslogd -r -l host name -f /etc/syslog.conf in the Linux prompt. .

## **Socket appender**

A socket appender publishes alerts to the socket. The configuration is as follows:

```
<category name='com.eibus.management.AlertSystem'>
    <priority value='info' />
    <appender-ref ref='SocketAppender' />
</category>
<appender xmlns='' class='org.apache.log4j.net.SocketAppender'
name='SocketAppender'>
    <param name='Port' value='4445' />
    <param name='RemoteHost' value='<machinename>' />
```

---

```

<layout class='org.apache.log4j.xml.XMLLayout'>
    <param name='locationInfo' value='true' />
</layout>
</appender>

```

## Email

An email consumer publishes alerts to the mail ID specified in the configuration file. While configuring an email consumer ensure that:

- Port 25 is enabled for the functioning of email appender and other services which are running on the port are disabled.
- The `mail.jar` file (in `<Process_Platform_installdir>/ext`) is placed in the classpath of the service container / web server.
- Multiple addresses specified in the To parameter are separated by a coma.

The configuration of the email consumer is:

```

<category name="com.eibus.management.AlertSystem">
    <priority value="info" />
    <appender-ref ref="e-mail" />
</category>
<appender class="org.apache.log4j.net.SMTPAppender" name="e-mail">
    <param name="SMTPHost" value="ADDRESS-OF-SMTP-Host" />
    <param name="To" value="DESTINATION@EMAIL" />
    <param name="From" value="SENDER@EMAIL" />
    <param name="EvaluatorClass" value="com.eibus.util.logger.TriggerEvent"/>
    <param name="subject" value="AppWorks Platform Alert"/>
    <layout class="org.apache.log4j.HTMLLayout">
        <param name="LocationInfo" value="true" />
    </layout>
</appender>

```

To view an example on configuring an alert, see [Sending alerts to email](#).

Multiple appenders can be configured simultaneously by specifying the corresponding `<appender-ref>` under `<category>` element.

The sample code for the configuration of multiple appenders is:

```

<category name='com.eibus.management.AlertSystem'>
    <priority value='info' />
    <appender-ref ref='processNamefile' />
    <appender-ref ref='e-mail' />
</category>
<appender
class='com.eibus.util.logger.appenders.ProcessNamedDailyRollingFileAppender'
name='processNamefile'>
    <param name='File' value='%N.xml' />
    <param name='DatePattern' value=".yyyy-MM-dd" />
    <layout class='org.apache.log4j.xml.XMLLayout' />
</appender>
<appender class='org.apache.log4j.net.SMTPAppender' name='e-mail'>
    <param name='SMTPHost' value='ADDRESS-OF-SMTP-Host' />
    <param name='To' value='DESTINATION@EMAIL' />
    <param name='From' value='SENDER@EMAIL' />

```

```

<layout class='org.apache.log4j.xml.XMLLayout' />
</appender>

```

For a detailed procedure, see [Configuring multiple appenders](#).

## **Sending alerts to the Windows Event Log**

You can set up the Windows Event Log to view log messages or alerts. The following example shows how to configure `NTEventLogAppender` to publish alerts to the Windows Event Log.

**Important:** Before making the changes to the `Log4jConfiguration.xml` file, make sure the content is valid XML. Any changes are applied immediately.

### **To add a logging category and appender:**

1. Open `Log4jConfiguration.xml` from `AppWorks_Platform_installdir/config`.
2. Add the following category.

```

<category name="com.eibus.management.AlertSystem">
    <priority value="info"/>
    <appender-ref ref="NTEventLogAppender"/>
</category>

```

3. Add the Appender for `NTEventLogAppender`.

```

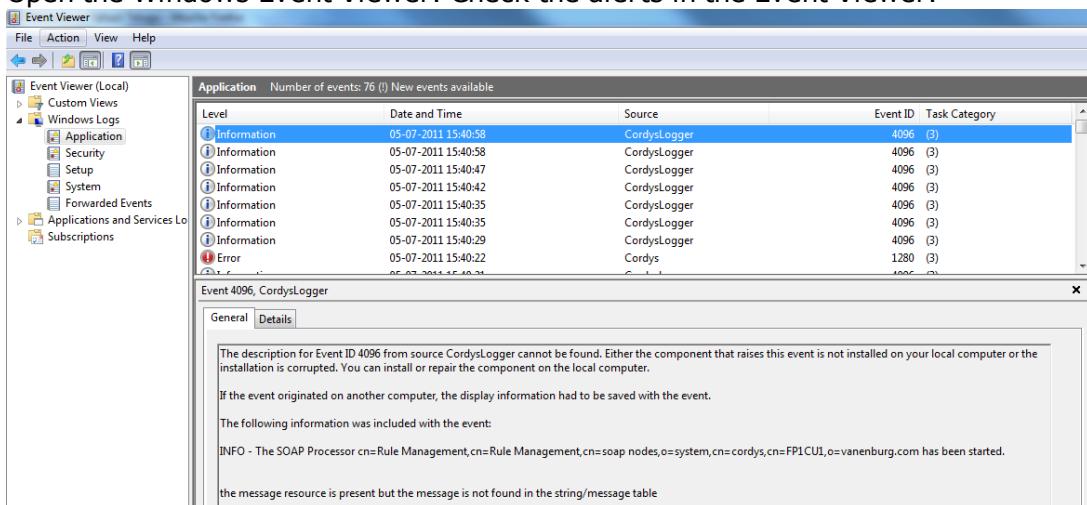
<appender class="org.apache.log4j.nt.NTEventLogAppender" name="NTEventLogAppender"
xmlns="">
    <layout class="org.apache.log4j.SimpleLayout"/>
    <param name="source" value="CordysLogger"/>
</appender>

```

4. Save the configuration.

### **To view log entries:**

1. Perform any operation that raises an alert, For example, restarting the Monitor, sending the license report, or restarting the database server.
2. Open the Windows Event Viewer. Check the alerts in the Event Viewer.



## **Sending alerts to SysLog**

A SysLog appender is the default consumer configured for the Linux Operating System. It can also be used on the Windows Operating System by modifying the <syslogserver> on the system that is running the syslogd service.

You can set up SysLog to view log messages or alerts. The following example shows how to configure the SyslogAppender that publishes alerts to SysLog.

**Important:** Before making the changes to the `Log4jConfiguration.xml` file, make sure the content is valid XML. Any changes are applied immediately.

### **To add a logging category and appender:**

1. Open `Log4jConfiguration.xml` from `AppWorks_Platform_installdir/config`.
2. Add the following category.

```
<category name="com.eibus.management.AlertSystem">
    <priority value="info"/>
    <appender-ref ref="SyslogAppender"/>
</category>
```

3. Add the Appender for SyslogAppender.

```
<appender xmlns='' class='org.apache.log4j.net.SyslogAppender'
name='SyslogAppender'>
    <param name='syslogHost' value='<machinename>' />
    <param name='Facility' value='local6' />
    <layout class='org.apache.log4j.SimpleLayout' />
</appender>
```

4. Save the configuration.

### **To view log entries:**

1. Perform any operation that raises an alert. For example, restarting the Monitor, sending the license report, or restarting the database server.
2. Open SysLog and check the alerts.

## **The syslogd Daemon**

### **To log messages to Syslog consumer:**

- Start the syslogd Daemon with remote logging set to **enabled**.

### **To stop the syslogd service:**

- Type `service syslog stop` in the Linux prompt.

---

## To start with remote logging as enabled:

- Type `syslogd -r -l host name -f /etc/syslog.conf` in the Linux prompt.

## ***Sending alerts to email***

An email consumer publishes alerts to the mail ID specified in the configuration file. While configuring an email consumer ensure that:

- Port 25 is enabled for the functioning of email appender and other services which are running on the port are disabled.
- The `mail.jar` file (in `<AppWorks_Platform_installdir>/ext`) is placed in the classpath of the service container / web server.
- Multiple addresses specified in the To parameter are separated by a comma.

The following example shows how to configure the SMTPAppender that publishes log messages and alerts to email:

**Important:** Before making the changes to the `Log4jConfiguration.xml` file, ensure that the content is valid XML. Any changes are applied immediately.

## To add a logging category and appender:

1. Open `Log4jConfiguration.xml` from `AppWorks_Platform_installdir/config`.
2. Add the following category:

```
<category name="com.eibus.management.AlertSystem">
    <priority value="info"/>
    <appender-ref ref="e-mail"/>
</category>
```

3. Add the Appender for email:

```
<appender class="org.apache.log4j.net.SMTPAppender" name="e-mail">
    <param name="SMTPHost" value="srv-ind-dm19jw"/>
    <param name="To" value="Admin@yourmail.com"/>
    <param name="From" value="JMX@yourmail.com"/>
    <param name="subject" value="AppWorks Platform Alert"/>
    <param name="EvaluatorClass" value="com.eibus.util.logger.TriggerEvent"/>
    <layout class="org.apache.log4j.HTMLLayout">
        <param name="LocationInfo" value="true"/>
    </layout>
</appender>
```

4. Save the configuration.

---

## To view log entries:

1. Perform any operation that raises an alert. For example, restarting the Monitor, sending the license report, or restarting the database server.
2. Check the mail box for the alert. 

## Configuring multiple appenders

Multiple appenders can be configured by specifying the corresponding <appender-ref> element under <category>. The following example illustrates the configuration of multiple appenders:

**Important:** Before making the changes to the `Log4jConfiguration.xml` file, ensure that the content is valid XML. Any changes are applied immediately.

### To add a logging category:

1. Open `Log4jConfiguration.xml` in `AppWorks_Platform_installdir/config`.
2. Add the following category which contains multiple appenders (`processNamefile` and `e-mail`).

```
<category name="com.eibus.management.AlertSystem">
    <priority value="info"/>
    <appender-ref ref="processNamefile"/>
    <appender-ref ref="e-mail"/>
</category>
```

3. Add the Appenders for `processNamefile` and `e-mail`.

```
<appender
    class='com.eibus.util.logger.appenders.ProcessNamedDailyRollingFileAppender'
    name='processNamefile'>
    <param name='File' value=''%N.xml' />
    <param name='DatePattern' value=".yyyy-MM-dd" />
    <layout class='org.apache.log4j.xml.XMLLayout' />
</appender>
<appender class='org.apache.log4j.net.SMTPAppender' name='e-mail'>
    <param name='SMTPHost' value='ADDRESS-OF-SMTP-Host' />
    <param name='To' value='DESTINATION@EMAIL' />
    <param name='From' value='SENDER@EMAIL' />
    <layout class='org.apache.log4j.xml.XMLLayout' />
</appender>
```

4. Save the configuration.

## To view log entries:

1. Perform any operation that raises the alerts, For example, restarting the Monitor, sending a license report, or restarting the database server.

- 
2. Alerts are displayed in AppWorks\_Platform\_installdir/Logs and can be received as emails.

## **Sending alerts to the file system**

AppWorks Platform provides a set of appenders or consumers that can be configured for several purposes. Apart from the appenders provided in AppWorks Platform, you can configure other appenders and consumers as per the requirement. Alerts are routed to various appenders through the `<appender>` element. The configuration for each of these appenders has been provided below.

The following are the types of appenders:

- File
- Windows Event Log
- SysLog Consumer
- Socket
- e-mail

File appender is the default consumer type built into the alert system. By default, this appender writes alerts to the log file at `<AppWorks_Platform_installdir>\<instance>\Logs`.

For each process, a log file with the alert and log messages is created. However, a file appender can be configured explicitly and the file name can be specified within the `name` and `value` attributes. For example, `system#xml_store_service#xml_store_processor.xml` is the name of the file in which spaces are replaced by underscore. This is the naming convention of the file.

<b>Process</b>	<b>Name of the log</b>
Service	<code>&lt;Organization Name&gt;#&lt;Service Group&gt;#&lt;Service Container&gt;.xml</code>
Monitor	<code>Monitor.xml</code>
Gateway	<code>Gateway.xml</code>

## **Types of Appenders**

- **ProcessNamedDaily RollingFileAppender** - This appender creates a daily rolling file for each process. This is the default consumer supported by the Alert System.

```
<category name='com.eibus.management.AlertSystem'>
  <priority value='info' />
  <appender-ref ref='processNamefile' />
```

```

</category>
<appender
  class='com.eibus.util.logger.appenders.ProcessNamedDailyRollingFileAppender'
  name='processNamefile'>
  <param name='File' value='%N.xml' />
  <param name='DatePattern' value="'.yyyy-MM-dd" />
  <layout class='org.apache.log4j.xml.XMLLayout' />
</appender>

```

**Note:** The naming convention for pattern is '%N.xml'. A file is created for each process with the name as process name, located at <AppWorks Platform\_installdir>\Logs. This file contains both the alert as well as log messages. If required, the alert messages can be published to a different file. The sample configuration is:

```

<category name='com.eibus.management.AlertSystem' additivity='false'>
  <priority value='info' />
  <appender-ref ref='alertfile' />
</category>
<appender
  class='com.eibus.util.logger.appenders.ProcessNamedDailyRollingFileAppender'
  name='alertfile'>
  <param name='File' value='Alert%N.xml' />
  <param name='DatePattern' value="'.yyyy-MM-dd" />
  <layout class='org.apache.log4j.xml.XMLLayout' />
</appender>

```

- **OSBasedEventAppender**- This is a combination of NTEventLogAppender and SysLog Appender. More information about them can be found in the later sections of this topic.

Sample configuration of this appender:

```

<category name='com.eibus.management.AlertSystem'>
  <priority value='info' />
  <appender-ref ref='OSBasedEventAppender' />
</category>
<appender name="OSBasedEventAppender">
  <layout>
    <param name="LocationInfo" value="true" />
  </layout>
</appender>

```

- **DB Appender** - This appender writes alerts to a database specified in the configuration.

Sample configuration of this appender is as follows:

```

<appender class="com.cordys.logger.appenders.CordysZeroConfJDBCAppender"
  name="dbappender"/>

```

```

<category name="com.eibus.management.AlertSystem">
    <priority value="info" />
    <appender-ref ref="OSBasedEventAppender" />
</category>
<category name="com.eibus.management.ManagedComponent">
    <priority value="info" />
</category>
<root>
    <priority value="error" />
    <appender-ref ref="file" />
    <appender-ref ref="dbappender" />
</root>

```

## Log4j Appenders

You can also configure default appenders provided by Log4j. They are:

- **File Appender** - This appender writes alerts to a file specified in the configuration. The sample configuration is:

```

<category name='com.eibus.management.AlertSystem'>
    <priority value='info' />
    <appender-ref ref='FileAppender' />
</category>
<appender xmlns='' class='org.apache.log4j.FileAppender' name='FileAppender'>
    <param name='File' value='<AppWorks Platform_installdir>\logs\Alerts.xml' />
    <layout class='org.apache.log4j.xml.XMLLayout'>
        <param name='locationInfo' value='true' />
    </layout>
</appender>

```

- **Daily Rolling File Appender** - This appender creates a daily rolling file based on the date pattern. The sample configuration is:

```

<category name='com.eibus.management.AlertSystem'>
    <priority value='info' />
    <appender-ref ref='DailyRollingFileAppender' />
</category>
<appender xmlns='' class='org.apache.log4j.DailyRollingFileAppender' name='DailyRollingFileAppender'>
    <param name='File' value='<AppWorks Platform_installdir>\logs\Alerts.xml' />
    <param name='DatePattern' value=''.yyyy-MM-dd' />
    <layout class='org.apache.log4j.xml.XMLLayout'>
        <param name='locationInfo' value='true' />
    </layout>
</appender>

```

- **Rolling File Appender** - This appender creates a rolling file based on the size. The sample configuration is:

```

<category name='com.eibus.management.AlertSystem'>
    <priority value='info' />
    <appender-ref ref='RollingFileAppender' />
</category>
<appender xmlns='' class='org.apache.log4j.RollingFileAppender'
name='RollingFileAppender'>
    <param name='File' value='<AppWorks Platform_installdir>\logs\Alerts.xml' />
    <param name='MaxFileSize' value='10MB' />
    <param name='MaxBackupIndex' value='5' />
    <layout class='org.apache.log4j.xml.XMLLayout'>
        <param name='locationInfo' value='true' />
    </layout>
</appender>

```

## **Sending alerts to a socket**

A socket appender publishes alerts to the socket. The configuration is as follows:

```

<category name='com.eibus.management.AlertSystem'>
    <priority value='info' />
    <appender-ref ref='SocketAppender' />
</category>
<appender xmlns='' class='org.apache.log4j.net.SocketAppender'
name='SocketAppender'>
    <param name='Port' value='4445' />
    <param name='RemoteHost' value='<machinename>' />
    <layout class='org.apache.log4j.xml.XMLLayout'>
        <param name='locationInfo' value='true' />
    </layout>
</appender>

```

## **Managing logging framework**

The logging framework is a key feature of AppWorks Platform that records certain critical activities during runtime. It is based on Log4j and is used for debugging.

The information gathered from the log files is used to analyze the error state and fix support calls. AppWorks Platform supports different kinds of log consumers such as file, and event service, including those from Log4j. This framework provides a web-based log viewer for viewing the messages. Apache's Chainsaw is an open source GUI-based tool that can also be used for viewing the log messages. [Using composite application logging \(CAL\)](#) is an extension of Logging Framework which facilitates application level logging.

The logging framework helps the support personnel or developers or administrators to analyze error states. See [Logging framework process](#) for information on the overall working of logging framework.

[Log consumer](#) uses these log messages. AppWorks Platform supports various kinds of log consumers, including those from Log4j, for consuming log messages.

---

**Note:** By default, the internal logging mechanism of log4j named LogLog is disabled. To enable it, set `com.cordys.management.log4j.internallogging` to `true` in the Platform Properties.

AppWorks Platform provides a classification of error messages. This enables you to categorize the error messages and assign an appropriate severity level. See [Categories and severity levels](#).

This section contains the following topics:

- [Providing log configuration details](#): Describes how to provide log configuration details.
- [Viewing log messages](#): Explains how to use the log viewer.

## **Examples of log messages**

- License key has been validated.
- Request has been received.
- Response has been sent.
- An exception occurred.

### **Log message format**

The output log messages are in the following format:

```
<LogMessage>
  <category>java</category>
  <severity>ERROR</severity>
  <date>1116933270643</date>
  <thread>Dispatcher-2/Worker-0</thread>
  <host>Machine</host>
  <messageid>{DBF49C4B-73B2-4EDA-9105-E71ABD7E9A55}</messageid>
  <user>cn=Steve,cn=organizational users,o=system,cn=cordys,o=vanenburg.com</user>
  <component>cn=FTP Service,cn=soap
nodes,o=system,cn=cordys,o=vanenburg.com</component>
  <message>This is an error.</message>
  <class>com.eibus.applicationconnector.java.Transaction</class>
  <method>process</method>
  <file>Transaction.java</file>
  <line>63</line>
</LogMessage>
```

### **Log message format for exceptions**

For exceptions, the log message format is:

```
<LogMessage>
  <category>ldap</category>
  <severity>ERROR</severity>
```

```

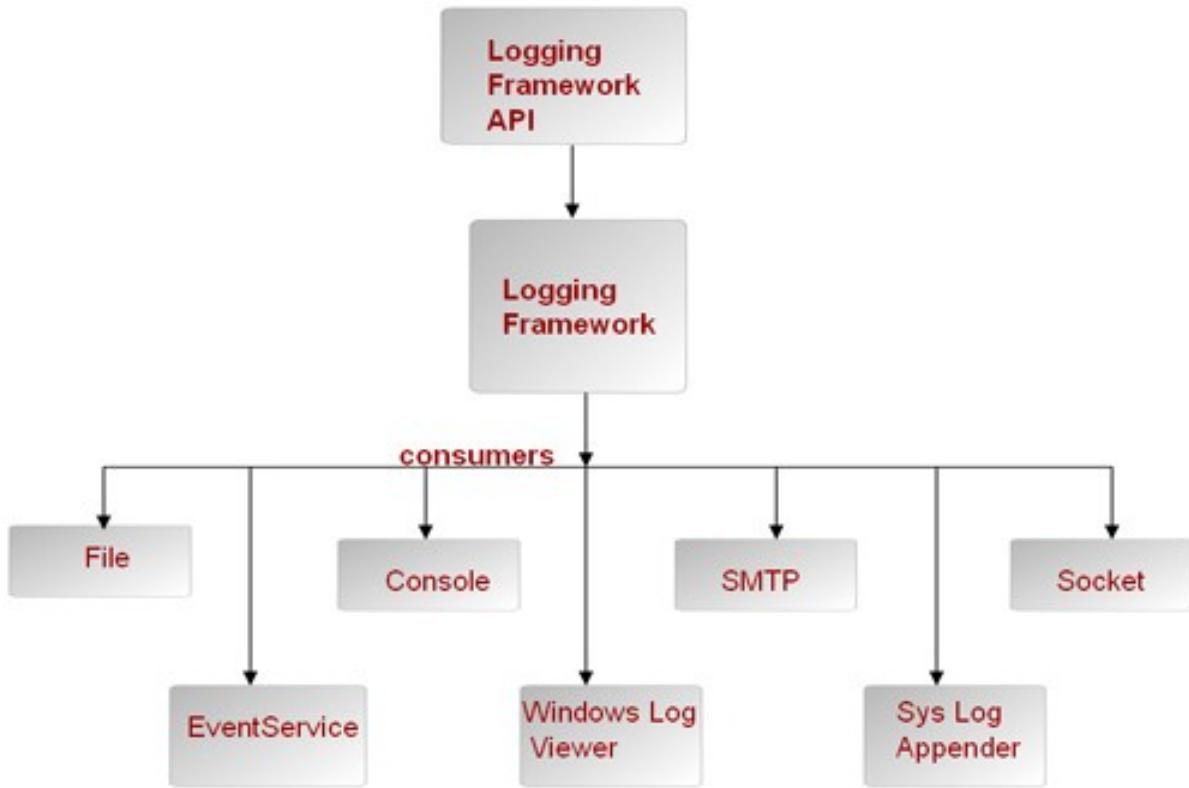
<date>1116933270658</date>
<thread>Dispatcher-2/Worker-0</thread>
<host>Machine</host>
<messageid>{DBF49C4B-73B2-4EDA-9105-E71ABD7E9A55}</messageid>
<user>cn=Steve,cn=organizational users,o=system,cn=cordys,o=vanenburg.com</user>
<component>cn=FTP Service,cn=soap
nodes,o=system,cn=cordys,o=vanenburg.com</component>
<message>FAILED</message>
<class>com.eibus.applicationconnector.java.Transaction</class>
<method>process</method>
<file>Transaction.java</file>
<line>76</line>
<Throwable>
    java.lang.ArithmeticException: / by zero
    at com.eibus.applicationconnector.java.Transaction.process
(Transaction.java:72)
    at com.eibus.soap.SOAPTransaction.handleBodyBlock(SOAPTransaction.java:405)
    at com.eibus.soap.SOAPTransaction.
    <init/>
    (SOAPTransaction.java:274)
    at com.eibus.soap.Processor.onReceive(Processor.java:430)
    at com.eibus.connector.nom.Connector.onReceive(Connector.java:300)
    at com.eibus.transport.Middleware$1.execute(Middleware.java:225)
    at com.eibus.util.threadpool.WorkerThread.run(WorkerThread.java:30)
</Throwable>
</LogMessage>

```

## **Logging framework process**

Following are the steps in the logging framework process:

1. The administrator enables the log option for a particular category (`com.eibus.transport`) with a predefined severity (Warn), on the Explorer. The administrator can also select the log consumers, such as the file or event service, to which the log messages must be written.
2. The source code instrumented with log messages is run.
3. The framework logs instrumented messages to the selected consumers.
4. Use the log viewer tools to view the logged messages.



## Severity Levels

The logging framework supports the following severity levels:

Severity	Description
Debug	Log messages for developers and support personnel for debugging the application and detecting error causes, if any.
Info	Log messages for the operators. For each application connector, log requests and responses with severity information.
Warn	Log messages for errors caused by incorrect user inputs and ignorable errors.
Error	Log messages for exceptions and errors.
Fatal	Log messages for critical errors, service being stopped, application crashing, and other similar errors.

## **Log consumer**

Any application that consumes log messages is a log consumer. AppWorks Platform supports multiple log consumers, including those from Log4j. Multiple consumers can be selected for publishing messages. The appropriate log consumer must be selected during service configuration.

Use the following log consumers to publish log messages:

- **File:** Writes log messages to the file and location specified during log configuration. The default path for this file is <AppWorks Platform\_installdir>\Logs\<filename.xml>. If the file name is not specified, the log messages are published to the console.

### **Note:**

- For installation on Linux, after installing the patch, you must have read and write permissions on the <AppWorks Platform\_installdir> to publish alerts and log statements to the file. If the file name contains folders, requisite permissions must be assigned to the folders.
- In the **Publish to File** field of the Logger Consumers pane on the Service - Parser Processor window, use a double-slash while specifying the folder path. For example, <Drive>:\folder1\testlog.txt in place of <Drive>:\folder1\testlog.txt.

Types of files:

<b>File type</b>	<b>Description</b>
Rolling file	Similar to writing to file, except when the log file crosses a specified file size, it is rolled over. The messages are then published to the file with a new name. If the name already exists, it writes to a file with a new name. The maximum file size must be specified, during service configuration.
Daily rolling file	Messages are published to the specified file and rolled over based on the specified time intervals, such as weekly, and daily. You must specify the file name and the time at which the file must be rolled over. Use the following formats: <ul style="list-style-type: none"><li>■ Roll over monthly: yyyy-MM</li><li>■ Roll over weekly: yyyy-ww</li><li>■ Roll over daily: yyyy-MM-dd</li><li>■ Roll over every hour: yyyy-MM-dd-HH</li><li>■ Roll over each minute: yyyy-MM-dd-HH-mm</li></ul>

- **Socket:** Publishes the log messages to the specified socket. Specify the machine name and port number while configuring the logger.
- **OS-based Event Log:** Publishes log messages to:
  - **Windows Event Log** for the Windows operating system
  - **Syslog** with the facility `LOCAL7` for the Linux operating system.

The default consumer to which log messages are written is `ProcessNamedDailyRollingFileAppender`. The file location for this consumer is `<AppWorks Platform_installdir>\Logs`. The naming convention depends on the process to which the log messages are written. The following log files are created for each process:

<b>Process</b>	<b>Log file</b>
Service	<code>&lt;Organization Name&gt;#&lt;Service Group DN&gt;#&lt;Service DN&gt;.xml</code> , for example, <code>system#xml_store_service#xml_store_processor.xml</code> <b>Note:</b> Replace the spaces with an underscore.
OS	<code>&lt;OS Process Name&gt;.xml</code> , for example, <code>MyOSProcess.xml</code> is created for an OS Process named <b>MyOSProcess</b> . View these log files by selecting the <b>View Log</b> option in the context menu available for the OS process.
monitor	<code>Monitor.xml</code> , which is the default file in the <code>&lt;AppWorks Platform_installdir&gt;\Logs</code> directory.
gateway	<code>Gateway.xml</code> , which is available in the <code>&lt;AppWorks Platform_installdir&gt;\Logs</code> directory.

#### **Note:**

- If the log messages that must be published do not belong to these log consumers, they are published to the `General.xml` in `<AppWorks Platform_installdir>\Logs` directory.
- AppWorks Platform Logger supports all the Log4j Appenders. You can use the log4J configuration for customized logging.

#### **Log message format**

The following formats are used for log messages:

<b>Log message</b>	<b>Format</b>
Output log	<code>&lt;LogMessage&gt;</code>

Log message	Format
message	<pre> &lt;category&gt;java&lt;/category&gt; &lt;severity&gt;ERROR&lt;/severity&gt; &lt;date&gt;1116933270643&lt;/date&gt; &lt;thread&gt;Dispatcher-2/Worker-0&lt;/thread&gt; &lt;host&gt;Machine&lt;/host&gt; &lt;messageid&gt;{DBF49C4B-73B2-4EDA-9105-E71ABD7E9A55}&lt;/messageid&gt; &lt;user&gt;cn=Steve,cn=organizational users,o=system,cn=cordys,o=vanenburg.com&lt;/user&gt; &lt;component&gt;cn=FTP Service,cn=soap nodes,o=system,cn=cordys,o=vanenburg.com&lt;/component&gt; &lt;message&gt;This is an error.&lt;/message&gt; &lt;class&gt;com.eibus.applicationconnector.java.Transaction&lt;/class&gt; &lt;method&gt;process&lt;/method&gt; &lt;file&gt;Transaction.java&lt;/file&gt; &lt;line&gt;63&lt;/line&gt;  &lt;/LogMessage&gt;</pre>
Log message for exceptions	<pre> &lt;LogMessage&gt;  &lt;category&gt;ldap&lt;/category&gt; &lt;severity&gt;ERROR&lt;/severity&gt; &lt;date&gt;1116933270658&lt;/date&gt; &lt;thread&gt;Dispatcher-2/Worker-0&lt;/thread&gt; &lt;host&gt;Machine&lt;/host&gt; &lt;messageid&gt;{DBF49C4B-73B2-4EDA-9105-E71ABD7E9A55}&lt;/messageid&gt; &lt;user&gt;cn=Steve,cn=organizational users,o=system,cn=cordys,o=vanenburg.com&lt;/user&gt; &lt;component&gt;cn=FTP Service,cn=soap nodes,o=system,cn=cordys,o=vanenburg.com&lt;/component&gt; &lt;message&gt;FAILED&lt;/message&gt; &lt;class&gt;com.eibus.applicationconnector.java.Transaction&lt;/class&gt; &lt;method&gt;process&lt;/method&gt; &lt;file&gt;Transaction.java&lt;/file&gt; &lt;line&gt;76&lt;/line&gt; &lt;Throwable&gt;      java.lang.ArithmetricException: / by zero     at com.eibus.applicationconnector.java.Transacti     on.process (Transaction.java:72)     at com.eibus.soap.SOAPTransaction.handleBodyBlock     (SOAPTransaction.java:405)</pre>

<b>Log message</b>	<b>Format</b>
	<pre> at com.eibus.soap.SOAPTransaction. &lt;init/&gt; (SOAPTransaction.java:274) at com.eibus.soap.Processor.onReceive (Processor.java:430) at com.eibus.connector.nom.Connector.onReceive (Connector.java:300) at com.eibus.transport.Middleware\$1.execute (Middleware.java:225) at com.eibus.util.threadpool.WorkerThread.run (WorkerThread.java:30)  &lt;/Throwable&gt; &lt;/LogMessage&gt;</pre>

### ***Log message search parameters***

This topic describes the search parameters used to look up log messages.

<b>Field Name</b>	<b>Description</b>	<b>Example</b>
Application	The Application name where the web-service has been implemented.	Cordys ESBServer
Category	Log messages are classified into a set of categories which help the user map the priority level.	com.eibus.xform.util.XFormsService
Correlation ID	This refers to the unique Transaction ID which is propagated across the request flow.	b3838240-e0e7-4a9c-a822-6cb9865dd5e2
Diagnostic Context	This fields	The XForms invoking this request flow can be identified by the prefix, 'XForms='. So in the given example the XForms

<b>Field Name</b>	<b>Description</b>	<b>Example</b>
	serves as an identifier to extract the log messages in a given context.  The Diagnostic context added by a logger context is placed here in name-value pairs.	can be deduced as '/logviewer/CALViewer.caf'. XForms=/logviewer/CALViewer.caf host=CIN0321I messageID=269AF5D8-331D-470B-9BE6-B3A186410EC6 processid=7772 Method=Restart
Hop Count	Defines the number of hops in the 'request-flow'. Every time a service-container invokes another service container, the counter is incremented.	1
Localizable Message	The XML representation of the localizable message	WSDLError occurred while reading definition "http://schemas.cordys.com/General/1.0/"> <cordys:MessageCode> Cordys.ESBServer.<cordys:LocalizableMessage xmlns:cordys= Messages. soapProcessorStarted </cordys:MessageCode> <cordys:Insertion> cn=Auditing,cn=Auditing,cn=soap nodes,o=system, cn=cordys,cn=build434,o=vanenburg.com</cordys:Insertion> </cordys:LocalizableMessage>
Localizable Message ID	The translatable unique ID of the log message.	Cordys.ESBServer.Messages.soapProcessorStarted
Service Container	The Service Container where the log message	cn=XForms,cn=XForms,cn=soap nodes,o=system,cn=cordys, cn=defaultinst,o=vanenburg.com

<b>Field Name</b>	<b>Description</b>	<b>Example</b>
	occurred.	
Thread	Specifies the thread at the Service Group, that invoked the log-message.	socket: cin0321I:44608/Worker-1
Throwable	Specifies the exception occurred.	<pre> com.eibus.xml.nom.XMLException: Unable to perform operation:Node.delete()     at com.eibus.xml.nom.Node. delete(Native Method) at com.eibus.xml.nom.Node.     delete(Node.java:468) at com.cordys.cws.internal. util.nom. XmlNomNodeManagerImpl\$NomNodeWrapper. clear(XmlNomNodeManager.java:87) at com.cordys.cws.internal. util.nom. XmlNomNodeManagerImpl.clear (XmlNomNodeManager.java:227) at com.cordys.cws.internal. util.nom. XmlNomNodeManager\$1.releasedInstance (XmlNomNodeManager.java:31) at com.cordys.cws.internal. util.nom.XmlNomNodeManager\$1. releasedInstance (XmlNomNodeManager.java:20)     at com.cordys.cws.context. SubjectObserverHelper.releaseInstance (SubjectObserverHelper.java:91) at com.cordys.cws.context. SubjectObserverHelper.access\$000 (SubjectObserverHelper.java:23) at com.cordys.cws.context. SubjectObserverHelper\$ObserverHelper. releaseContext(SubjectObserverHelper. java:106) at com.cordys.cws. internal.context. ObserverSubject\$ObserverInstance. release(ObserverSubject.java:43) at com.cordys.cws.internal. context.ObserverSubject.release (ObserverSubject.java:87) at com.cordys.cws.session. SessionManager\$SessionImpl. releaseInternal(SessionManager.java:204) </pre>

Field Name	Description	Example
		at com.cordys.cws.session.SessionManager\$SessionImpl.release (SessionManager.java:184) at com.cordys.cws.session.SessionManager.releaseSession (SessionManager.java:617) at com.cordys.cws.internal.operation.CustomOperationThread.run (CustomOperationThread.java:195)
Timestamp	Mentions the time of log message occurrence.	02/05/2008 09:44:08.57
User	The organizational user DN who initiated the SOAP request.	cn=test,cn=organizational users, o=system, cn=cordys,cn=defaultinst, o=vanenburg.com
Severity	This field defines the degree of information required by a user about a log message. See <a href="#">Categories and severity levels</a>	Error

### ***Log viewer interface for a service container***

The following table describes the fields on the log viewer interface:

Parameter	Description
Severity	Degree of information required by a user about a log message.
User	Organizational user DN who initiated the SOAP request.
Thread	Thread at the service group, that invoked the log message.
Message	Exception occurred.
Msg ID	Message ID in a SOAP request.
Timestamp	Time of log message occurrence.

Parameter	Description
Category	Log messages are classified into a set of categories, which help the user map the priority level.
Host	Name of the computer, where the message has been logged.
Component	Triggers the SOAP request.
Process ID	ID of the service container.

## Logging configuration interface

The following table contains the fields in the logging configuration interface:

Field name	Description
Enable Logging	Enables the log categories and severities.
Enable System Policy	Disables all the log categories and severities because the system-level configuration is applied. Select this option to apply the <code>Log4jConfiguration.xml</code> file in the <code>&lt;AppWorks Platform_installdir&gt;</code> to a service.
Root Severity	Turns the log option on for all the severity levels, including those that are not listed on the screen. The severity levels selected here apply to all categories. See <a href="#">Categories and severity levels</a> for details. <b>Note:</b> Enabling the lowest severity level automatically enables the remaining severity levels. However, enabling or selecting the highest severity level enables only that severity.
Category-wise Severities	Severity for a specific category. If the <b>Root Severity</b> is selected, it automatically turns the <b>Category-wise Severities</b> option on for all categories.
Logger Consumers	Enabled only when the <b>Enable System Policy</b> option is not selected. AppWorks Platform supports various kinds of log consumers, including those from Log4j for consuming log messages. One or more consumers can be selected for publishing messages. The appropriate log consumer must be selected when configuring the service container. The following options are available: <ul style="list-style-type: none"> <li>■ <b>Publish to File:</b> Publishes the log messages to the file specified during log configuration. The default path for this file is <code>&lt;AppWorks Platform_installdir&gt;\Logs\&lt;filename&gt;.xml</code>. If no file name is specified, log messages are published to the console.</li> </ul> <b>Note:</b> Use a double-slash for folder path specification, for example, <code>&lt;Drive&gt;:\\folder1\\testlog.txt</code> . If the service container runs in an

Field name	Description
	<p>application server or OS process, the log messages of all the service containers that share the same application server or OS process are written to the same log file belonging to that application server or OS process. The default name of the log file for the application server is <code>Application_Server.xml</code>.</p> <ul style="list-style-type: none"> <li>• Do not roll: Select this option if you do not want to roll a file.</li> <li>• Roll File every: Rolls over files based on your defined time intervals, which can be monthly, weekly, daily, hourly, or for every minute.</li> <li>• Roll File for every: Writes the logs to the file. When the log file exceeds a specified file size, it is rolled over with a <code>filename.n</code>. Here, <b>n</b> is the number of times the file has exceeded the specified size.</li> <li>■ <b>Publish to Database:</b> Publishes the log messages to the database specified during AppWorks Platform installation.</li> <li>■ <b>Publish to Remote Host :</b> Publishes the log messages to the specified remote host. You must specify the Host Name and Port Number.</li> <li>■ <b>Publish to OS-based event log:</b> Publishes log messages to either the Windows Event Log for Windows OS or Syslog, with facility <code>LOCAL7</code> for Linux OS.</li> </ul>

**Note:**

- If a severity option is selected under both **Root Severities** and **Category-wise Severities**, the option selected under Category-wise Severities takes precedence.
- For a Linux installation, you must have read and write permissions on the `<AppWorks Platform_installdir>` to publish the alerts and log statements to the file. If the file name contains folders, requisite permissions must be assigned to the folders.

### **Logger context**

Logger context is the detailed information about the context of an application. It is propagated across all the activities or transactions involved in that application. For example, the name of a business process model can be set as a context under which all operations, such as service containers and UI applications, that are involved in the process, inherit this context.

While analyzing messages that are logged by different services, involve the chain of relevant messages which hold a causal relation with one another. For example, an XForm initiates a short-lived process, whereas, the WS-AppServer code accesses another Web service causing an error. This error is logged at all levels such as Web service, WS-AppServer, short-lived process, and XForm. You can view the full set of log messages and their causal relationship with one another.

To ensure causal relation between log messages, Composite Application Logging (CAL) helps propagate the logger context across Web applications and SOAP services, which are all related by a unique ID. Logger context can be visualized as an **application diagnostic context**, where all the applications and SOAP services inheriting the context push entries in their **local** diagnostic context.

A composite application developer can add custom entries into a Logger Context. Use the setDiagnosticContext of the Logger API to set the custom diagnostic context for a web application. For details, see Logger in the *AppWorks Platform API Reference Guide*.

### **Log viewer summary**

The following table describes the fields in the log viewer summary:

Type	Description
From	Name of the table from where the upload process is started.
To	Destination table where the data is processed.
Status	Status of the process.
StartTime	Start time of uploading the entities.
EndTime	End time of uploading the entities.
TotalRecords	Total number of records uploaded.
Errors	Errors in the upload process.

### **Icons and buttons**

The following table describes the icons and buttons:

Type	Description
	Refresh the page.
Close	Close the page.

### **Viewing log messages**

Composite Application Logging (CAL) implementation facilitates comprehensive logging mechanism, where multiple applications can log their details at a single place. The Logging Service Group is necessary for database logging and is enabled by default for all the service containers. You must configure a database that serves as the single location to store log messages generated during the execution of a composite application. This location is specified during AppWorks Platform installation in the Database Information screen of the wizard.

The information gathered from the log files helps analyze the error state and fix support calls. AppWorks Platform supports different kinds of log consumers such as file and event

---

service, including those from Log4j. You can view the log messages using a service container, the Log Viewer, or Apache Chainsaw.

**Important:** You must have the role of a Log Viewer Admin to view log messages.

**To view log messages using a service container:**

1. On the Welcome page, click  (System Resource Manager).  
The System Resource Manager lists the available service containers in the Service Containers App Palette.

**Tip:** You can click  to view the existing service containers.

2. Right-click <Service Container> and click **View Log**.  
The Log Viewer - <Service Container> displays these log messages.

Parameter	Description
<b>Category</b>	Categories that help the user map the priority level
<b>Component</b>	Component that triggered the SOAP request
<b>Host</b>	Computer where the message was logged
<b>Message</b>	Specifies the exception occurred
<b>Msg ID</b>	Message ID in a SOAP request
<b>Process ID</b>	ID of the service container
<b>Severity</b>	Degree of information required by a user
<b>Thread</b>	Thread that invoked the log message
<b>Timestamp</b>	Time of log message occurrence
<b>User</b>	Organizational user DN who initiated the SOAP request

The event details are published to the file specified during log configuration. In the absence of log configuration, the log messages are read from the file

`ProcessNamedDailyRollingFileAppender` in `<AppWorks Platform_installdir>\Logs`.

The naming convention depends on the process to which the log messages are written:

- **Service Container** - `<Organization Name>#<Service Group DN>#<SOAP Processor DN>.xml`, for example, `system#xml_store_service#xml_store_processor.xml` (spaces are replaced with an underscore).
- **OS process** - `<OS Process Name>.xml`, for example, `MyOSProcess.xml` is created for an OS Process named MyOSProcess.
- **Monitor** - `Monitor.xml` in `<AppWorks Platform_installdir>\Logs`.
- **Gateway** - `Gateway.xml` in `<AppWorks Platform_installdir>\Logs`.

If log messages do not fall in any of the four categories, they are published to `<AppWorks Platform_installdir>\Logs\General.xml`.

---

## To view log messages in the Log Viewer:

The AppWorks Platform Log viewer helps you find the log messages using search criteria.

1. On the Welcome page, click  (Log Viewer).

The Log Viewer displays log messages generated in the last 24 hours. Only messages with Severity type Error and Fatal are displayed.

2. Expand **Search Filters**.

- In Diagnostic Context , enter the combination of a name-value pair that matches the content from the logs, for example, host=hostname&processid=1234.
- To view the logs generated in a specific time frame, set the dates in From and To. If you set only the From date, logs generated from that date onward are displayed. If you set only the To date, the logs generated in the last 24 hours are displayed. If you do not set the time period, all available logs are displayed.
- Select a level of severity. Messages from that severity level and all the levels beneath it are displayed. For example, if you set INFO as the severity level, the messages belonging to the levels, INFO, WARN, ERROR, and FATAL are also displayed.

3. Click **Search**.

**Note:** Select **Advanced Search** to search by criteria other than Severity, Timestamp, Message, or Category. Right-click the table header and select **Column Chooser**. Then select **Application** or **Add Filter**. Application is where the Web service was implemented. Add Filter allows you to create a custom filter using strings or integers. For example, a timestamp uses a string type, such as **Is** (for the exact log message) or **Contains** (for one or more words of the message you specify). A Hop Count is an example of an integers type (valid values are <, >, =). To remove a custom filter, click .

4. Select a log message from the search results.

Log messages and corresponding details are displayed. See [Categories and severity levels](#).

**Note:** Context-specific information is provided in Diagnostic Context. This information is provided by the respective component. You can use these details as name value pairs for further grouping of messages by applying the search criteria. Ensure to delimit your values by '&', for example, host=CIN0321L&processid=5948&messageid={BEABD11B-BB14} is an MDC value.

## To disable logging to a database:

1. Click **System Resource Manager > Service Groups App Palette > Logging > ServiceGroup Properties - Logging**.
2. Clear the check mark for **Enable Composite Application Logging**.

## To view log messages using Apache Chainsaw

The Apaches Chainsaw is an open source GUI-based tool to view log messages.

1. In Provide Log Configuration Details, in the Logger Consumers frame, do the following:
  - Select **Publish to Remote Host**.
  - Type the computer number in **Host Name**.
  - Type the port number as 4445. This is the default port for Chainsaw. For information about changing the configuration, see the Chainsaw documentation.
2. Run the following command from the command prompt:
  - `java org.apache.log4j.chainsaw.Main`

### ***Message codes for database errors***

The following table lists the database related errors, their interpretation and the required actions to correct the errors.

<b>Message code</b>	<b>Error message</b>	<b>Cause</b>	<b>Action</b>
Cordys.Database.Native.Messages.duplicateKeyError	Inserting a record in the object '{0}' has failed as the primary or the unique key constraint '{1}' has been violated.	An INSERT or UPDATE statement attempted to create a duplicate value to the primary or the unique key field.	Provide a valid value for the key column.
Cordys.Database.Native.Messages.foreignKeyConstraintError	Inserting a record in the object	An INSERT or UPDATE statement attempted to create a foreign key value which has no matching primary key value.	Provide a valid value for a

<b>Message code</b>	<b>Error message</b>	<b>Cause</b>	<b>Action</b>
	'{0}' has failed as the foreign key constraint '{1}' has been violated.		foreign key column or add a matching primary key in the referenced table.
Cordys.Database.Native.Messages.notNullConstraintError	The value 'NULL' cannot be inserted into '{0}'.	An INSERT or UPDATE statement attempted to create a NULL value in a field which is defined as NOT NULL.	Provide a valid value for the column.
Cordys.Database.Native.Messages.checkConstraintError	Inserting a record in the object '{0}' has failed as the Check constraint '{1}' has been violated.	The values being inserted do not fulfill the named Check constraint.	Provide a valid value for the column.
Cordys.Database.Native.Messages.object	The	Objects such as table, view,	-

<b>Message code</b>	<b>Error message</b>	<b>Cause</b>	<b>Action</b>
ctDoesNotExist	object '{0}' does not exist.	procedure, or function specified in the insert / update / delete / query /DML /DDL request does not exist in the database.	
Cordys.Database.Native.Messages.objectAlreadyExist	The object '{0}' already exists.	An attempt to create a table or view using the DDL, which already exists in the database.	-
Cordys.Database.Native.Messages.syntaxError	A database error occurred as the syntax of the statement is incorrect. The error is '{0}'.	The syntax of DML, DDL, or Query statement is incorrect. For example: <pre>&lt;query&gt;SEECT * FROM Employees WHERE ID=1&lt;/query&gt;</pre> Here the 'SEECT' keyword is incorrect.	Correct the syntax
Cordys.Database.Native.Messages.databaseOrDriverError	An error occurred while processing the request. An error	All the errors are thrown by the database or driver and are not mapped to any specific error code.	-

<b>Message code</b>	<b>Error message</b>	<b>Cause</b>	<b>Action</b>
	from the database server or driver '{0}'.		

## AppWorks Platform user privileges

A AppWorks Platform user is a special operating system user created during installation. All the AppWorks Platform services run with AppWorks Platform user context. This prevents security hacks and grants lesser privilege, avoiding services from disturbing the system environment. AppWorks Platform, by default, runs under its own local user context, such as DefaultInst. A typical user name is InstanceName, optionally \_<number>> is added if there exists multiple users of the same name, for example, DefaultUser\_1.

The permissions to access different components by an AppWorks Platform user are as follows:

<b>Component</b>	<b>Permissions</b>
Database components	A AppWorks Platform user must have permissions on any relevant folders that are outside the <AppWorks Platform_installdir>, for example, the Oracle installation folder. If Windows authentication is enabled on MS SQL Server, the AppWorks Platform user must be added as a user in the MS SQL Server.

## Change password for key store and trust store

By default, the AppWorks Platform installer creates passwords for the private key, key store, and trust store. For security reasons, you can change this password.

---

## To change the password for key, key store and trust store:

1. Enter this command at the command prompt to change the key password:  
keytool -keypasswd -alias bcp -keystore <full path of the key store file>  
The full path must include the file name, for example: D:\Program Files\AppWorks Platform\certificates\keystore\cordys-srv-del-6c-ks.jks.
2. Enter the current and new password for the key.
3. Enter this command at the command prompt to change the key store password:  
keytool -storepasswd -keystore <full path of the key store file>  
The full path must include the file name, for example: D:\Program Files\AppWorks Platform\certificates\keystore\cordys-srv-del-6c-ks.jk.
4. Enter the current and new password for the key store.

**Note:** The new key and new key store password must be the same.

5. Enter this command at the command prompt to change the trust password:  
keytool -storepasswd -keystore <full path of the trust store file>  
The full path must include the file name, for example: D:\Program Files\AppWorks Platform\certificates\truststore\cordys-srv-del-6c-ks.jks.
6. Enter the current and new password for the trust store.
7. Generate the Base 64 encoded password, by modifying the passwords in wcp.properties.
  - **Windows:** Type the following at the command line:  
cd <OpenText CARS\_installdir>\bin EncodePassword <password>
  - **Linux:** Type the following at the command line:  
cd <OpenText CARS\_installdir>\sbin ./EncodePassword.sh <password>

**Note:** Remember the encoded password and use it for filling the bus.ssl.keystorepassword / bus.ssl.truststorepassword in wcp.properties.

8. Change the key store and trust store passwords in AppWorks Platform by modifying the following attributes in wcp.properties:
  - bus.ssl.keystorepassword=<base64 encoded password>
  - bus.ssl.truststorepassword=<base64 encoded password>

## Configuring file system permissions for CWS folders on Linux

When the file system permissions for the Collaborative Workspace (CWS) folders are not correct on Linux environments, you can encounter issues due to the strict way Linux deals with file system permissions. This mainly applies to the CWS synchronization folder (sync folder) and the CWS build folder.

By default, these are located at:

- <AppWorks\_Platform\_installdir>/cws/sync
- <AppWorks\_Platform\_installdir>/cws/build

---

Many issues occur after manually performing file system actions on the sync or build folder such as:

- Moving or copying existing files or folders from outside the sync folder into it.
- Creating new files or folders in the sync folder.

Therefore, you must ensure that:

- The default OpenText AppWorks Platform user is the owner of the sync and build folder and all the child folders and files.
- The default OpenText AppWorks Platform user has read and write access to the sync and build folder and all the child folders and files.

## **Configuring file system permissions**

Before you begin, ensure the default OpenText AppWorks Platform user is the owner of the files that are copied or created, and has read and write permissions.

**Note:** You can modify the owner of the sync folder to the default OpenText AppWorks Platform user as follows:

```
~> chown -R <default_platform_user> <path-to-sync-folder>
```

where <default\_platform\_user> must be replaced with the user name of the operating system user account of the default OpenText AppWorks Platform user.

By default, the sync folder is automatically created on first synchronization of an arbitrary development workspace. The file system permissions of the sync folder are set correctly.

Issues arise when:

- Files or folders are manually copied to or created in the sync folder.
- Users other than the default OpenText AppWorks Platform user need full and permanent access to the sync folder.

### **To configure file system permissions:**

Create a new group to provide every developer the proper rights on files and directories by setting the permissions on the new group and adding all the developers to this group.

1. Create a new group, for example, `cwsdevelopers`, by running the following command:  
~> groupadd cwsdevelopers
2. Add the default OpenText AppWorks Platform user to the group. For example, to add the default OpenText AppWorks Platform user to the group `cwsdevelopers`, run the following command:  
~> usermod -G cwsdevelopers platformDefaultinst

**Note:** Every file operation within the <AppWorks Platform\_installdir>, including the sync folder, is run by the default OpenText AppWorks Platform user, often denoted by `platformDefaultinst`. This user must be added to the group initially. If this user is not a part of the group, all the files and directories created by the default OpenText AppWorks Platform user are not allowed to be modified by any other user or group.

3. Add the required OS users to the group by running the following command:  
~> usermod -G cwsdevelopers <user\_name>  
where <user\_name> must be replaced with the user name of the operating system user account of that developer.
  4. Set the group as owner of the sync folder to provide all the group members full access to the synchronization folder. The group must be set as its owner. For example, to make the group `cwsdevelopers` the owner of the synchronization folder, run the following command:  
~> chgrp -R cwsdevelopers <sync-folder>
  5. Set the `setgid` bit on the sync folder to ensure that all the group members have full access to any file or folder that is created in the synchronization. As a result, newly created files have the group as its owner instead of the user who created the file. To set the `setgid` bit for the synchronization folder, run the following command:  
~> chmod -R g+s <sync-folder>
  6. Modify the umask of the default OpenText AppWorks Platform user.  
Open the `.bash_profile` of the OpenText AppWorks Platform user with vi and add  
`umask 0002`
- Note:** Every file operation within the <AppWorks Platform\_installdir> including the synchronization folder are run by the default OpenText AppWorks Platform user, often denoted by `platformDefaultinst`, who needs the `umask 0002` for every file operation performed by the processes started under the context of this user. For more information, see [umask](#).
7. Restart the OpenText AppWorks Platform (<instance name>) to make the modifications effective.  
The file system permissions are now configured for the OpenText AppWorks Platform user.

## Configuring Samba file sharing

Several issues arise due to file actions that are performed on the synchronization folder through a remote connection, for example, by creating a file share to the shared folder. If the synchronization folder is shared through Samba, the Samba configuration file, often located at `/etc/samba/smb.conf`, must include the following lines for the respective configuration section.

### /etc/samba/smb.conf file

```
...
# sharing configuration for the CWS synchronization folder
[<share-name>]
...
create mask = 0774
directory mask = 0775
```

## Connecting to the Linux Server through SFTP or SSH/Telnet clients

If you connect to the Linux server using SFTP, for example, WINSCP; or SSH/Telnet clients, for example, PuTTY, you must ensure that every directory you create in the <AppWorks Platform\_installdir>, including the synchronization directory, is fully accessible to every other member of the group. This can be done by properly setting the umask property in the user profile. For most Linux distributions, the umask property is by default set to 0022. This provides you only read/write access to files, and read/write/search access to directories you own, and all others have read access to your files and read/search access to your directories.

To provide all the members of the newly created group full access to files and directories created by you, the umask property must be set to 0002. This must be done by including `umask 0002` in the user profile `.bash_profile`, or you can put it in the system wide profile which is applicable to all the users `/etc/profile`.

## Creating a configuration profile for an FTP server

**Important:** Before you begin, you must have the role of a FTP Admin.

To perform file transfer operations, you must invoke the appropriate FTP SOAP Request, which requires the FTP server details to process the request. See **FTP Connector** in the *AppWorks Platform API Reference Guide*. Instead of specifying the details of an FTP server in every request, you can create a configuration profile for the FTP server. With the **FTP Configurations Manager**, you can create and store FTP configuration profiles in the XMLStore. These profiles comprise FTP server details such as name, description, server name, authentication, and connectivity details.

### To create a configuration profile for a FTP server:

1. On the **Welcome** page, click  (FTP Configurations Manager).  
The FTP Configurations Manager window opens.
2. Click  and specify the details of the FTP server in the pane below. See [FTP Configuration interface](#).
3. Click  (Save).  
A profile of the FTP configuration details is created. These details are stored as an FTP Configuration profile at **XMLStore Explorer > Collection > Cordys > WCP > FTP > <ftpconfiguration>**. Sample configuration:

```
<ftpconfiguration>
  <name>Microsoft_Public_FTP</name>
  <description>Microsoft FTP Server Configuration</description>
  <data>
    <server>ftp.microsoft.com</server>
    <port>21</port>
    <username>ftpuser</username>
```

```

<password>ZnRwdXNlckBjb3JkeXMuY29t</password>
<protocol>FTP</protocol>
<transfermode type="Passive">
    <externalipaddress/>
    <portrange>
        <from/>
        <to/>
    </portrange>
</transfermode>
<basepath>C:\Inetpub\ftproot</basepath>
</data>
</ftpconfiguration>

```

In this example, Microsoft\_Public\_FTP is the name of the FTP configuration profile. You can use this information in a SOAP Request, as indicated in the next code snippet and do not have to define the FTP server details. When the SOAP request is invoked, these server details are retrieved from the defined FTP configuration profile.

**Note:** You must refer to the name of the FTP configuration profile within the <configuration> tag of the SOAP request.

```

<getListOfFiles xmlns="http://schemas.cordys.com/ftpconnector/1.1">
    <configuration>Microsoft_Public_FTP</configuration>
    <notification-subject>FTP</notification-subject>
    <directory>\</directory>
</getListOfFiles>

```

**Note:** To edit the details of an FTP configuration profile, select the configuration profile in the table. The details pane at the bottom is enabled for editing. Make the necessary changes and click  (Save).

**Note:** To delete an FTP configuration profile, select the check box next to the name of the profile and click  (Delete).

## Removing header details from SOAP responses

The headers of SOAP responses contain details about their senders and recipients. This information may be used for sending malicious requests and in DoS attacks. Removing these details also increases the network bandwidth. This procedure removes these details and retains only the message ID in the header.

1. Click **Start > Programs > AppWorks Platform > Instance Name > Tools > Management Console**.  
The Management Console window opens.
2. Click **Security Properties** in Management Console window.  
The Security Properties window opens.
3. On **Others**, select **Enable Strip Outgoing Traffic**.  
Header details from SOAP responses are removed.

## Restrict access to public SOAP methods

AppWorks Platform Web Gateway manages sending and receiving SOAP requests and responses from the service groups. Initially, the gateway determines the authenticity of the user in the AppWorks Platform environment. Later, based on the Web service interface details and the namespace contained in the request, the gateway sends the SOAP request to the service container without checking for permissions that the user needs to access the Web service interface or namespace. This process involved the risk of exposing all the public Web service operations on LDAP.

As a security measure, AppWorks Platform provides a restricted environment for sending requests to service groups, based on the sandbox enabled model approach. Before sending the request to the service container, the gateway checks if the Web service interface and namespace have required permissions for the user. When the request is sent, the gateway prepares a permission object containing the user reference, Web service interface, and namespace. This is acted upon by the access controller. The access controller verifies the permission object against the security policy. The security policy comprises two lists, a black list containing users and Web service operations that are denied access, and a white list containing users and Web service operations that are granted access. The request is sent to the service group only when the Web service operations have necessary permissions.

## Starting a batch of service containers

On computers with limited CPU capacity, when there are many service containers, the OpenText AppWorks Platform (<instance name>) cannot start all of them at once. This is because simultaneous start of all these service containers results in insufficient CPU share among the Virtual Machines (VMs). This causes errors or time outs during start up.

To resolve this, a set of service containers is started in batches, so that the CPU utilization is spread over a period of time. This enables the required share of CPU cycles for all service containers.

This solution is needed only:

- During startup of the OpenText AppWorks Platform (<instance name>) and not after it has started.
- For automatic start, enabled for service containers.

The role of a system administrator is to set the batch size and timeout required for the initialization of service containers.

Attributes	Batch size	Timeout
Definition	Maximum number of service containers that can be started simultaneously.	Maximum timeout allowed for a service container to start before accepting another service in the batch.

<b>Attributes</b>	<b>Batch size</b>	<b>Timeout</b>
Usage	With the help of batch size, Monitor determines the number of service containers that are to be started simultaneously.	With the help of timeout, Monitor determines the start of a service container (Monitor assumes that service container has started after the timeout expiry).
Storage	Stored at Monitor Service Configuration.	Stored at Monitor Container configuration.
Size	The batch size can be changed dynamically.	The timeout period can be changed dynamically.
Value	The default value to specify a batch size is a large integer value (Integer MAX-VALUE).	The default value to specify a Timeout is 60,000 milliseconds, that is, one minute.

At start up, on the basis of the dependency list, the Monitor sorts all the configured service containers on its computer. Depending on the specified batch size, it identifies the number of service containers that are to be started simultaneously. The predefined timeout governs the maximum time for initialization of each service container, after which the Monitor picks up the next SOAP Processor from the sorted list.

The following steps indicate the functionality of starting a set of service containers in batches.

- At startup, Monitor reads all the service containers that are configured on its computer.
- Each configuration of the application connector is read to retrieve the dependency list.

**Note:** Dependency list refers to a set of name spaces that the application is dependent on at start up (namespaces used to invoke any Web service in the open() Web service operation of the application connector). The dependency list is associated with the application connector, so it must be a part of the application connector configuration. In case of no dependency list, Monitor treats it as **no dependency to start the service container**, which implies that such a service container is in the initial batches. The dependency list must not contain namespaces of LDAP and Monitor.

- Monitor consolidates all these dependency lists and sorts the service containers accordingly.

**Note:** The predefined batch size is used to determine the number of service containers to start simultaneously.

- A service container is initialized by invoking its application connector's open() Web service operation. After this Web service operation is invoked successfully, the client VM communicates to the Monitor that the initialization is complete. The timeout defined at the monitor govern the maximum time for this entire operation.
- After the timeout expiry or successful initialization of the service container, the Monitor picks up the next service container from the sorted list until complete.

## User authentication

This table describes the user authentication properties.

Property	Default value	Description
com.eibus.certificate.identity	issuer+serial	By setting this property, the authenticated users are mapped to their digital certificate depending on the IssuerDN and SerialNumber. This mapping constrains the users to sign in only through the specified certificate and helps the administrator restrict all those signing in through other certificates.

**Note:** The value of this property can also be set to subject (value=subject), where the authenticated users are mapped to their digital certificate depending on the SubjectDN. This mapping allows the users to sign in through multiple certificates, which hold the same SubjectDN. OpenText recommends not using this option because it is deprecated.

## JDBC details interface

This topic describes the fields displayed when the JDBC driver is selected as the database.

Field name	Description
Connection String	Mandatory. Displays information such as the database server and port number, in the required JDBC format. It uses a connection to the database. For example, <code>jdbc:sqlserver://&lt;&lt;server_name&gt;&gt;:&lt;&lt;port_no&gt;&gt;</code> is the format for Microsoft SQL Server, if you are using the Microsoft SQL Server JDBC driver. <b>Note:</b> <ul style="list-style-type: none"><li>■ If you are connecting to Oracle using the Oracle Thin driver, the connection string format can be <code>jdbc:oracle:thin:@&lt;&lt;server_name&gt;&gt;:&lt;&lt;port_no&gt;&gt;:&lt;&lt;SID/service_name&gt;&gt;</code> or <code>jdbc:oracle:thin:@&lt;&lt;server_name&gt;&gt;:&lt;&lt;port_no&gt;&gt;/&lt;&lt;SID/service_name&gt;&gt;</code></li></ul>

Field name	Description
	name>>, depending on the Oracle version and configuration.
JDBC Driver XA Class	Optional. Implementation class. Stored as a part of the database configuration. Provide the fully qualified class name to enable support for distributed transactions in the service container configuration page. Depending on the type of JDBC driver selected, that is, Microsoft SQL Server, Oracle, or PostgreSQL, the text box is populated with the default implementation driver class. You can modify the class name.
DB User	Optional. Authorized user name for authentication to the database. Required only when the selected database needs authentication.
Use SSL	<p>Optional. Enable Secure Socket Layer (SSL) connectivity that enables encryption of data during transfer over any network. To use this option in AppWorks Platform, configure the database server to use SSL.</p> <p><b>Note:</b> To use SSL connectivity with the Oracle DB server and Oracle Thin driver, use a proper connection string with <code>tcps</code>, as specified in the Oracle documentation, for example,</p> <pre>jdbc:oracle:thin:@(DESCRIPTION=(ADDRESS=(PROTOCOL=tcps)(HOST=&lt;&lt;servername&gt;&gt;) (PORT=&lt;&lt;port number&gt;&gt;)) (CONNECT_DATA=(SERVICE_NAME=&lt;&lt;servicename&gt;&gt;)))</pre>
JDBC Driver	<p>Mandatory. Select the required JDBC driver from the list. If it is not present in the list, select <b>Other</b> and enter the value of the driver in <b>Driver Class</b>, for example, for Oracle database, select Oracle Thin.</p> <p><b>Note:</b> The computer on which the service container is configured must have the JDBC driver installed, and the driver JAR must be set in the System classpath.</p>
Password	Optional. Password for authentication.
Default Schema	Optional. Name of a database, that is, the

Field name	Description
	<p>catalog or schema, to be connected. For example, if you want to connect to a particular database in the SQL Server, this database name must be mentioned in this field. If no database name is mentioned, the service container is connected to your default database, for example, master for 'sa' user.</p> <p><b>Note:</b></p> <ul style="list-style-type: none"> <li>■ AppWorks Platform works with the default schema of the user or database for the SQL server and PostgreSQL. In PostgreSQL, you can connect to a different schema by adding the optional parameter <code>currentSchema=&lt;&lt;schema name&gt;&gt;</code> in the connection string. The default value is public.</li> <li>■ For an Oracle server, this field must contain the schema to which the service container must be connected. If no schema is specified, the service container is connected to your default schema. All Read and Update requests are directed only to the connected database or schema.</li> <li>■ For proper functioning of the ACL, specify the default database in its original case (upper or lower).</li> </ul>

## Process Engine

Process Engine is an important component of the AppWorks Platform workflow management system and forms the execution layer of an application. Process Engine executes process models that are designed in the AppWorks Platform business process modeling environment. The process models are stored in CoBOC as BPML documents. When Process Engine receives a request, it loads the process model from the CoBOC repository using the name of the process, prepares its executable copy, and instantiates the process.

## Memory status interface

The following table describes the parameters in the memory status interface:

Parameter	Description
Virtual Memory Usage	Virtual memory allocated to the service container.
Resident Memory Usage	Physical memory currently allocated to each service container.
NOM Nodes Memory	Memory for NOM nodes.
NOM Memory	NOM memory details.

## New toolbar item configuration interface

The following table describes the fields in the New Toolbar Item Configuration interface.

Field	Description
url	Any website. Applications can be any website, URL, MS-Word-document, PDF file, Powerpoint presentation, and so forth, as long as it is available somewhere on a website and a browser is capable of viewing it as URL
id	A mandatory field that provides the identity of the application. Unless an application ID is specified, the application cannot start. This must be a unique value. It is used for identifying or referring to the application
caption	The caption that is shown as the taskbar or tab caption in the application window. This is also displayed in the title bar.
description	The description that is shown in the tree node (as part of the menu). This value, if entered, is shown on title bar text as caption-description.
frame	The frame that you select for the application to be nested in. If you select <b>New Window</b> , the application opens in an independent browser window.
icon	An image file that can be used as the icon to represent the application on the toolbar. If no file is specified, a default application icon is used for the new item.
organization	The distinguished name (DN) in LDAP of the organization that is used when the application must run in a different organization context. By default, the application runs in the context of the user's default organization.
docked	An option that specifies whether the application must be started docked in the specified frame or undocked. Select this check box if the application should be docked in an existing frame. If cleared, specify the coordinates for the application window. Possible values are true and false. The default value is true. You can specify the

<b>Field</b>	<b>Description</b>
	height, width, left, and top attributes if the value is false.
left, top, height, width	The co-ordinates of the application.
focus	A boolean value that, when set to True, gives the application focus when it opens.
display	The visibility of the application. Possible values are hidden, visible, and current.
title	An option that defines whether a title bar is visible for the application. Possible values are True and False. The default value is True (the title bar is visible).
blink	A boolean value that, when set to True, blinks the application to highlight it.
border	A check box that defines a border for the application window.
taskbar	An option that specifies whether to show the application on the taskbar. Possible values are True and False. The default value is True.
data	Any additional data for the application. You can use the browse button to browse and select the required data.
class	An option that is used by developers to identify and categorize applications. This can have any value.

## Object tree settings

ACL settings can be set at the following levels:

<b>Level</b>	<b>Description</b>
Service groups	From System Resource Manager, users can navigate to method sets and methods. They can set ACL permissions for service groups, web service interfaces, and web service operations.
Metadata services	From the Metadata Services folder, developers and administrators can navigate to tables and fields. They can set ACL permissions for tables, fields, related tables, stored procedures, and others.
XML Store level	From the XML Store Explorer folder, users can navigate to the Web service interfaces and Web service operations in the Applications. They can set ACL permissions for web service

<b>Level</b>	<b>Description</b>
	interfaces and web service operations. They can also set ACLs for custom methods.
LDAP Object level	From the LDAP Explorer, users can navigate to the required LDAP objects and set ACLs on them.

## Operational-level views

Operational-level views display the monitored results of an aspect of all processes. Following are the views available at the operational level:

<b>View</b>	<b>Description</b>	<b>Corresponding composite control</b>
Process Load for Period	<p>Displays a graphical view of the number of instances started for all the processes by grouping data based on the specified period.</p> <p>The data displayed on the graph is grouped in terms of the Groupby (Hour, Day, Week, Month, and Year) period you select. For example, if you select Week as the Groupby period, the graph groups and displays the data on a weekly basis.</p>	ProcessLoadForPeriod_AcrossProcess
Aggregate Process Instances per Status	<p>Provides a high level summarized view of the number of process instances per status for all the processes for the specified period.</p> <p>This view displays the Aborted, Waiting, Complete, Suspended, and Terminated statuses for all the processes.</p>	ProcessInstancesPerStatus_AcrossProcess
GetTopXProcessesByStatus	Displays the top 'X'	TopXInstancesPerStatus_

View	Description	Corresponding composite control
	<p>processes having the highest number of instances for a status.</p> <p>Provide the required integer in the Top field, select the required status from the Process for Status drop-down list, and click  to view the corresponding data. Click the table header to sort the table view.</p>	AcrossProcess

**Note:** The time values displayed are based on the time zone of the client browser in a 12-hour format.

## Permitted activities in a transaction

You can use a short-lived process in the transaction of another business process model. That is, a short-lived process is called as a synchronous sub-process from the transaction of the main process. You can also include non-transactional activities within a group of transactional activities. For example, you can model a one-way notification that invokes a log web service.

The following activities are permitted in a transaction:

- WS-AppServer methods: By default, a WS-AppServer method participates in a transaction. If the **Participate in Transaction** check box is cleared and the methods are a part of the transaction, the activities to which these methods are attached do not display transactional behavior and are just called as normal web services. If the **Participate in Transaction** check box is selected, these methods are executed in a process (for short-lived and long-lived processes). These services should also point to the same database. For example, assume that a transaction has three activities - Activities A, B, and C, out of which Activities A and B participate in the transaction. In this case only Activities A and B, which are marked as **Participate in Transaction** behave as a part of the transaction. If an exception occurs during execution of Activity B, Activity A is rolled back, but Activity C, although a part of the transaction, if it is not marked as **Participate in Transaction**, it executes as a normal web service.
- In a short-lived process, the reply property in the input message for a WS-AppServer database insert/update/delete activity enables you to define whether a response is required for the SOAP request. If the reply property is set to no, the response message of this method is empty in the message map. By default, this property is set to yes.

- 
- External web services: You can define an external web service as part of a transaction. However, external services do not display transactional behavior. An example of a non-transactional external web service is obtaining information about the weather from an external source.
  - Asynchronous sub-processes containing manual activities: The process is executed independently and in parallel with another process. Only the asynchronous sub-process is triggered and it runs outside the transaction.
  - Synchronous sub-processes (where the sub-process is a short-lived process): In this case, the process waits for the response of the process that executes before it. You must model the synchronous sub-process within the constraints of a transaction.
  - Manual activities of type Info: You can include manual activities of type Info (for example, notifications) within a transaction.

The following activities cannot be defined within a transaction:

- Manual activities of type task.
- Receive Messages.
- Delay Events.
- Synchronous sub-processes with manual tasks, Delay and Receive Message events.
- Nested transactions.
- Synchronous sub-processes that do not have the Participate in a Transaction property marked.

## Overview of integration with WS-AppServer

Exposing business logic as web services typically involves developing business logic for an application and writing code for complex backend methods. However, using AppWorks Platform Web Service Application Server (WS-AppServer), you can directly expose an application's business logic as a web service. This enables you to focus on developing business logic pertaining to an application.

WS-AppServer provides the following features:

- An infrastructure for quick and easy object-oriented application development.
- An abstraction layer over the database system that supports basic methods.
- Built-in transaction management and connection pooling.
- Integration with AppWorks Platform XForms.
- Support for only a single level of aggregation. AppWorks Platform XForms does not support integration with nested classes and the corresponding methods of WS-AppServer.

WS-AppServer contains the business logic of an application as web services. You can use suitable front-end client tools to consume the web services and generate applications from them.

---

It is possible to use AppWorks Platform XForms to generate applications from available AppWorks Platform methods.

- No client-side validation code is written or repeated for any application; you can use the methods generated using the web services of WS-AppServer.
- You can generate applications by dragging methods, thus reducing the client-side coding required. This also enables the integration of business logic from heterogeneous data sources and the generation of user interfaces.
- The application development life cycle is shorter.

## Prerequisites for installing applications

You must satisfy the following preliminary requirements before installing the applications.

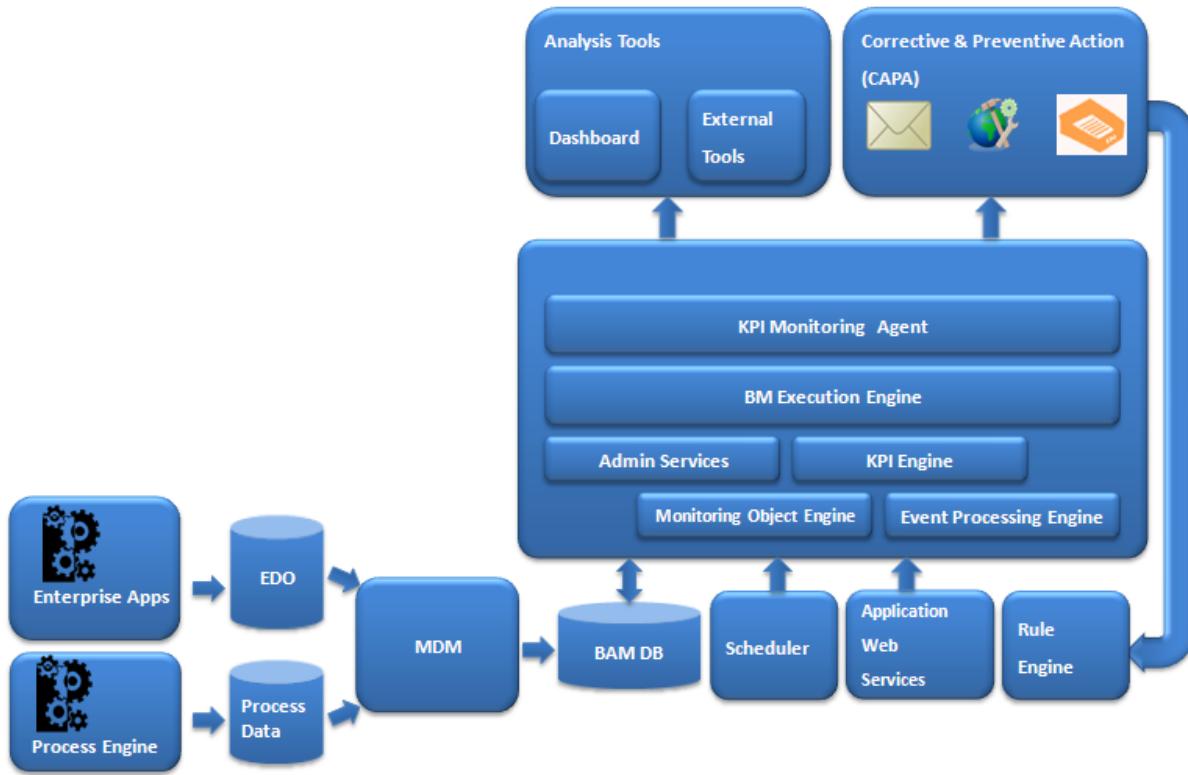
- You must have the role of systemAdmin.
- The system organization must be your default organization.
- You must have the role of Process Developer to install an application with business process model content.
- The AppWorks Platform installation process creates a user named Cordys <Instance Name> in the machine (Example: CordysLocal, CordysDefaultInstal). This user must have write permissions on the <AppWorks Platform\_installdir> where the application is installed. By default, the installer provides write permissions on this directory.
- To install an application with system environment variables, DLL registration, or virtual folder creation content on a Windows-based machine, see [Loading an application with system environment variables content](#).

## Accessing Web services using the Web services explorer

Web Services Explorer tool enables quick access to Web services. See Web Services Explorer in the *AppWorks Platform Advanced Development Guide*. The business logic in AppWorks Platform is contained in the form of Web service operations. These Web service operations are contained in a Web service interface, which is exposed as a Web service. Through the Web Services Explorer, it is possible to access the WSDL for Web service operations. To access the Web Services Explorer, type `http://<machinename>:<port number>/cordys/services` in the address field of your Web browser and press Enter.

## Runtime architecture

From a real-time monitoring perspective on BAM, the right data should be available at the right time and harmonized to process it and relevant actions must be triggered. Different aspects of BAM runtime architecture are depicted in the following diagram:



## 1. Data collection and harmonization

AppWorks Platform BAM has three sources of data:

- AppWorks Platform business process data
- Application web services
- Enterprise data objects

AppWorks Platform business process data and web services data is harmonized as Process Monitoring object.

- Enterprise Data Object: The EDOs are synchronized using the Cordys MDM component. The Enterprise application database acts as the MDM spoke node and the BAM database acts as the MDM Hub node. When any changes happen in the application database, data is synchronized with the corresponding tables residing in the BAM database - data is synchronized from the spoke node to the Hub node.

## 1. Event processing

Process events are identified by the BAM server based on process state transitions in the database. When a process event occurs, The event processing engine determines whether it is a trigger event for any Business Event Response. A pre-processing of the event condition is done using Xpath expression evaluation to check if the event condition will be met in the rule engine and whether the event condition is going to be met then Business Event Response object is inserted in BAM repository to trigger rules and corresponding corrective and preventive actions (CAPA).

---

When the temporal (time-based) events are defined on process or activity lead time, an event monitor is started, which is embedded schedule in BAM server, with the maximum lead time defined as event monitoring time to check completion of process or activity. This concept is also known as proactive surveillance, where the system proactively does surveillance on happening of certain events.

You can define temporal condition along with the data conditions.

## **2. Business measure execution**

To execute Business Measure web services, fetch and execute the corresponding SQL query from the BAM meta database.

## **3. KPI monitoring**

The monitoring agent (Scheduler) monitors KPIs by invoking the KPI web service with parameters specified during design-time in the KPI Editor. The KPI web service is hosted in the BAM server. The BAM server invokes corresponding the Business Measure web services and output is given to the KPI engine where the expression is evaluated to calculate the KPI value. The KPI Monitor object, filled with the KPI value, is given to the rule engine, which evaluates the corresponding rules against the defined KPI ranges and corresponding corrective or preventive actions (CAPA) are triggered. KPI values are stored in the BAM repository for trend analysis.

## **4. Admin services**

Different admin services such as publish, unpublish, delete, and so on are available as part of BAM Admin services.

## **5. Dashboard**

The BAM dashboard charts are built using fusion charts, flex based technology created from the AppWorks Platform XForms designer. On drill-down from one view to another, parameters pass from the x-axis of source view to target view. Users can also drill-down to the default view of the Process Monitoring object and from there they can navigate to process instance graphical or grid view.

# **Removing a certificate from a trust relation**

You can remove a certificate for a collection of service groups. However, if the group has only one certificate associated with it, you cannot delete the certificate.

### **Before you begin:**

- You must have the role of Security Administrator.

### **To remove a certificate from a trust relation:**

1. On the Welcome page, click  (Security Administration).  
The Security Administration window opens.
2. Click the Service Group Trust Configuration tab.  
All service groups belonging to a group are displayed in a column. A description for the collection of service groups is displayed on top of the column. The description is

---

optional. All certificates associated with the collection of service groups are displayed below the description.

3. Right-click the certificate in the certificate section of the group and select **Remove Certificate**.
4. Click  (Save).

The certificate is removed from the collection of service groups.

**Note:** At any point during configuration, click  (Revert) to revert changes to the last saved state.

## Reloading the database metadata

To update the contents of a WS-AppServer package with changes to database tables, you need to reload the database metadata that is mapped to that database.

### Before you begin:

- Reset the WS-AppServer service container mapped to this database.

You can reload the database metadata in two ways.

### To reload the database metadata using Workspace Documents (My Recent Documents):

- Click  (Database Metadata), select **Actions**, and then select **Reload Database Metadata**.

### To reload the database metadata using Workspace Documents (Explorer):

1. Open <solution> > <project> >  <database metadata>.
2. Right-click the <database metadata> and select **Reload Database Metadata**.  
The database metadata is reloaded with the updates to the database, if any.

### After you complete this task:

- Update the WS-AppServer package using the reloaded database metadata.

## Replacing tags in a query

When you specify the conditional details for a query in the Where text box of the Custom Web Service Operations Generator page, the value of the field and the operator are displayed as tags. You need to replace these tags.

**Note:** The Where text box can be used to build conditions but has limitations in handling hard-coded strings and string concatenation operators such as +.

---

### To replace tags in a query:

1. Select the `opr` tag.
2. In the Expressions section, double-click the operator.  
You can also manually type the operator replacing the `opr` tag.
3. Right-click the `value` tag in the Where text box of the Custom Web Service Operations Generator page.  
You can also manually type the value replacing the `value` tag. The related fields are displayed on the context menu.
4. Select an option from the context menu.  
The **Set Field Type** option is used when the query is an associated query or an inner query that refers to a parent table. The `value` tag is replaced with the selection.

The tags are replaced with the related field name/value.

A parameter value can be recognized only if it has a colon (:) before the field name. It must be provided when the developer types the value of the parameter manually.

## ACL types

ACL is of two types.

- Conditional: ACL evaluation is based on the data and metadata of the object. See Conditional ACL in the *AppWorks Platform API Reference Guide*.
- Unconditional: ACL evaluation is based on the metadata of the object. See [Unconditional ACL](#).

## Using APIs for database access

### Before you begin:

All database requests in AppWorks Platform need to follow a standard protocol. The XQY API is a component in AppWorks Platform that handles the AppWorks Platform database protocol. This XQY API is accessible using class

`com.eibus.applicationconnector.sql.DBConnectionPool`. An instance of `DBConnectionPool` represents a pool of connections to the database via XQY API. This pool exposes different APIs to route requests to the database. This enables the user to contact the database directly in the AppWorks Platform environment. It also decreases the response time and enables users to perform dynamic query execution.

### To use APIs for database access:

1. Create a connection pool.  
The public Java class `DBConnectionPool` present in the `com.eibus.applicationconnector.sql` package can be used to create a connection pool by supplying all the details required to connect to the DB server in XML form.

```

String dsoXml = null;
dsoXml = <configuration>
    <update-connections>10</update-connections>
    <read-connections>10</read-connections>
    <dso jdbcDriver="com.microsoft.sqlserver.jdbc.SQLServerDriver"
        driver="JDBC"
        connectionString="jdbc:sqlserver://<<server>>:<<port>>"'
        defaultDB="<<DB name>>"
        userId="<<user>>"
        xmlencoding="false"
        password="<<base64 encoded password>>"'
        update="true" >
        <query-cache>
            <size>50</size>
            <refresh-interval>3600</refresh-interval>
        </query-cache>
        <cursor-cache>
            <size>50</size>
            <refresh-interval>3600</refresh-interval>
        </cursor-cache>
    </dso>
    <connection-pool>
        <min-update-connections>1</min-update-connections>
        <min-read-connections>1</min-read-connections>
        <refresh-interval>3600</refresh-interval>
    </connection-pool>
</configuration>";
int dso = xmlDoc.parseString(dsoXml);
DBConnectionPool pool = DBConnectionPool._createInstance(dso, null, null, null);

```

This creates a read and an update connection in the pool. For information about `createInstance`, see the Java SDK documentation.

2. Compose the database request. The database request, as required, can be composed using the following web service operations:

<b>Method</b>	<b>Description</b>
<code>query</code>	Reads data from the database.
<code>update</code>	Updates data in the database.
<code>getMetaData</code>	Retrieves the metadata of the database.
<code>validateCommand</code>	Validates a query against the database.
<code>commitCurrentTransaction</code>	Commits the current transaction to the database.
<code>abortCurrentTransaction</code>	Aborts the current transaction.
<code>executeDDL</code>	Executes the input DDL against the database.

### To query the database:

```
<dataset>
    <constructor language="DBSQL">
        <query>select * from Employees where EmployeeID = :EmployeeID</query>
        <parameters>
            <EmployeeID dd="Employees.EmployeeID">1</EmployeeID>
        </parameters>
    </constructor>
</dataset>
```

### To retrieve the metadata of the database:

```
<dataset>
    <metadata requestType="getTableInfo" tableCatalog="Northwind"
        tableName="Employees" tableSchema="" tableType="TABLE"/>
</dataset>
```

### To validate a query against the database:

```
<implementation>
    <validate>
        <query> select * from Employees where EmployeeID = :EmployeeID</query>
    </validate>
</implementation>
```

### To execute an input DDL against the database:

```
<implementation>
    <executeDDL>create table temp_table (field1 int primary key, field2 varchar
(20))</executeDDL>
</implementation>
```

### To update the database:

```
<update>
    <tuple>
        <old>
            <Employees>
                <EmployeeID>1</EmployeeID>
                <FirstName>Nancy</FirstName>
            </Employees>
        </old>
        <new>
            <Employees>
```

```

<EmployeeID>1</EmployeeID>
<FirstName>Nancy123</FirstName>
</Employees>
</new>
</tuple>
</update>

```

### To execute an input DML against the database:

```

<implementation>
  <executeDML>
    <DML>
      <command> INSERT INTO Employees(FirstName, LastName, Title, City) values
(:FirstName, :LastName, :Title, :City) </command>
      <parameters>
        <FirstName dd="Employees.FirstName">Test</FirstName>
        <LastName dd="Employees.LastName">Test</LastName>
        <Title dd="Employees.Title">Test</Title>
        <City dd="Employees.City">Test</City>
      </parameters>
    </DML>
  </executeDML>
</implementation>

```

### 3. Get an appropriate connection from the pool.

The connection pool contains both read and write connections. Read connections can be used to process requests like query, metadata, and validate. Write connections can be used to process requests such as update records, DDL requests, and DML requests where transaction is necessary. Based on the request to be processed, applications can use appropriate APIs to get a read or update connection.

```

WCPDBConnection con = null;
con = pool.getReadConnection();
OR
con = pool.getWriteConnection();

```

### 4. Using the appropriate APIs of the WCPDBConnection class, send a request to the DB server.

This class provides the following APIs to execute the request.

Method	Description
query	Reads data from the database. Pass the <constructor> node xml to query API.
update	Updates data in the database.

<b>Method</b>	<b>Description</b>
getMetaData	Retrieves the metadata of the database.
validateCommand	Validates a query against the database.
commitCurrentTransaction	Commits the current transaction to the database.
abortCurrentTransaction	Aborts the current transaction.
executeDDL	Executes the input DDL against the database.
executeDML	Executes the input DML against the database.
createAuditTable	Creates an audit table from an existing table.
purgeAuditTable	Removes the contents of audit table.

5. Analyze the return value of the database response.

Once the database requests are sent, you receive database responses. The database responses are in XML format. On the basis of the return value, you can determine whether a request is processed successfully or an error occurred. If the return value is 0, the request is processed successfully. If it is less than zero, the request execution resulted in an error. Based on the return value, the application can either commit or abort the transaction for the operation that needs transaction support.

```
If(ret <0)
{
// error happened
con.abortTransaction(); // for write connections
}
else
{
// Processing is success
con.commitTransaction();
}
```

The sample responses received for success and failure follow.

```
Success
<dataset>
  <constructor language="DBSQL">
    <query>select EmployeeID, FirstName, LastName from Employees where
EmployeeID = :EmployeeID</query>
    <parameters>
      <EmployeeID
        dd="Employees.EmployeeID">1</EmployeeID>
    </parameters>
  </constructor>
```

```

<tuple>
  <old>
    <Employees>
      <EmployeeID>1</EmployeeID>
      <FirstName>Nancy123</FirstName>
      <LastName>Davolio</LastName>
    </Employees>
  </old>
</tuple>
</dataset>

Error
<dataset>
  <constructor language="DBSQL">
    <query>select EmployeeID, FirstName, LastName1 from Employees where
EmployeeID = :EmployeeID</query>
    <parameters>
      <EmployeeID
        dd="Employees.EmployeeID">1</EmployeeID>
    </parameters>
  </constructor>
  <error
    TYPE="Enumeration">
    <elem>ColumnsRowset formatting problem</elem>
    <elem>Failed to setQueryMetaData()</elem>
  </error>
</dataset>

```

- After processing is complete, place the connection back into the pool. This is a required step.

Otherwise, this connection cannot be used for any other request processing. You can do this by using web service operations provided in the DBConnectionPool class.

```
Pool.putReadConnection(con); OR Pool.putWriteConnection(con);
```

## Viewing memory status details

You can view the memory being consumed by each Service Container in the System Resource Manager window.

### Before you begin

You must have the role of systemAdmin or organizationalAdmin to view the details of memory status for a service container.

### To view memory status details:

- On the Welcome page, click  (**System Resource Manager**).  
The System Resource Manager window opens and the available service containers are displayed in the Service Containers App Palette.

- Note:** Alternatively, to view the existing service containers, you can click .
2. To view the memory status details, double-click the service container. Corresponding memory details are displayed in the Memory Status App Palette in a graphical representation.

**Note:** See [Memory status interface](#) for details on each parameter.

## Sending and receiving SOAP messages using Simple Client

The Simple Client enables you to send and receive SOAP messages. The tool offers a feature to compose and edit the outgoing SOAP message that is to be sent to a specific back end.

**Note:** An anonymous client is a socket on the client computer, taken from the available free sockets. Such clients are not registered with the LDAP.

### Before you begin

You must have the role of a developer to perform this task.

### To send and receive SOAP messages using Simple Client:

1. Click **Start > Programs > AppWorks Platform > <Instance Name> > Tools > Simple Client**.  
The AppWorks Platform Simple Client window opens.
2. Do any of the following:

Option	Task
To send a SOAP message as an anonymous client	<ol style="list-style-type: none"><li>a. In the <b>Connection</b> pane, select the <b>Use Anonymous</b> option and click <b>Open</b>. The service groups are displayed in the <b>Select Operation</b> pane.</li><li>b. Go to <b>&lt;Service Group&gt; &gt; &lt;Web Service interface&gt;</b> and select the <b>&lt;Web service&gt;</b> for which the SOAP message has to be sent.</li><li>c. Select the required configuration options. See <a href="#">Configuration option for SOAP message</a>.</li><li>d. Click <b>Compose</b>.</li></ol>
To send a SOAP message for one of the services in LDAP	<ol style="list-style-type: none"><li>a. In the <b>Connection</b> pane, select the <b>Set</b> option.</li><li>b. Click <b>Browse</b>. The list of available services in the LDAP are displayed in the Service Groups Listed in the LDAP dialog box.</li><li>c. Select the required service and click <b>OK</b>.</li></ol>

Option	Task
	<p>d. Click <b>Open</b>.  The corresponding service groups are displayed in the <b>Select Operation</b> pane.</p> <p>e. Go to &lt;<b>Service Group</b>&gt; &gt; &lt;<b>Web Service interface</b>&gt; and select the &lt;<b>Web service</b>&gt; for which the SOAP message has to be sent.</p> <p>f. Select the required configuration options. See <a href="#">Configuration option for SOAP message</a>.</p> <p>g. Click <b>Compose</b>.</p>

The SOAP request composed for the selected method is displayed in the **SOAP Request** pane. This is the skeleton of the SOAP message that is to be sent.

**Note:** The **Type Body Here** has to be replaced by the body of the selected method.

3. Provide the other details in the AppWorks Platform Simple Client window. See Simple Client interface in the *AppWorks Platform Advanced Development Guide*.
4. Click **Send** to send the SOAP message.  
After the SOAP Request is sent, the corresponding events (including the SOAP response) are listed in a table in the **Events** pane.

**Note:**

- To sort the events, click the respective column header in the **Events** pane.
- To view the details of any event, double-click its entry in the table. A new dialog box opens, displaying the details. Click < and > buttons to browse through the events populated in the table. Click **Close** to close this window.
- To copy the SOAP message to the clipboard, right-click the text area and select the **Copy to ClipBoard** option on the context menu. If you select a part of text and use this option, only the selected text is copied to the system clipboard. Alternatively, if you choose this option without selecting any text, then the entire SOAP message is copied to the system clipboard.
- To clear the events in the **Events** pane, click **Clear**.
- If the changes made to the LDAP entries after the Simple Client is started are not reflected in the Simple Client window, click **Refresh** in the **Select Operation** pane to reload the entries from the LDAP.
- To clear all the content on the screen, click **Reset** in the **Connection** pane.
- Before sending the request, the tool checks whether the message is well-formatted. If not, an error message is displayed indicating the anomaly.

## Add new registry details interface

This topic describes the Add New Registry Details interface.

The following table contains the registry details:

Field	Description
Name	Mandatory. Name of the new registry entity.
Description	Optional. Describes the entity.
Inquiry URL	Mandatory. URL of a site where a UDDI registry is hosted, for example, <a href="http://server/uddi/inquiry">http://server/uddi/inquiry</a> .
Publish URL	Mandatory. A UDDI Registry has a different URL for publishing Web services, for example, <a href="http://server/uddi/publish">http://server/uddi/publish</a> .
User Name	Optional. Registered user name. To publish Web services to a UDDI Registry, register with that UDDI registry.
Password	Optional. Password used to connect to the UDDI Publish URL.
Category	Optional. Any custom category scheme or category that is required. This is a search filter to search for Web services belonging to this category only, for example, NAICS, and UNSPSC. For more information, see <a href="http://uddi.org/taxonomies/UDDI_Taxonomy_tModels.htm">http://uddi.org/taxonomies/UDDI_Taxonomy_tModels.htm</a> .
Identifier	Optional. Any custom identifier scheme that is required. This is also a search filter, for example, a Dun & Bradstreet D-U-N-S® Number.
Set as Default UDDI Registry	Optional. Select this option to set the registry as default. Status field on the UDDI Registry Manager page denotes the registry as default. <b>Note:</b> To change the default setting, click on the row containing the registry name and select Set as Default UDDI Registry option in the registration details page of that registry. This overwrites the previous default registry settings.

## Advanced search options in Log Viewer

Advanced search options in the Log Viewer help you find the log messages that exactly match the search criteria. To use these options, select Advanced Search in the Log Viewer window. The following search options appear as lists:

- Application: Application name where the Web service has been implemented.
- Add Filter: Various filtering criteria. See [Log message search parameters](#). You can add more than one filter and as you keep adding them, they are displayed in the same section. To remove any filter criterion, click  for the intended filter. You can set one of the following attributes to each filter criterion.

Type	Parameters
Strings	<ul style="list-style-type: none"> <li>• Is: Contains the exact log message</li> <li>• Contains: Contains at least one or more of the words of the message you specify.</li> </ul> <p>For example: Timestamp</p>
Integers	<, >, = <p>For example: Hop Count</p>

## Result Messages interface

The following table describes the fields in the Result Messages interface.

Field	Description
Number	The number of times a SOAP message was sent or received in that particular session.
Time	The day, date, and time when the request was sent and response was received.
Type	The type of the message. It can be a request or response.
Message	The number of the message type.

## Private-public key pair

A private-public key pair is a combination of keys for use in cryptography. A pair consists of a private and public key. The public key is published, while the private key is kept secret by the owner. Often, the public key is contained in a certificate to make sure that a public key belongs to a certain owner. With the public key, you can verify that the signer of the message holds the private key. Also, you can use the public key to encrypt a message so that only the holder of the private key can decrypt it. This signing and encryption is used for code validation, SSL connections, and so on.

## Application package loading interface

The application package loading wizard helps in loading an application package containing business processes into an application space.

**Table 1. Fields in the Selected Application Packages for Loading section**

Field	Description
Application Packages	Displays the list of application packages that have been selected for loading into a system.

**Table 2. Fields in the Provide required inputs section**

Field	Description
Select Business Process Management service	Displays the Business Process Management Service to load the application package.
Business Processes	Displays the business process models packed in the selected application package to load.
View Report	Click <b>View Report</b> to display a dialog box listing the validation errors and warnings encountered while loading the application packages.
Overwrite existing Business Processes	Determines whether the loaded business processes in the application package are overwritten. Options available are: <ul style="list-style-type: none"><li>■ No: The business processes in the application package that are already existing in the system are not overwritten.</li><li>■ Yes: The business processes in the application package that are already existing in the system are overwritten.</li></ul>

**Table 3. Fields in the Start Loading the Application Packages section**

Field	Description
Save the inputs to a file	Select <b>Save the inputs to a file</b> and enter a filename to record the Web-based application packages loading inputs to a file. The generated input file is saved in the following location: \XMLStore\collection\Cordys\WCP\Installation Templates.
Timeout (in minutes)	Specify the default timeout in minutes.
Application Package	Displays the status of the application packages being loaded in to the selected system.

---

## AppWorks Platform Web gateway

The AppWorks Platform Web gateway is the HTTP interface of AppWorks Platform. The AppWorks Platform Web gateway forwards Web requests to and receives responses from service groups. It functions as a bridge between the Web browser and the service groups that operate in AppWorks Platform. Gateway is a Java object and is loaded as a Web server object in the Web server. After it is loaded, the gateway searches for specific service groups in AppWorks Platform and passes the requests back and forth between the service groups and the Web browser.

Unlike any other AppWorks Platform services such as the service containers, the AppWorks Platform Web gateway is not a standalone Java process. It is hosted inside the Web Server and serves AppWorks Platform service requests coming over HTTP or HTTPS.

Following are the functions of the AppWorks Platform Web gateway:

- Mapping an authenticated user to an AppWorks Platform user
- Redirecting the SOAP request to service containers
- Redirecting the SOAP response to clients
- Performing security policy checks

The AppWorks Platform Web gateway does not perform authentication. The Operating System (Basic Authentication), Single Sign-On service of AppWorks Platform (SSO authentication), or any third party Web server plug-in performs the authentication.

The AppWorks Platform Web gateway cannot process the SOAP request on its own and therefore, it must redirect it to one of the service containers. This redirection is done based on the namespace and the method combination.

The Web gateway is part of the AppWorks Platform installation and is loaded into the Web server during the AppWorks Platform installation.

Following are the applications of the gateway in AppWorks Platform that work with documents:

- Upload: This gateway handles requests for uploading documents. There is a library of methods associated with Upload that facilitate uploading documents to a service in AppWorks Platform. See [Upload in the AppWorks Platform API Reference Guide](#).
- Download: This gateway handles requests for downloading documents. There is a library of methods associated with Download that facilitate downloading documents from a service in AppWorks Platform. See [Download in the AppWorks Platform API Reference Guide](#).

The Web gateway can also help in resolving SAML artifacts with user credentials within AppWorks Platform. For information about SAML artifacts, see [Using SAML artifacts](#). To configure the Web gateway with security options to perform policy checks, see [Managing gateway with security options](#).

## Publishing event logs

### To publish event logs:

1. Click <User> > <Organization> > **System Administrator** > **Monitor Web Gateway**.  
The Monitor Web Gateway window opens.
2. Select the **Publish Log Events** check box.

**Note:** If required, the gateway events can also be written to a specified file by checking the **Write Log Events to File** check box (you can specify the file name here). By default, the file is created in the system user's current working directory, unless the path is specifically indicated here.

## Audit Viewer interface

The Audit Viewer User interface contains two tables:

- Artifacts: Displays all the artifacts that meet the search criteria in the search pane.
- Actions: Displays all the audit actions from a selected artifact in the Artifacts table that meet the search criteria in the search pane.

For every action in the action table, the action input data can be viewed through a context menu on the action row. This action input data contains the data that is, for example, added, updated, or deleted.

### Artifacts

Field	Description
Artifact Type	Type of the artifact.
Artifact	Qualified name of the artifact.
Description	Description of the artifact.
Organization	Organization of the artifact.

### Actions

Field	Description	Note
Performed At	Date when the action started.	
Artifact Type	Type of the artifact.	Displayed only when no Artifacts table is displayed.
Artifact	Qualified name of the artifact.	Displayed only when

Field	Description	Note
		no Artifacts table is displayed.
Organization	Organization of the artifact.	Displayed only when no Artifacts table is displayed.
Action	Type of the action, for example, Add, Update, and Delete.	
Status	Status of the action, for example, Incomplete, Completed, and Failed. When the status is incomplete, the action is started but not complete.	
Performed By	User who performed the action on the artifact.	
Description	Description of the action.	

## Base class

A Base class generated by WS-AppServer contains information that WS-AppServer internally uses for application execution. The Base class generated out of the Standard class maps to the corresponding database table. Since this class is system generated, it must not be modified. It contains the following information:

- ClassInfo that describes the class.
- Accessors and mutators (getandsetmethods) that act on the attributes.

Use the Extension class to customize the generated class and add business logic. See [Extension class](#).

## Relationships between objects

An object may be involved in multiple relations. For example, a Product object might have a relation not only to a Supplier object, but also to Order and Pricing Scheme objects. For each relation that the object has, a relation object is bound to the object. Such a relation object stores:

- A related object
- Required information (metadata) about the relation

There are different types of relation objects: One-to-One (1:1), One-to-Many (1:N), and Many-to-One (N:1).

## Methods involving relationships

Relations are described in terms of Primary Key (PK) and Foreign Key (FK). Each Relation object has a RelationInfo object attached, which describes the relation. The Relation\_FK object has a corresponding RelationInfo\_FK object. The RelationInfo\_FK object stores the following information:

- Identifier: Unique identifier to identify the Relation and RelationInfo objects. Within a class, each relation must have a unique identifier.
- localisPK: Whether the origin side of the relation holds the PK.
- localAttributes: Attributes that form the FK or PK.
- loadMethod: Generated method to invoke to load the related objects.
- relatedClass: Name of the class at the target side of the relation.
- relatedAttributes: Names of the attributes that form the PK or FK at the target side.
- relatedIdentifier: Identifier of the relation at the target side.

The RelationInfo object has a method to create a Relation object (either SingleRelation\_FK or MultiRelation\_FK) based on its content. The RelationInfo\_FK objects are stored with the ClassInfo object of a BusObject. The FK/PK model is the underlying model of relations used in WS-AppServer. Changing the FK changes the relation. Therefore, all methods that update the relation (setSupplier, addProduct, and removeProduct) are translated into a call to update the FK fields.

In the Base class of any object, code is generated to initialize the relation objects for each relation. The following code is generated per relation:

- Relation identifier (public final static String)
- RelationInfo object, which is a static attribute
- Relation object, including the generated RelationInfo object
- Method to load the objects in the relation (for example, loadSupplier())
- API of the relation (getSupplier, setSupplier, getProducts, addProduct, and removeProduct)

All relation objects from an object are collected in a list. See a detailed example of the generated code covering both 1:1 and 1:n relationship models. For the Product-Supplier relationship, the following code shows the methods that are generated at the Product side, which is the 1-side of the relation (ProductsBase.java).

```
public Suppliers getSupplierIDObject()
{
    return (Suppliers)getSingletonObject(REL_SupplierIDObject);
}
```

```

public void setSupplierIDObject(Suppliers a_Suppliers)
{
    if (a_Suppliers == null)
    {
        this.setNull("SupplierID");
    }
    else
    {
        this.setSupplierID(a_Suppliers.getSupplierID());
    }
    this.update();
}

public Suppliers loadSupplierIDObject()
{
    String queryText = "select * from Suppliers where SupplierID = :SupplierID";
    QueryObject query = new QueryObject(queryText);
    query.addParameter("SupplierID", "Suppliers.SupplierID", QueryObject.PARAM_INT,
new Integer(getSupplierID()));
    query.setResultClass(Suppliers.class);
    return (Suppliers)query.getObject();
}

```

For the Product-Supplier relationship, the following code shows the methods that are generated at the Supplier side, which is the n-side of the relation (SupplierBase.java).

```

public BusObjectIterator getProductsObjects()
{
    return getMultiRelationObjects(REL_ProductsObjects);
}

public void addProductsObjects(Products a_Products)
{
    a_Products.setSupplierID(this.getSupplierID());
    a_Products.update();
}

public void removeProductsObjects(Products a_Products)
{
    a_Products.setNull("SupplierID");
    a_Products.update();
}

public BusObjectIterator loadProductsObjects()
{
    String queryText =
        "select * from Products where SupplierID = :SupplierID";
    QueryObject query = new QueryObject(queryText);
    query.addParameter("SupplierID", "Products.SupplierID",
        QueryObject.PARAM_INT,
        new Integer(getSupplierID()));
}

```

```

        query.setResultClass(Products.class);
        return query.getObjects();
    }
}

```

In this manner, the Relationship methods can add, remove, or set related objects.

## Business Object Lifecycle tab

The Business Object Lifecycle tab is displayed when you select the Business Object Lifecycle check box while creating a hub data entity. The tab contains the following fields.

Field	Description
State Model	Name of the state model. Click  to specify a state model. For information about designing state models, see Designing a state model in the <i>AppWorks Platform Advanced Development Guide</i> .
State Instance Table Name	Database table name that saves details of the state changes for a business object.
Event Triggers	Option to select the trigger type that was specified when designing the state model: <ul style="list-style-type: none"> <li>■ Insert Trigger: When a record at the spoke entity is inserted, it enters the first state of the state engine. If this record exists in the state engine, specify the event name using which this record moves to another state of the state model from the current state.</li> <li>■ Update Trigger: When a record at the spoke entity is updated, specify the event name using which this record moves to another state of the state model from the current state. If this record does not exist in the state engine, it enters the first state of the state model.</li> <li>■ Delete Trigger: When a record at the spoke entity is deleted, specify the event name using which this record moves to another state of the state model from the current state. If this record does not exist in the state engine, it enters the first state of the state model.</li> </ul>
Enable Registry	Option to enable the registry. The following fields are displayed: <ul style="list-style-type: none"> <li>■ Registry Table Name: Specify a name for the database table to store the registry details.</li> <li>■ ID Generator: Click  for ID Generator. The Select a sequence generator dialog box opens. Select the relevant sequence</li> </ul>

Field	Description
	<p>generator.</p> <p>The Sequence Generator dialog box appears if you have selected a default Sequence Generator. For information on the fields on the Sequence Generator, see <a href="#">Sequence Generator Properties interface</a>. If you selected a customized sequence generator, the property sheet appears with the properties that you specified. You can edit the properties, if necessary.</p>
Enable Data Quality	<p>Option to enable data quality.</p> <p>The Data Quality tab is displayed with the following fields:</p> <ul style="list-style-type: none"> <li>■ Select a Data Quality Plan: Select or create a data quality plan. Select a data quality plan to configure a hub entity to use a third party data quality management tool and to integrate it with Cordys MDM.</li> <li>■ Select a State Model for Match Object Life Cycle: Click  to select a state model for the candidate object's lifecycle, if the selected data quality plan has a match operation defined and if it returns a set of candidate objects as a response to this operation. Each candidate object undergoes the lifecycle of the selected state model.</li> <li>■ Candidate Table for Match: Type the table name that saves details of the candidate object changes for a business object.</li> </ul>

## Business Object tab

The Business Object tab contains the following fields.

Field Name	Description
Business Object Type	<p>Business object types for data synchronization. Select one of the following.</p> <ul style="list-style-type: none"> <li>■ Database Table: Synchronizes from or to a database table</li> <li>■ Others (External Web Service, etc): Synchronizes from or to any other business object type other than the database table.</li> </ul> <p><b>To select the business object for data synchronization:</b></p> <p><b>Note:</b> Use Generic Update is selected by default. Database Table is selected for a hub entity by default, and Others (External Web Service, etc) is cleared.</p> <ol style="list-style-type: none"> <li>1. Click  (Lookup) for Business Object. The Select a Schema Fragment Object dialog box opens.</li> </ol>

Field Name	Description
	<p>2. Select the schema fragment object.  The selected schema fragment object is displayed in the text box for Selected Document.</p> <p>3. Click <b>OK</b>.  The selected schema fragment is displayed for Business Object and the namespace of the schema fragment is displayed for Name Space.  The Select Keys button is displayed.</p> <p>4. Click <b>Select Keys</b> to select fields as the primary key.  The Select Key Fields dialog box opens with a list of fields.</p> <p>5. Select the check boxes of the fields in the Key column for the primary key to enable MDM to uniquely identify an instance of the selected schema fragment object.</p> <p>6. Click <b>Select text field(s)</b> in the Data Type column for the check boxes you selected in the Key column to view the lists.  The data type list is displayed with data types in the Data Type column.</p> <p>7. Select the relevant data type, and then click <b>OK</b>.  The business object is selected. For information about generating Web service operations on the database, see Generating web service operations on databases in the <i>AppWorks Platform Advanced Development Guide</i>.</p>
Enable State Engine check box	<p>Option to enable the State Engine check box to have fine control over synchronization.</p> <p><b>To enable the State Engine check box:</b></p> <ul style="list-style-type: none"> <li>■ Select <b>Enable State Engine check box</b>.  The Sniffer tab is no longer visible. The State Engine tab is displayed.</li> </ul>
Name Space	Namespace of the respective schema fragment.
Use Generic Update check box	<p>Option to use the Generic Update check box to perform insert, update, and delete operations using a single Web service such as the Update and UpdateBusinessObject Web services generated in AppWorks Platform over database tables.</p> <p><b>To use the Generic Update check box:</b></p> <ol style="list-style-type: none"> <li>1. Select <b>Generic Update check box</b>.</li> <li>2. Clear the check box to select three distinct Web services for Insert, Update, and Delete.  The Read, Insert, Update, and Delete Web services are displayed under Web Services.</li> </ol> <p><b>Note:</b> The Use Generic Update check box is selected by default when you select the Database Table business object type but is disabled when you select the Others (External Web Service, etc) business object type.</p>

Field Name	Description
Web Services	<p>List of Web services available for the data entity, which are displayed depending on the Data Store Usage option selected while creating a hub data store. See <i>Creating a data store in the AppWorks Platform Advanced Development Guide</i>. The following Web services are displayed depending on the selected option:</p> <ul style="list-style-type: none"> <li>■ Send data only option: Read Web service</li> <li>■ Receive data only option: Update Web service</li> <li>■ Send and receive data option: Read and Update Web service</li> </ul> <p>Click the Web service, and specify its properties. See <a href="#">Web service properties interface</a>.</p> <p><b>Note:</b> The namespace of the Web service must be the same as the namespace of the schema fragment that is displayed in Name Space.</p>

## Certificate types

A certificate chain is the hierarchy of certificates and includes all certificates from the root certificate to the certificate itself. Each certificate in the certificate chain is preceded by the certificate of its issuer. The type of certificate depends on its position in the certificate chain.

A certificate is issued by a certificate authority (CA) or is self-signed. See [Intermediate certificate authorities](#).

The following certificate types are available.

Type	Description
Self-signed certificate	A self-signed certificate is not issued by any authority and therefore cannot be validated externally.
Root CA certificate	Root CAs are certification authorities that use self-signed certificates. Many certification authorities do not issue certificates to users. A root certificate issues certificates to other certification authorities.
Intermediate CA certificate	Intermediate CAs are certified by a root certificate. An intermediate certificate authority can issue certificates to other intermediate certificate authorities and users. The certificate might or might not contain Certificate Revocation List (CRL) details. However, without CRL details, the validation process of a certificate is incomplete because its revocation status cannot be known.
End user certificate	Any certificate that authenticates an entity is an end user certificate. End user certificates are issued by a CA or are self-signed.

## Changing access rights of the user

1. Right-click **My Computer** on your system and select **Manage** option.  
The Computer Management window opens.
2. Go to **Computer Management (Local) > System Tools > Local Users and Groups > Groups**.  
All the groups accessing your machine are displayed.
3. Select and double-click **Administrators**.  
The Administrator Properties dialog box appears and lists the administrators.
4. You can add or remove administrators.

To add a user as an administrator, do the following:

- a. Click **Add**.  
The Select User, Computers, or Groups dialog box appears.
- b. In **Enter the object names to select**, enter the user name.
- c. Click **OK**.

To remove a user as an administrator, do the following:

- a. In the Administrator Properties dialog box, select the user.
- b. Click **Remove**.
5. Click **Apply** and **OK** in the Administrator Properties dialog box.

Administrative privileges are changed to the user.

## Changing access rights of users

Administrators can change access rights of users.

### To change the access rights of a user in a Windows environment:

1. Right-click **My Computer** on your computer and select **Manage**.  
The Computer Management window opens.
2. Go to **Computer Management (Local) > System Tools > Local Users and Groups > Groups**.  
All the groups accessing your machine are displayed.
3. Select and double-click **Administrators**.  
The Administrator Properties dialog box opens and lists the administrators.
4. To add a user as an administrator:
  - a. Click **Add**.  
The Select User, Computers, or Groups dialog box opens.
  - b. Enter the username in **Enter the object names to select**, and then click **OK**.

5. To remove a user as an administrator:
  - a. Select the user to remove from the Administrator Properties dialog box.
  - b. Click **Remove**.
6. Click **Apply** and **OK** in the Administrator Properties dialog box.  
Administrative privileges are changed for the user.

## Rule Repository service interface

The Rule Repository service is used to configure the settings for rule execution.

The following table describes the fields on the Rule Repository tab.

Field	Description
Select Database Configuration	<p><b>To create a database configuration:</b></p> <ul style="list-style-type: none"> <li>■ Select <b>New Database Configuration</b> from the drop-down list and provide the required information in the dialog box that opens. See <a href="#">Creating a database configuration</a>.</li> </ul> <p><b>To continue with an existing database configuration:</b></p> <ul style="list-style-type: none"> <li>■ Select the database configuration from the drop-down list. The associated fields are automatically filled.</li> </ul>
Advanced Options	<p>Expand the group box and provide the necessary details. See Advanced properties in the AppWorks Platform Advanced Development Guide.</p> <p><b>Note:</b> Setting <b>Cursor Cache Size</b> to a high value results in high memory consumption. An administrator should set an optimal value based on the application usage. This is also applicable to the Query Cache size.</p> <p>The optimal cursor cache size can be between 50 - 100, depending on the application usage.</p>
Runtime Properties	<p><b>Rule Action Threads</b></p> <p>The number of threads required to execute the external actions, such as</p> <ul style="list-style-type: none"> <li>■ Send notification</li> <li>■ Invoke web service</li> <li>■ Trigger business process</li> </ul> <p>If the number of transactions performed by the rule engine, involving external actions, is huge, increase the size of the rule action threads. The default size is 5.</p> <p><b>Max. value for Dispatcher queue size</b></p> <p>The maximum number of asynchronous rule actions that can be queued in the dispatcher queue. The default size is 50000.</p>

Field	Description
	<p>Setting the value to 0 allows an infinite number of rule actions in the queue and may result in an Out of Memory error. Depending on the application's requirements, change the size of the dispatcher queue.</p> <p>Setting the parameter to a higher value allows for higher memory and lower value limits memory usage.</p> <p>It is exposed to JMX as a cold setting.</p> <p><b>Trace Rule Execution</b></p> <p>Enables tracing of the rules. Select the check box to enable rule tracking.</p> <p>Enabling this feature impacts performance as it demands the recording of details in the database during the rule execution.</p> <p><b>Monitor Request and Response</b></p> <p>Enables the persistence of SOAP request and response of external actions</p> <p><b>Custom Rule Monitor Listener</b></p> <p>Provide a fully qualified class name of the custom listener java class to access the rule execution data. This class should implement the Java interface <code>com.cordys.bre.rim.RuleMonitor</code>. The Rule Monitor implementation should return the class implementing <code>com.cordys.bre.rim.RuleSessionListener</code>. The <code>RuleSessionListener</code> interface contains the various sets of events that users need to implement. The events provide the API's to access Input / output messages for a rule, rule set details, request and response details of rule actions, and so forth. See the Java documentation.</p>

## CoBOC

Collaborative Business Object Cache (CoBOC) is a repository of business objects created and managed using AppWorks Platform. AppWorks Platform provides a robust capability for effective management of business objects, which includes a mechanism for object state management, configurable business rules, and real-time state monitoring. CoBOC also offers a repository plug-in that allows data to persist in a database or application.

## Business Objects

The objects that transact in CoBOC are XML objects. These XML objects are instances of templates that have been created and are also known as business objects. For example, an object of type Purchase Order may be inserted in CoBOC and can be made to interact with another object of type Sales Order to perform a particular transaction. Likewise, several thousands of objects can be made to interact with each other. Throughout their existence in

---

the cache, a whole range of business processes may act on them that can determine the very nature of the collaboration setup of an enterprise.

## CoBOC - A Common Repository for Collaborative Business Objects

The implementation of a business process may call for the creation of several business objects collaborating with each other, contributing to the overall implementation of the process. From this perspective, based on the design of the application, which may call for the implementation of several business process flows, a multitude of business objects may need to collaborate to perform business functions. Speed of implementation and ease of access to these business objects become critical factors that could decide the effectiveness of business processes.

CoBOC accomplishes this by providing a common repository where all business objects can collaborate in a fast and an efficient manner throughout their lifecycles. The repository can hold business objects and XML documents such as schema definitions, process models, and mappings. CoBOC is a powerful layer on the database and abstracts the user from querying the database directly. Instead, developers can query CoBOC itself and use CoBOC's powerful features to manipulate business objects to their specifications.

**Important:** CoBOC has no restrictions on the case-sensitivity but depends on the case-sensitivity of the database and the file systems. If the database is case-sensitive, CoBOC is also case-sensitive and vice versa. For example, SQL server is by default case-insensitive, but Oracle by default is case-sensitive. Different file systems have different behaviors. Windows is case-insensitive, while Linux is case-sensitive. Therefore, while designing an application, a developer must take care of these issues.

## Key Features of CoBOC

Following are the key features of CoBOC:

- Repository for business objects
- Object state management
- Database-independent persistence of business objects
- In-memory management of business objects

## Deploying CoBOC

You can deploy CoBOC in two ways:

- As a service, where CoBOC functionality is delivered through SOAP requests and responses.
- Embedded CoBOC, where CoBOC functionality is delivered as APIs. For more information about embedded CoBOC functionality, see Java APIs.

## CoBOC Transaction Monitor

**Note:** CoBOC Transaction Monitor is deprecated in this version.

Depending on the complexity of the applications you develop, thousands of transactions might be triggered inside CoBOC. It is critical that you have the facility to monitor such transactions to gather important information. The CoBOC Transaction Monitor helps you monitor transactions related to both CoBOC and the rule engine. It not only displays the transactions, but also enables you to drill down and view critical information about transaction operations and details about the objects involved in each operation.

You can use the Transaction Monitor to view the following:

- Transactions based on selected criteria
- CoBOC operations
- Rule engine operations
- XML of each object

## CoBOC Transaction Monitor interface

**Note:** CoBOC Transaction Monitor is deprecated in this version.

The following table describes the criteria you can use from the Select Criteria section to display the transactions.

Field	Description
User	User who initiated the transaction.
Status	Different status types for a transaction. <ul style="list-style-type: none"><li>■ Started</li><li>■ Aborted</li><li>■ Committed</li></ul>
From	Start date and time to display transactions that occurred since the specified date and time. Click  to select the start date. Specify the time (hh:mm format) in the text box next to  .
To	End date and time to display processes that have been instantiated until this date. Click  to select the date. Specify the time (hh:mm format) in the text box next to  .

The following table describes the fields in the Transaction Grid section.

<b>Field</b>	<b>Description</b>
Transaction ID	ID of the transaction.
SOAP Processor	SOAP Processor to insert the transaction into the database.
Status	Status of the transaction.
User	User who initiated the transaction.
Start Time	Time at which the transaction started.
Duration	Time elapsed since the transaction started.

The following table describes the fields in the Operation Grid section.

<b>Field</b>	<b>Description</b>
Operation ID	ID of the operation in the transaction.
Transaction ID	ID of the transaction.
Status	Status of the operation.
Component	Component (CoBOC or Rule Engine) that has performed the operation.

The following table describes the fields in the CoBOC Operation Grid section.

<b>Field</b>	<b>Description</b>
Object ID	ID of the object.
Operation	Operation performed by the object.
Date	Date on which the operation was performed.
Object Details	New value of the object after the operation.

The following table describes the fields in the Rule Engine Operation Grid section.

<b>Field</b>	<b>Description</b>
Rule ID	ID of the rule.
Version	Version of the rule.
Date	Date on which the rule was triggered.
Rule Action	Type of rule action.

## Template interface

**Note:** Object Template is deprecated in Cordys BOP 4.1. Use the WS-AppServer Custom class instead.

---

**Important:** For backward compliance, Cordys BOP 4.1 still supports existing object templates, that is, existing templates upgraded from earlier AppWorks Platform releases. However, users can no longer create an object template. Existing applications that use an object template must adopt the suggested alternative because future releases may not support object templates and the CoBOC feature. These features might be removed from AppWorks Platform. The descriptions provided for the template features in this topic are applicable only for the existing templates.

### **Common properties**

You can set an object template's properties while creating it. The following table describes the fields on the Template Properties tab.

Field	Description
Schema Fragment	Refers to the schema fragment to build this object template. Click  for Schema Fragment, select the required schema fragment from the BOD dialog box that opens, and then click <b>OK</b> .
Plugin	Defines the persistence logic of the template. It is the Java interface that handles customized database interactions for the instance of this template. This interface is responsible for handling all database interactions such as insert, delete, and update operations on the template. Type DEFAULT if you want CoBOC to persist business objects (data and metadata) in the CoBOC database.

### **Advanced properties**

The following table describes the advanced properties of the object template.

Field	Select to
Validate Schema	Validate objects of the template against the schema.
Execute Rule(s)	Enable rules to execute for objects of the template.
Persist in DB	Persist business objects of the template in the applicable store, based on the plug-in property. If this option is not selected, the objects are active until the transaction is complete and thereafter removed from the cache.
Audit	Audit operations of the objects based on that template. This enables you to gain access to details of operations carried out on an object. The details of the operations are available in the CoBOC Transaction Monitor.

### **To view the XML structure of the object template:**

- Click **View Schema** to view the XML structure of the object template.  
The schema of the object template is displayed in the schema editor.

---

## Configuration option for SOAP message

**No Reply** - To invoke a SOAP request without expecting a reply, select the **No Reply** check box. For example, few notification messages can be sent and need not be acknowledged on their receipt. Hence, a reply is not expected in such cases.

## Configuring AppWorks Platform security

The various security features in AppWorks Platform can be administered through the Security Properties window of the Management Console.

The Security Policy feature allows administrators to set the security policy tree for an authenticated user. This policy lists the SOAP services and Web service interfaces on which access permissions are set. It enables only those users who are granted permissions to access the respective SOAP services and Web service interfaces.

## Running multiple service containers in a single JVM

A service container created on AppWorks Platform occupies a Java Virtual Machine (JVM). This results in performance issues, since the number of service containers running concurrently drastically increases the number of JVMs and the memory that they require.

To address this, AppWorks Platform enables running multiple service containers in a single Java Virtual Machine (JVM). To enable this feature, you can create and modify OS processes. Once these OS processes are configured within a monitor service, they start every time the monitor starts. Through this feature, the system administrator can create OS processes for the JVMs. When a new service container is created, the user can specify whether the service container must run in a new or existing JVM by choosing the OS process for the service container.

**Note:**

It is not advisable to configure a service container to run in the same JVM as the Monitor service container. This is because if a service container crashes, for any reason, all the service containers running within the same JVM also crash. Therefore, it is recommended that the Monitor always run in an independent JVM.

Assume that the classpath of an OS process is changed. To reflect the changes, all the service containers within the OS process must be stopped and started. This also means that when a service container is moved from one OS process to another and the classpath settings of the service container are not already included in the OS process, all the service containers within the OS process must be stopped and started.

When you add or change an OS process, you must restart AppWorks Platform (<instance name>).

# Configuring certification authorities in the trust store

Before users try to sign in using their certificates, the certification authorities (CA) associated with those certificates must be configured in the trust store of the server. If this step is skipped, users cannot sign in using their digital certificates.

## To configure certification authorities in the trust store:

1. Click **Start > Run** and type **mmc**, and then press **ENTER**.  
The Console window opens.
2. Click **File > Add/Remove Snap-in**.  
The Add/Remove Snap-in dialog box opens.
3. Click **Add**.  
The Add Standalone Snap-in dialog box opens.
4. Select **Certificates** from the list, and then click **Add**.  
The Certificates snap-in wizard opens.
  - a. Select the Computer account, and then click **Next**.
  - b. Continue with the default options, and then click **Finish**.
  - c. Click **Close** to return to the Add/Remove Snap-in dialog box.
5. Click **OK** to return to the console.
6. In the tree structure of the Console window, expand **Certificates**, right-click **Trusted Root Certification Authorities**, and then click **All Tasks > Import**.  
The Certificate Import Wizard opens.
7. Follow the wizard to complete importing the root CAs.  
The certificates are imported.

**Note:** You must import intermediate CAs as well. Follow the above procedure and in step 6, right-click **Intermediate Certification Authorities**, and then complete step 7.

# Configuring security settings

You must configure the security settings on the client machine before you start a Baan IVc session from a business process or from a notification task.

## To configure security settings:

1. In Internet Explorer, click **Tools > Internet Options**.  
The Internet Options dialog box opens.
2. Click the **Security** tab.
3. Select **Trusted sites** as the web content zone.
4. Click **Sites**.  
The Trusted sites dialog box opens.

5. Type **https://www.cordys.com** in Add this Web site to the zone, and then click **Add**.  
The URL is added to the list of websites.
6. Click **OK**.
7. Click **Custom Level**.  
The Security Settings dialog box opens.
8. Select **Enable** under ActiveX controls and plug-ins, and then click **OK**.  
ActiveX controls and plug-ins are enabled.  
**Note:** You can also select Local Intranet instead of Trusted sites to enable these settings.  
The security settings are configured.

## Configuring security settings for external Web services

To access an external Web service from the local intranet zone, you must change the security settings of your browser.

### To configure security settings for an external Web service:

1. In Internet Explorer, on the Tools menu, click **Internet Options**.  
The Internet Options dialog box opens.
2. Click the **Security** tab, and then click the Local intranet zone.
3. Under Security level for the zone, click **Custom Level**.  
The Security Settings dialog box opens.
4. Under the Miscellaneous section, select **Enable** or **Prompt** for Access data sources across domains.
5. Click **OK**, and then click **Apply** for the changes to take effect.

## Configuring the XForms service container

You can manage the response time of the XForms service container by specifying the cache size and the time interval for content expiry. Cache refers to the location where frequently-accessed content is stored.

You can specify the time period for which requested content is retrieved from the browser cache rather than from the server. On requesting an XForm for the first time, the .caf file is converted to HTML, which is stored in the server cache and rendered in the browser. On receiving subsequent requests for the same XForm, the HTML file is rendered directly from the server cache. This eliminates the need to convert .caf files to HTML files unless you modify an XForm and publish it to runtime.

### To configure the XForms service container:

1. On the Welcome page, click  (System Resource Manager).

- 
2. Right-click **XForms**, and then select **Properties**.  
The Properties - XForms window opens. The General, JRE Configuration, Log Settings, XForms, and Foundation Framework tabs are displayed.
  3. Click the **XForms** tab to specify settings for content expiry and cache size.
  4. Select an option in the Content Expiry area to specify the time period for which the cache must be maintained on your computer.
    - Select **Immediately** to retrieve the HTML for the .caf file.  
All requests are sent to the back end, and the HTML is returned with the 200 status code. Selecting this option disables Cache size, indicating that no cache will be maintained.
    - Select **Everytime** to simultaneously delete the cache and ensure that all requests go to the back end.  
The XForms cache returns the 304 status code if its last modified date is greater than the last modified date of the .caf in the XML store; otherwise, it returns the HTML for the .caf with the 200 status code.
    - Select **After** and type a numerical value in **days** to specify the number of days after which to delete the cache. The XForms Gateway sets the cache expiry from the date specified, and requests to the back end are sent only after the specified date.
    - Select **On** and select an option from the calendar control to specify the date to delete the cache.
  5. When you request an XForm or refresh it at runtime, the XForms Gateway sets the modification date of the XForm that is available in the cache as its last modified date. This date is used to identify and retrieve the latest version of the XForm from the backend.  
Depending on the selected content expiry option, the cache on your computer is periodically deleted and refreshed from the back end when you send a request. By selecting the appropriate option, you can ensure access to the latest information from the back end.
- Note:** Cache expiry also depends on the option set for Check for newer versions of stored pages in the Internet Options dialog box of Internet Explorer. For more information, see [How Internet Explorer cache settings affect Web browsing](#).
6. Enter a numerical value in Cache size to specify the maximum number of XForms to cache.  
The cache size must be set depending on various conditions such as the computer hardware and the application's deployment size, extent of usage, and desired performance level. As these factors differ, the optimal cache size is specific to each installation.  
For example, a small cache value might render the cache ineffective in cases where a large number of XForms are accessed and used. On the other hand, a large cache value consumes limited system resources and might affect performance. In this case, to define an optimal cache size, you must take into account factors such as the resources available and the scale of Web service operations.

- 
7. To select a free port number to enable cache coherence manually, select **Manual** and provide an unused free port number in Port Number. By default, the Manual field is Not selected and the framework automatically assigns an unused free port number.

**Note:** You must restart the service container to apply the change.

8. Enter the interceptor implementation class names in Class Name. These entries must be separated by the OS-specific classpath delimiters.
9. Click .

The configuration specified for the XForms service container is saved.

**Note:** For information about the settings required on the General and Log Settings tabs, see Service container configuration interface in the *AppWorks Platform Advanced Development Guide* and [Logging configuration interface](#) respectively.

**After you complete this task:**

You must restart the XForms service container to affect the modified service container configuration.

## Configuring trusted certificates for WS-security

SAML assertions are signed using the corresponding identity providers' signing certificate. These signatures follow the XML Signature standards. See [XML Signature Syntax and Processing](#). The assertion contains the public key of the corresponding private key used for signing. For example, in AppWorks Platform, SSO is the signs assertions and the `<KeyInfo>` tag of the assertion contains the SSO service's X509 certificate information. For verifying the signature, the information in the Keyinfo tag is used. But as per the XML Signature standards, the presence of this tag is not mandatory. If Keyinfo is absent, there must be a mechanism for selecting a default certificate to use. To accomplish this, the administrator must configure the following properties in the `wcp.properties` file:

- `com.eibus.security.x509.validation.certificate.default=<alias of the default certificate>`

**Note:** A certificate with this alias must be present in the list of certificates in a trust relation.

- `com.eibus.security.x509.validation.certificate.organizationfallback=<true or false>`

If the `com.eibus.security.x509.validation.certificate.organizationfallback` property is `true`, the trusted certificates for an organization, that is the organization from which the SOAP request is sent, are retrieved from the trust store and assertions are validated with the corresponding certificate. This trust store is a fallback. If this property is `false`, `com.eibus.security.x509.validation.certificate.default=<default certificate alias>` is verified in the `wcp.properties` file. For more information on adding a property to the `wcp.properties` file, see Adding a new property in the *AppWorks Platform Advanced Development Guide*. If this property too is not configured, the list of certificates in the trust relation is used for validation.

**Note:** It is recommended to configure the `com.eibus.security.x509.validation.certificate.organizationfallback` property when each organization trusts a certificate.

### To configure trusted certificates for organizations:

1. Add the following property to the `wcp.properties` file:

```
com.eibus.security.x509.validation.certificate.organizationfallback =  
true
```

See Adding a new property in the *AppWorks Platform Advanced Development Guide*.

2. Restart OpenText AppWorks Platform (<instance name>).

3. Select the **AddOrganizationSpecificTrust** Web service operation from the SecAdmin Web service interface in the SOAP Service Test Tool from the user's organization from which the SOAP request is sent. See Testing web service operations using service test tool in the *AppWorks Platform Advanced Development Guide*.

```
<SOAP:Envelope xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/">  
    <SOAP:Body>  
        <AddOrganizationSpecificTrust  
            xmlns="http://schemas.cordys.com/1.0/secadmin">  
            <data>  
                <fallBackCertificate>  
                    <X509data>-----BEGIN CERTIFICATE-----  
MIIB4DCCAUmAwIBAgIQUo/TnAQJT5q+C8ItaZiz...SSO Certificate -----END CERTIFICATE--  
---</X509data>  
                </fallBackCertificate>  
                <certificateChain>  
                    <certificate>  
                        <X509data>-----BEGIN CERTIFICATE-----  
MIIB/jCCAWegAwIBAgI.... Certificate Chain of SSO certificate -----END  
CERTIFICATE-----</X509data>  
                    </certificate>  
                    <certificate>  
                        <X509data>-----BEGIN CERTIFICATE-----  
MIIB/jCCAWegAwIBAgI.... Certificate Chain of SSO certificate -----END  
CERTIFICATE-</X509data>  
                    </certificate>  
                </certificateChain>  
            </data>  
        </AddOrganizationSpecificTrust>  
    </SOAP:Body>  
</SOAP:Envelope>
```

The trusted certificate details are configured for the organization.

## Contingency management

This topic discusses various situations that you may encounter while managing your AppWorks Platform licenses and specific solutions.

## License key expiration

The following scenarios explain the license key expiration.

- The demo license key is valid only for a trial period. To continue using the applications, customers must send a usage report to the AppWorks Platform License Service and obtain a new key. If customers do not send the usage report on time, the service containers cannot be started, and therefore, customers cannot work with the applications.
- The operational license key is supplied by the AppWorks Platform License Service on receipt of a usage report. Customers must send usage reports at regular intervals (monthly) to the AppWorks Platform License Service and obtain these keys. If customers do not send the usage report on time, they can continue using the applications, but a warning is displayed during each transaction, stating that the license has expired. Customers can either click Continue to continue using the applications with the warnings or click Renew License to open the License Manager from where the license can be renewed.

**Note:** When license keys are updated by the Standalone/Master or through the Management Console, the OpenText AppWorks Platform (<instance names>) and the service containers must be restarted.

## Invalid usage report

Following are some instances and consequences of invalid usage reports being sent to the AppWorks Platform License Service.

- The Contributor sends an old report to the Master: Since every report has a date and time stamp, the Master recognizes the usage report as an old one and does not consolidate it. The master report, therefore, does not contain the details of that particular Contributor. When the master report is sent to the AppWorks Platform License Service, the report is read, and license keys are issued to all the Contributors of that report. Therefore, the Contributor that sent an old report does not receive a license key.
- The Contributor does not send the usage report to the Master: The Master report does not contain the details of a particular Contributor. When the Master report is sent to the AppWorks Platform License Service, the report is read, and license keys are issued to all the Contributors of that report. The Contributor that does not send a usage report does not receive a license key.

## Internet failure

In case of an Internet failure (when the usage report cannot be sent directly to the AppWorks Platform License Service through HTTP), the usage report must be sent to the AppWorks Platform License Service through email. For this purpose, the usage report can be taken from the folder where it is stored. By default, the usage report is stored in the folder specified below:

- **Standalone or Contributor computer:** The report is stored in <AppWorks\_Platform\_installdir>\license\log\report.
- **Master computer:** The master report is stored in <AppWorks\_Platform\_installdir>\license\master\log\report. The contributor reports are stored in <AppWorks\_Platform\_installdir>\license\masterstore.

The user can change the default folder (<AppWorks\_Platform\_installdir>) to store the usage report by changing the corresponding property in the wcp.properties file. The property to change in each case is listed:

- **Standalone or Contributor computer:** The folder to be specified in com.eibus.license.LogDir.report.
- **Master computer:** The folder for Master reports to be specified in com.eibus.license.MasterLogDir.report. The folder for Contributor reports to be specified in com.eibus.license.LicMasterStoreDir.

After the usage report is sent, the license key must be obtained in a similar manner and updated using the License Configuration Tool of the Management Console. See [Management console](#).

## Master computer becomes unresponsive

When the Master computer becomes unresponsive, one of the Contributors must be made the Master.

### To make the Contributor as the Master:

1. Ensure that the user has administrative privileges in the computer that must be configured as the Master.
2. Create the following directory structure under the <AppWorks\_Platform\_installdir>:
  - AppWorks\_Platform\_installdir>\license\master\log\report
  - AppWorks\_Platform\_installdir>\license\master\log\key
  - AppWorks\_Platform\_installdir>\license\masterstore
3. Create a file named contributors.xml in the master store directory as follows:

```
<contributors> <contributor> <computer>master computer name</computer>
</contributor> </contributors>
```

4. Attach the LicenseMaster method set of the AppWorks Platform ESB Server to the Monitor service group. Restart the monitor.
5. Go to wcp.properties and set the following property to the ID of the user who is the administrator in both AppWorks Platform and the current computer:  
com.eibus.web.license.LicReportGateway.licuser.

6. Go to the Management Console, and then click **Content Management**. In the `licenseconfig` attribute of the `licinfo` entry, change the `<type>` to `master`.
7. Go to the License Configuration Tool of the Management Console of the Master and Contributors, and change the name of the Master computer.
8. Save the changes.
9. Inform other Contributors about the identity of the new Master.
10. Obtain the latest license keys for all the Contributors from the AppWorks Platform License Service.

## Create Class from Object Layout interface

The interface supports filling details for the custom class. The following information is classified keeping in view the creation of different custom classes: pure, derived, inherited, and nested (class within class) for pure custom classes and derived custom classes. Similarly, you can add different types of attributes to a class: simple, derived, and aggregated.

### Creating custom classes

The following table lists the fields to create a pure custom class.

<b>Field</b>	<b>Description</b>
Name	Name of the pure custom class.
Type	Type of the custom class. Select <b>Custom</b> .

The following table lists the fields to create a derived custom class.

<b>Field</b>	<b>Description</b>
Name	Name of the derived custom class.
Type	Type of the class. Select <b>Derived</b> .
Package	Name of the package where this class is created. Click  to select the package.
Derived From	Root class from which the new class is derived. Select the class from the list.

The following table lists the fields to create an inherited custom class.

<b>Field</b>	<b>Description</b>
Name	Name of the inherited custom class.

<b>Field</b>	<b>Description</b>
Type	Type of the custom class. Select <b>Inherits</b> .
Package	Name of the package where this class is created. Click  to select the package.
Inherits	Class from which the new class is inherited. Select the class from the list.

## Creating nested classes

The following table lists the fields to create a nested custom class, which is a custom class within a pure custom class.

<b>Field</b>	<b>Description</b>
Name	Name of the nested custom class.
Occurrence	Occurrence of the nested custom class. Select: <ul style="list-style-type: none"><li>■ 1: single occurrence</li><li>■ *: multiple occurrences</li></ul>

The following table lists the fields to create a nested derived class, which is a derived custom class within a derived custom class. You can create the nested derived class only if the root class is a derived class.

<b>Field</b>	<b>Description</b>
Name	Name of the nested derived class.
Derived From	Class from which the new class is derived. Select the class from the list.
Occurrence	Occurrence of the nested derived class. Select: <ul style="list-style-type: none"><li>■ 1: single occurrence</li><li>■ *: multiple occurrences</li></ul>

## Adding attributes to classes

The following table lists the fields to create a simple attribute for a custom class.

Field	Description
Name	Name of the simple attribute.
Type	Type of the attribute. Select <b>Simple</b> .
EISName	<p>Name of the attribute as it exists in an external Enterprise Information System (EIS) to refer and use in the custom class.</p> <p><b>Note:</b> This field is required only if there are references to data in external systems. It is applicable only for the simple attributes of any custom class. The Object Layout displays the custom class containing an attribute with the EISName value as:</p> <pre data-bbox="507 635 1057 762">&lt;Classname&gt;   &lt;attributename eisName="\[name\]"&gt;\[datatype\]&lt;/attributename&gt;     &lt;attribute&gt;</pre> <p>The generated Java code displays this attribute in its BaseClass as part of <code>AttributeInfo</code> code. For information about the methods, see package <code>com.cordys.cpc.bsf.classinfo</code>. <code>AttributeInfo</code> in the WS-AppServer SDK.</p>
Data Type	Data type of the attribute.
Changeability	<p>Changeability property of the attribute. The permissible values are:</p> <ul style="list-style-type: none"> <li>■ Changeable: The value can be modified anytime.</li> <li>■ addOnly: The value can be assigned anytime (during insert or update operation) but only once and cannot be modified subsequently.</li> <li>■ Frozen: The value is assigned during object creation but cannot be modified subsequently.</li> </ul>
Persistence	<p>Mode of persistence. For a custom class, the value is always transient. The attribute does not map to a database column and is not persisted. The value remains in memory as long as the transaction requires it and is removed from the memory the moment the transaction is complete.</p>
Pattern	<p>Pattern that the attribute must match based on the attribute type. It requires a regular expression in which strings of text such as words, numerals, or specific characters bind together.</p> <p>For example, if an email ID must be entered in a specific format, you might set the pattern to <code>[a-z]+.+@.[a-z]+</code> (the dot separator and @ are a must for an email ID). During runtime, a check is performed on the entered format against this pattern and results in an error if they do no match.</p>

Field	Description
	Similarly, if an attribute must contain only alphabets a-z (case-sensitive) representing a name, the pattern could be set to [a-z]+. This pattern ensures that no other character such as a numeral is entered to fill that attribute. <b>Note:</b> This property is applicable for all data types except ui1, dateTime, bin.base64, boolean, and guid.
Scale	Number of digits to the right of the decimal point. This property is not applicable if you select String as the data type.
Precision	Total number of digits used.
Initial	Initial value of the attribute, applicable only for validate requests.
Min Length	Minimum length of the attribute. This is applicable only for the String data type.
Max Length	Maximum length of the attribute. This is applicable only for the String data type.
Min Inclusive	Lower limit of the value, including the value. This is applicable for all data types excluding String.
Min Exclusive	Lower limit of the value, excluding the value. This is applicable for all data types excluding String.
Max Inclusive	Upper limit of the value, including the value. This is applicable for all data types excluding String.
Max Exclusive	Upper limit of the value, excluding the value. This is applicable for all data types excluding String.
Unique	Whether the attribute is unique. If an attribute is marked unique, it becomes an important entity in the execution of the relevant Web service operations, for example, GetObject.
Required	Whether it is necessary to provide a value for the attribute.

The following table lists the fields to create a derived attribute for a custom class.

Field	Description
Name	Name of the derived attribute.
Type	Type of the attribute. Select <b>Derived</b> .
Derived From Attribute	Attribute from which the new attribute is derived.

The following table lists the fields to create an attribute of type aggregation for a custom class.

Field	Description
Name	Name of the attribute.
Type	Type of the attribute. Select <b>Aggregation</b> .
Aggregation	Attribute with which this attribute is aggregated.
Occurrence	Occurrence of the attribute. Select: <ul style="list-style-type: none"><li>■ 1: single occurrence</li><li>■ *: multiple occurrences</li></ul>

## Creating a sample key store

Create a key store and self-signed certificate with the corresponding public or private key.  
For example:

```
keytool -genkey -alias Smith -keyalg RSA -validity 365 -keystore Mykeystore
where Smith is the name of the key and Mykeystore represents the key store file name.
```

See the sample key store.

```
Enter key store password: password What is your first and last name? [Unknown]: Smith
What is the name of your organizational unit? [Unknown]: AppWorks Platform Software
What is the name of your organization? [Unknown]: AppWorks Platform
What is the name of your City or Locality? [Unknown]: Hyderabad
What is the name of your State or Province? [Unknown]: AP
What is the two-letter country code for this unit? [Unknown]: IN
Is CN=Smith, OU=AppWorks Platform Software, O=AppWorks Platform, L=Hyderabad, ST=AP,
C=IN correct? [no]: yes
Enter key password for <Smith> (RETURN if same as key store password): <CR>
```

This is the key store that the server uses.

## Creating a sample trust store

After you create a key store, export the same into the trust store.

### To create a sample trust store:

1. Export and examine the self-signed certificate.

```
keytool -export -alias Smith -keystore Mykeystore -rfc -file smith.ce
where smith.cer is the name of the certificate file.
```

```
Enter key store password: password Certificate stored in file <smith.cer>
```

---

## 2. Import the certificate into a new trust store.

```
keytool -import -alias Smithcert -file Smith.cer -keystore Mytruststore,  
where Smithcert represents the name of the certificate and Mytruststore is the  
name of the trust store.
```

```
Enter key store password: trustword Owner: CN=Smith, OU=Cordys, O=AppWorks  
Platform R&D, L=Hyderabad, ST=AP, C=IN Issuer: CN=Smith, OU=Cordys,  
O=AppWorks Platform R&D, L=Hyderabad, ST=AP, C=IN Serial number: 435dfb6f  
Valid from: Tue Oct 25 15:01:27 GMT+05:30 2005 until: Tue Nov 01 15:01:27 GMT+05  
:30 2005 Certificate fingerprints: MD5:  
CA:CF:D3:28:5B:7E:40:68:1E:FF:C5:BC:00:5F:FC:F8 SHA1:  
F4:B9:23:A9:5A:7E:7F:C3:81:9B:38:C8:F0:51:05:EC:A0:87:18:E7 Trust this  
certificate? [no]: yes Certificate is added to key store.
```

## Creating a schema fragment

You can create a schema fragment.

### To create a schema fragment:

1. Right-click the newly created XML schema from the Workspace Explorer and select **Add Schema Fragment**. Alternatively, open the schema from My Documents, click  (View Children) on the toolbar, and then click in the Add Children pane. The Untitled Business Object Document - Business Object Document wizard opens.
2. Type the required information in the Name and Description fields of the Untitled Business Object Document - Business Object Document wizard, and then click **Finish**. The XML schema editor opens.
3. Create a schema fragment in one of the following ways:

To create a schema fragment using	Do the following
Tree - it is the graphical way of creating a schema fragment	Click the <b>Tree</b> tab. For information about the schema editor contents, see Schema editor in the <i>AppWorks Platform Advanced Development Guide</i> . In the toolbox, click the element to include in the schema. Specify the necessary properties for the element in the Properties section.
Text	Click the <b>Text</b> tab, and type the schema for the template.
Instance	Click the <b>Instance</b> tab, and type the XML structure of the template. While creating the schema, OpenText recommends not providing values or defining attributes for XML tags because they will be ignored.

---

**Note:** You can synchronize the Tree and Text views of the XML Schema. Any changes you make on the Tree tab reflect on the Text tab and vice versa.

## Creating an email template

Use the create email template to send the process and related contextual information to the user, when a specified condition based on a business objective is fulfilled.

In the email template, you can include the information, which is relevant to the business process, user, and required response of external web services. You can customize the contents of the email template based on the requirements such as the background color, font size, and formatting.

The components of the email template can be modeled through changing the HTML code. The attributes related to the business event response, such as process attributes and contextual attributes are available in the **Message** area of the Message data tree of the email template. These attributes can be used in any of the above components by placing them in the specific area.

The email template can be exposed as a Web service. It can also be used to define conditions to trigger actions, by invoking Web service. The created email template is available as a Web service in the custom space, which is available in the current user organization as **Business Event Response**. The user must select the method while invoking the Web service in the **Conditions** and **Actions** section.

**Note:** For more information on the email model template, see Creating a delivery model in the *AppWorks Platform Advanced Development Guide*.

### Important:

- The email template integration with Business Activity Monitoring (BAM) handles the Message Template creation.
- The Namespaces tab is not required for BAM.
- The Customizing Email Models and Custom Method Generator Interface features are not used in BAM.
- Currently, the Message Metadata (E-mail Template > Model Source > Message > Data > Message Metadata) section on the Model Source tab page is not used. If you select Message Metadata, the email template does not change.
- Build rules (conditions) and define actions step is available as the Step 5 of 6 on the Business Event Response Composer. See Define conditions and actions in the *AppWorks Platform Advanced Development Guide*.

**Note:** If the leadtime is included in the email template, and LeadTimeGreaterthan or LeadTimeGreaterthanEqualto is the selected condition template, the value displayed is the elapsed time of the process or activity.

## Custom settings

The JMX console provides an overview of the JVM instances of the system resources being deployed. It provides a graphical illustration of the memory usage, resource consumption, and performance of the system resources running on AppWorks Platform. The console plays a key role in monitoring the health and performance of AppWorks Platform.

There are many properties that compliment the view of performance counters defined by an application. One such property is Moving Average Window size, which defines the time frame or slots that contain the event data. By increasing the size of the Moving Average Window, the counter's data granularity is enhanced. When the data is plotted as a graph over time, the larger the Moving Average Window size, the better is the smoothness of a graph.

You can set the length of Moving Average Window by setting the `com.cordys.management.counters.mawlength` in the `wcp.properties` file. See [Global configuration properties for AppWorks Platform](#).

By default, the length of the Moving average window is 60 seconds and the minimum length is 5 seconds.

## Custom views

A custom view provides a view on the content to monitor in a process, for example, Total number of orders waiting for Approval for Gold customers. You can define a view on a process by using a business measure built through the Business Measure Editor or a KPI that is shown as a KPI trend. Custom views are made available as composite controls on business measures or KPI creation.

A user can view the dashboard only if the user or role has access permissions on all the underlying Web services of the business measure or KPI objects used in the dashboard. These Web services are available with the composite controls of the business measure or KPI.

## Database configuration for applications

Few applications require a database configuration, where you must configure the database details. Consider the following while providing the details:

- If any applications require a database configuration, the Application Installer wizard displays the Provide required inputs page that lists the applications on separate tabs to configure the database details.
- If required, you can skip this step of providing database inputs. Post installation, you must run the database scripts and configure the database.

- 
- You can either use the existing database configuration by selecting a database configuration from a list or create a database configuration. See [Creating a database configuration](#).
  - Select the Use same inputs for a machine option to apply the same details to all the databases. To apply these details across multiple computers, select the Use same inputs for all machines option.
  - Enter the appropriate details in the fields available on the respective tabs and click Next. For information about installing CoBOC templates, data transformations, rules and rule groups, and notification models, see [Deployment Descriptor interface](#).
  - Select Create Database Tables and Create Service Group Configuration options to create tables and service groups automatically.

## Deleting a certificate

### To delete a certificate:

1. On the Welcome page, click **Security Administration**.  
The Security Administration window opens. The certificates are displayed on the Certificates tab.
2. Select the certificate to delete, and then click **Delete**.  
An alert appears stating that the certificate is deleted.

## Deleting a trust relation

### To delete a trust relation:

1. On the Welcome page, click **Security Administration**.  
The Security Administration window opens.
2. Click **Service Group Trust Configuration**.  
The service groups belonging to a collection are displayed with the description of the SOAP nodes group. All the certificates associated with the SOAP nodes group are displayed below the description.
3. Select the trust relation to delete, and then click **Delete**.  
An alert appears stating that the trust relation is deleted.

## Deploying applications

An application package is a bundle of documents that together form a working application. It can contain components such as tasks, roles, Web service operations, web pages, XML content, and XDS content. With the help of an application package, you can ship the application content across any installation.

The following topics include information about deploying applications:

- **Managing applications:** Step-by-step procedures to install, uninstall, upgrade, and upload applications.
- **Customizing application installer wizard:** Procedures to customize the application installation by specifying custom pages and the Operating System compatibility.

## Deployment Descriptor interface

The following table describes the information to be specified on the Provide required inputs page of the Application Installer wizard.

The Deployment Descriptor interface contains the following fields.

Field	Description
Select Organization	<p>Select the organization to deploy the application.</p> <p><b>Note:</b> The organizations displayed in the list are based on the selected CoBOC service.</p>
Select Service Container	<p>Specify the service to use for installing the application. Click , select the required service container from the Select a Service Container dialog box, and then click <b>OK</b>.</p>
Overwrite Existing Content	<p>Determine how the application must be loaded.</p> <ul style="list-style-type: none"> <li>■ If you are loading the package for the first time, click <b>No</b>. During deployment, if there is any conflict at the database layer, the application does not install.</li> <li>■ If you are loading a service pack of a previously deployed application, click <b>Yes</b>. During deployment, if there is any conflict at the database layer, the entries that were previously deployed are deleted, and the new content is registered, thereby making the service pack operational.</li> </ul> <p><b>Note:</b> The integrity of the content must be considered while installing the applications.</p>
Deployment Proxy User	<p>When you install the application, a request is sent to the relevant service container. The request can only be sent through a user who has permissions to use the relevant Web</p>

Field	Description
	service interface. You can select the user from this list. The users in this list are based on the selected organization.

## Diagnostic options for LDAP entries

Follow are the diagnostic options for LDAP entries.

Normalization	Normalization is the process of changing LDAP entries to normal distinguished names (RFC 1485 format). A distinguished name uniquely identifies an entry in the LDAP directory.
Referential Integrity	<p>A referential integrity check is conducted to verify a set of rules based on the AppWorks Platform LDAP schema. It checks whether the relationship between the entries and their attributes is valid. The following checks are performed as part of referential integrity:</p> <ul style="list-style-type: none"> <li>■ ACL Integrity Check: Checks if the ACL and service attributes point to a distinguished name.</li> <li>■ Service Container Integrity Check: Checks if there is at least one service container under each service group, and at least one connection point under each service container. If not, a warning message is displayed. The diagnostic tool does not check the service container and the connection point values.</li> <li>■ Service Group Integrity Check: Checks whether the Service Group contains the properties (or attributes), labeled URI (namespace), and busmethodset as specified in the AppWorks Platform schema. These attribute values are associated with Application busmethodsets and the busmethods. After the check is performed, the results</li> </ul>

	<p>are published with appropriate messages in the Event Viewer.</p> <ul style="list-style-type: none"> <li>■ <b>User-Role Integrity Check:</b> Checks for references based on subscribed users for association of busorganizationalrole and busauthenticationuser (bi-directional) and for the authentication user specified for all the organization users.</li> </ul>
Cyclic References	Checks for LDAP entries which hold references to each other.
Invalid Entries	Checks for invalid entries and junk data in the AppWorks Platform Directory Search Root on the basis of the LDAP schema.
Schema	Validates the schema.
URI Scanner	Identifies service containers that are configured to the same port on the same computer. The tool displays the DN of the connection points that share the same labeled URI. This is applicable only to TCP/IP, that is to socket and zip socket.

## Editing a custom application connector

You can change the configuration details such as logging categories, startup dependencies, and the registered Web service implementation of custom application connectors.

### Before you begin:

Configure a custom connector. See Creating and configuring a custom connector in the *AppWorks Platform Advanced Development Guide*.

### To edit a custom application connector:

1. In My Recent Documents or the Explorer view of the Workspace Documents window, double-click the connector. For information about changing the view, see Working with Workspace Documents.  
The <Application Connector Name>-Application Connector window opens.
2. In the <Application Connector Name>-Application Connector window:

To change	Steps
Connector details	a. Click  (Quick Access Menu) > Properties.

To change	Steps
	<p>The Properties - Document Properties dialog box opens.</p> <p>b. On the General tab, change the necessary details, and then click <b>OK</b>.</p>
Configuration details	<ul style="list-style-type: none"> <li>■ See Configuration parameters for custom application connectors in the <i>AppWorks Platform Advanced Development Guide</i>.</li> </ul>

3. Click .
4. Publish the updated connector to make it available for the current organization. Do one of the following:
  - a. In the Quick Access Menu, select **Publish**.
  - b. In the Explorer view of the Workspace Documents window , right-click the connector, and then select **Publish to Organization**.
5. Change the necessary details and click .

**Note:** To delete a connector, do one of the following:

- In the My Recent Documents view, mouse over the connector, click to the right of the connector name, and then select **Delete** from the context menu.
- In the Explorer view, go to the connector, right-click it, and then select **Delete**.

## Editing role details

You can edit the role name and roles associated with it.

### Before you begin:

You must have the role of a systemAdmin or organizationalAdmin to edit role details.

### To edit role details:

1. On the Welcome page, click  (User Manager).  
The User Manager window opens.
2. Select one of the following views:

View	Description
Roles - Roles	See <a href="#">Roles - Roles view</a> .
Roles - Users	See <a href="#">Roles - Users view</a> .
Roles - Tasks	See <a href="#">Roles - Tasks view</a> .

- 
3. Right-click a role and select **Edit**.

The Role Details window opens.

4. Edit the details, and then click .

**Note:** Only organizational roles can be edited. Application roles and the systemAdmin role cannot be edited.

## Editing user details

Depending on the type of user selected, the appropriate details are displayed to edit. For information about the user types, see [Creating users](#) .

### Before you begin:

You must have the role of a systemAdmin to edit user details.

### To edit user details:

1. On the Welcome page, click  (User Manager).  
The User Manager window opens.
2. Select one of the following views:

View	Description
Users - Roles	See <a href="#">Users - Roles view</a> .
Users - Tasks	See <a href="#">Users - Tasks view</a> .

3. In the Users pane, right-click a user to edit and select **Edit**.  
The User Details window opens.
4. Edit the details, and then click .

## Email modeler

Email modeler has a rich text editor with a toolbar and a rich editing text area. Following are some of its important features:

- Flexible font styles, sizes, boldfacing, and color schemes
- Ability to embed tables
- Ability to indent and align text
- Ability to drag and drop message data from messages

To personalize and model messages that are sent to emails, click the required XML element and make the necessary changes.

To modify the style, logo, or layout of the model, click **Email Model**. To preview your changes, click **Preview**.

The following table describes the XML elements that are available on the Email Modeler interface.

XML element	Description
Signature	Text and pictures that are automatically added to the end of an outgoing email message.
Salutation	Title of the person, for example, Mr., Mrs, or Miss.
Header	Area in the top margin of a page.
Footer	Area in the bottom margin of a page.
Body	<p>Content of the message. You can type the required message in the text box and also use the XML elements from the model tree structure of the message data.</p> <p>Drag the elements to the text box and build the required message. The XPath of the elements are displayed in the text box, complying to the XPath 1.0 standards.</p>
Application details	<p>Link of the default application to launch. Type the URL of the application in Application URL.</p> <p><b>Note:</b></p> <ul style="list-style-type: none"><li>■ If you are loading an XForm application, the path of the XForm is displayed by default. Modify the URL to display other applications.</li><li>■ If you are loading HTML or other applications, you can enter both the absolute path and the relative path. The relative path must have the URL prefix that was entered while configuring the notification processor.</li><li>■ Select <b>Image (path)</b> and type the path of the image in the text box. Alternatively, you can select <b>Text</b> and type the text as a link to launch the application.</li></ul>

## Embedding Inbox as a URL

The Inbox can be embedded into an application by providing its URL. You can display the tasks sent to a worklist, team, or role. You can also display the Inbox containing tasks in various states or further drill down to display the tasks that are assigned to a user and the tasks that the user has not started yet by appending the URL with relevant query strings.

To embed an Inbox into an application, you must supply the URL of the Inbox as shown below.

```
http://<cordysserverurl>:<port>/cordys/com/cordys/notification/workflow/worklistinbox.caf
```

For example, the following sample code displays the worklist section of the Inbox in your application.

```
http://cordysserver.mydomain.com/cordys/com/cordys/notification/workflow/worklistinbox.caf
```

By appending the URL with a set of query parameters, you can selectively display the tasks in the Inbox. For example, you can display all the tasks in a worklist by appending the URL with a query string such as `tw=bc5aaaf8d-8943-4b8c-8395-49f393c90e47` as shown in the following sample.

```
http://cordysserver.mydomain.com/cordys/com/cordys/notification/workflow/worklistinbox.caf?tw=bc5aaaf8d-8943-4b8c-8395-49f393c90e47
```

The following query parameters can be used with the Inbox URL:

Parameter	Displays
tw	Tasks in the worklist. Enter the worklist ID.
tt	Tasks in a team. Enter the organization unit ID.
tr	Tasks of a role. Enter the role DN.
sn	Tasks filtered by the source. Enter the name of the business process or case.
an	Tasks filtered by activity. Enter the name of the business process activity or case activity.
adn	Tasks filtered by assignee. Enter the DN of the user to whom this task has been assigned.
sid	Tasks filtered by Source Instance ID. Enter the business process instance ID or case instance ID.
aid	Tasks filtered by activity ID. Enter the activity ID.
ts	Tasks filtered by task state. Enter one of the following task states: <ul style="list-style-type: none"><li>■ NEW</li><li>■ ASSIGNED</li><li>■ INPROGRESS</li><li>■ SUSPENDED</li><li>■ COMPLETED</li><li>■ PAUSED</li><li>■ OBSOLETE</li></ul>
py	Tasks filtered by priority. Enter the priority values:

<b>Parameter</b>	<b>Displays</b>
	<ul style="list-style-type: none"> <li>■ py=1 for LOW priority tasks</li> <li>■ py=3 for MEDIUM priority tasks</li> <li>■ py=5 for HIGH priority tasks</li> </ul>
ddt	<p>Tasks filtered by due date. Enter the due date in the YYYY-MM-DDTHH:mm:ss.ms format.</p> <p>For example, to display only those tasks that are due on 14th April 2010, the URL must be appended with ?ddt=2010-04-14T18:30:00.0.</p>
sdn	Tasks filtered by sender. Enter the sender DN.
sdt	Tasks filtered by start date. Enter the start date in the YYYY-MM-DDTHH:mm:ss.ms format, for example, sdt=2010-04-14T18:30:00.0.
rdt	Tasks filtered by received date. Enter the received date in the YYYY-MM-DDTHH:mm:ss.ms format, for example, rdt=2010-04-14T18:30:00.0.

## Sample URLs

The following are some of the sample URLs to display an embedded Inbox in an application.

### Show all tasks in a work list

```
http://cordysserver.mydomain.com/cordys/com/cordys/notification/workflow/worklistinbox.caf?tw
=bc5aaaf8d-8943-4b8c-8395-49f393c90e47
```

### Show all tasks of a role

```
http://cordysserver.mydomain.com/cordys/com/cordys/notification/workflow/worklistinbox.caf?tr
=cn=ProjectManager,cn=organizational roles,o=system,cn=cordys,cn=wwip,o=mydomain.com
```

### Show all new tasks in a worklist

```
http://cordysserver.mydomain.com/cordys/com/cordys/notification/workflow/worklistinbox.caf?tw
=bc5aaaf8d-8943-4b8c-8395-49f393c90e47&ts=CREATED
```

### Show all completed tasks in a worklist

```
http://cordysserver.mydomain.com/cordys/com/cordys/notification/workflow/worklistinbox.caf?tw
=bc5aaaf8d-8943-4b8c-8395-49f393c90e47&ts=COMPLETED
```

```
ox.caf?tw  
=bc5aaaf8d-8943-4b8c-8395-49f393c90e47&ts=COMPLETED
```

### Show all high priority and incomplete tasks in a worklist

```
http://cordysserver.mydomain.com/cordys/com/cordys/notification/workflow/worklistinb  
ox.caf?tw  
=bc5aaaf8d-8943-4b8c-8395-49f393c90e47&ts=CREATED&py=5
```

## Enabling Composite Application Logging

AppWorks Platform-based applications use multiple service containers to achieve the functionality. They generate component-specific logs that are stored at different places, which makes it complex to debug problems occurring in an application.

Composite Application Logging (CAL) helps you debug a particular application for information. This helps you diagnose how or what a Web service is executing. In addition, CAL aids in tracing an error or a SOAP fault through a specific log entry in a file. CAL allows you to publish, collect, store, and view log messages pertaining to multiple applications or composite applications.

The following activities help you gain an understanding of the deployment scenarios:

- Using Logging Service in a business process: Business processes are mission critical to any organization. Much before implementing a business process model, it is essential to locate and correct errors in an application to ensure that business process models operate successfully. The business logging feature in AppWorks Platform facilitates you to model log entries for your business process models to enable you to know the statuses of activities at the time of debugging and to address them much before their actual implementation. See [Defining log messages for an activity](#) in the *AppWorks Platform Advanced Development Guide*.
- Viewing log messages: AppWorks Platform provides the Log viewer tool, which helps the administrator search for log messages using search criteria, such as queries and filters. The information gathered from the log files helps in analyzing the error state and fixing support calls. See [Viewing log messages using the Log Viewer](#).

## Enterprise Data Object

The Enterprise Data Object is a persistence object for enterprise applications such as SCM and CRM. Such objects are synchronized to the BAM monitoring hub database by AppWorks Platform Master Data Management (MDM) and are referred to as MDM entities. There are database tables corresponding to each MDM entity in the BAM monitoring hub database. The

---

MDM entities are available while creating business measures if they reside in the same MDM hub database where the MDM hub for BAM resides.

## Extended key usage options

Extended key usage of a digital certificate indicates one or more purposes for which the certified public key may be used in addition to or in place of the basic purposes indicated in the Key Usage field. See [Key usage options](#). Extended key usage has the following options:

- Email Protection
- Client Authentication
- Server Authentication
- Time Stamping
- Code Signing (to sign software files)
- OCSP Signing
- ipsecEndSystem
- ipsecTunnel
- ipsecUser

## Extension class

An Extension class is generated by WS-AppServer and works as a supplement to the Base class (extends from the Base class). Similar to the Base class, an extension class maps to the corresponding database table. If you want to add any business logic to the generated class, you may include it in the Extension class. For example, you may use the Extension class to implement logic pertaining to event listeners. See Event Listener in the *AppWorks Platform API Reference Guide*.

The Extension classes for Standard classes are different from the Extension classes for the Custom classes. The Extension class for a Standard class contains only the BusObject construct. Whereas, the Extension Class for the Custom class contains the BusObject construct as well as the three basic methods, onInsert, onUpdate, and onDelete.

## Gateway Statistics

The Gateway Statistics area contains the following fields.

Field	Description
Total requests	Total number of requests passed through the gateway.
Total processing time	Total time taken for processing the requests.
Average processing time per	Average time taken to process a single request.

<b>Field</b>	<b>Description</b>
request	
Total back end wait time	Time delay at the server while waiting for a response.
Invalid messages	Total number of invalid messages passed through the gateway.
Unknown users	Total number of unknown users accessing the gateway.
Invalid organization requests	Total number of invalid organization requests sent to the gateway.
Messages sent to Server	Total number of messages sent to the gateway.
Response from Server	Total number of replies received from the gateway.
Timed out requests	Total number of timed out requests (replies not received due to time out).
LDAP errors	Total number of LDAP errors encountered.
Errors	Total number of errors that occurred.

**Note:**

- Click Refresh List to refresh the Gateway Statistics information.
- Click Clear Statistics to clear the Gateway Statistics information.

## HTTP connector features

The HTTP connector provides connectivity to Web servers over the HTTP protocol and supports common HTTP operations such as GET, POST, PUT, and DELETE. It can invoke services over HTTP or HTTPS, standard Web services, or REST-based services with XML or JSON payloads.

The HTTP connector has the following features that help applications communicate with external Web servers.

### Multiple server support

Using the same HTTP connector or container, you can connect to multiple hosts and invoke services. These services can be internal to the organization or available on the Internet. The HTTP connector configuration has connection details of various servers, identified by a unique name or ID. While processing the request, you must provide the connection ID that helps the connector identify proper connection details.

See the sample configuration in the XML store.

```

<configurations xmlns="http://httpconnector.opentext.com/1.0/configuration">
    <connections>
        <connection id="CONN-DELICIOUS-FEED">
            <url>http://feeds.delicious.com</url>
        </connection>
    </connections>
    <connections>
        <connection id="PRocess-PLATFORM-TASKS">
            <url>http://myserver:9123/cordys</url>
            <username>testuser</username>
            <password>YzByZHz</password>
        </connection>
    </connections>
</configurations>

```

## Organization-aware

The HTTP connector is organization-aware by default. The same HTTP connector in the shared space or system organization can process requests from organizations or tenants with organization-specific configurations. The HTTP connector configuration is stored in the XML store, which is organization-aware by default. The configuration is selected based on the organization context of the user. If the configuration is available in a specific organization, the HTTP connector uses the configuration available in that organization; otherwise, it uses the configuration available in the shared space.

## Standard web services and REST services support

The HTTP connector follows a pluggable request and response handling architecture. It supports the following:

- Web services with XML payload

<b>Request handler</b>	com.opentext.applicationconnector.httpconnector.impl.StandardRequestHandler
<b>Response handler</b>	com.opentext.applicationconnector.httpconnector.impl.StandardResponseHandler
<b>Supported payloads</b>	XML

See the sample code.

```
<implementation type= "HTTP" xmlns=
```

```

"http://httpconnector.opentext.com/1.0/implementation" xmlns:SOAP=
"http://schemas.xmlsoap.org/soap/envelope/" >
.....
<request-handler class =
"com.opentext.applicationconnector.httpconnector.impl.StandardRequestHandler" />
<response-handler class =
"com.opentext.applicationconnector.httpconnector.impl.StandardResponseHandler" />
....
</implementation >
```

- REST-based services with JSON and XML payloads

<b>Request handler</b>	com.opentext.applicationconnector.httpconnector.impl.RestRequestHandler
<b>Response handler</b>	com.opentext.applicationconnector.httpconnector.impl.RestResponseHandler
<b>Supported payloads</b>	JSON, XML

See the sample code.

```

<implementation type= "HTTP" xmlns=
"http://httpconnector.opentext.com/1.0/implementation" xmlns:SOAP=
"http://schemas.xmlsoap.org/soap/envelope/" >
.....
<request-handler class =
"com.opentext.applicationconnector.httpconnector.impl.RestRequestHandler" />
<response-handler class =
"com.opentext.applicationconnector.httpconnector.impl.RestResponseHandler" />
....
</implementation >
```

## Multiple payload format

HTTP connectors can send input to services or receive output from services either in the XML or JSON format. Since AppWorks Platform supports only SOAP-based Web services, all requests from AppWorks Platform are in SOAP or XML format and the HTTP connector transforms them to the appropriate format.

If the input request is of the type HTTP POST and expects a JSON input, set Content-type=application/json in the request header.

```

<implementation xmlns= "http://httpconnector.opentext.com/1.0/implementation"
xmlns:SOAP= "http://schemas.xmlsoap.org/soap/envelope/" type= "HTTP" >
```

```

.....
<connection-id>JSON-TEST</connection-id>
<request-handler class =
"com.opentext.applicationconnector.httpconnector.impl.RestRequestHandler" >
<req-headers>
<header name= "Content-Type" >application/json</header>
</req-headers>
</request-handler>
<response-handler class =
"com.opentext.applicationconnector.httpconnector.impl.RestResponseHandler" />
<valid-response-code> 200 </valid-response-code>
.....
</implementation>

```

## Request and response transformation capability

You can use XSLT capability to transform a request to another request that is expected by the service or to transform the response from the service to the XML expected by the application. The XSLT can be stored in the XML store and can be used inside the HTTP connector.

```

<implementation xmlns= "http://httpconnector.opentext.com/1.0/implementation"
xmlns:SOAP= "http://schemas.xmlsoap.org/soap/envelope/" type= "HTTP" >
.....
<connection-id>JSON-TEST</connection-id>
<request-handler class =
"com.opentext.applicationconnector.httpconnector.impl.RestRequestHandler" >
<req-headers>
<header name= "Content-Type" >application/json</header>
</req-headers>
<xslt xmlstore= "xslts/request.xslt" />
</request-handler>
<response-handler class =
"com.opentext.applicationconnector.httpconnector.impl.RestResponseHandler" >
<xslt xmlstore= "xslts/response.xslt" />
</response-handler>
<valid-response-code> 200 </valid-response-code>
.....
</implementation>

```

## Authentication support

Though HTTP connector supports basic authentication, it also supports HTTPS.

## Custom HTTP header support

You can add custom HTTP headers to the request while sending the request.

```

<request-handler class =
"com.opentext.applicationconnector.httpconnector.impl.RestResponseHandler" >
...
<req-headers>
<header name= "x-ms-version" > 2012 - 03 - 01 </header>
<header name= "Content-Type" >application/xml</header>
</req-headers>
..
</request-handler>

```

## URI parameters mapping

The HTTP connector supports URI parameters in the request, which can substitute the corresponding placeholders in the URI while sending the request to the Web server.

```

Method implementation
<implementation type="HTTP"
xmlns="http://httpconnector.opentext.com/1.0/implementation"
xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/">
    <connection-id>CONN-DELICIOUS-FEED</connection-id>
    <http-method>GET</http-method>
    <uri>/v2/rss/{0}</uri>
    <request-handler
class="com.opentext.applicationconnector.httpconnector.impl.RestRequestHandler">
        <uri-parameters>
            <parameter type="xpath">./Username</parameter>
        </uri-parameters>
    </request-handler>
    <response-handler
class="com.opentext.applicationconnector.httpconnector.impl.StandardResponseHandle
r"/>
        <valid-response-code>200</valid-response-code>
        <namespaces/>
    </implementation>

```

The URI parameter formats the URI according to the specified format, for example: /v2/rss/{0}. The {0} and {1} parameters are filled with the parameters included in the URI parameters.

**Note:** The connection URI parameters start at 0.

See the SOAP request from the application to the HTTP connector.

```

SOAP Request
<SOAP:Envelope xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/">
    <SOAP:Body>
        <GetBookmarksForUser xmlns="http://com.opentext.httpptest/rest">
            <Username>RestUser</Username>
        </GetBookmarksForUser>
    </SOAP:Body>
</SOAP:Envelope>

```

```
</SOAP:Body>
</SOAP:Envelope>
On execution of the above web service operation, the value of the Username tag is
used as a URL parameter. The actual URL will be <server url in configuration>
/v2/rss/ RestUser
```

## Proxy support

The HTTP connector can connect to the services using a proxy server in between with proper authentication support.

## JMX counters

The following HTTP connector-specific performance counters provide a breakup of the total time spent in processing requests:

- **RequestTransformation:** Time spent in transforming incoming requests to the HTTP requests.
- **HTTPRequestProcessing:** Time spent in waiting for the HTTP response after sending the request.
- **ResponseTransformation:** Time spent in transforming the HTTP responses to the responses that are eventually sent to the client.
- **XMLToJsonTransformation:** Time spent in converting XML in the SOAP request to JSON before sending it to the REST service.
- **JsonToXMLTransformation:** Time spent in converting JSON in the response from the service to XML before sending the response to AppWorks Platform.

The HTTP connector internally uses the AppWorks Platform connector too to send requests to the XML store. It is a managed component and can monitor the performance of the reads on the XML store.

## HTTP connector service contract

Using the HTTP connector, an application on AppWorks Platform uses services that are available over HTTP or HTTPS. These services are HTTP services such as Web services or REST-based services. Since AppWorks Platform follows the SOAP protocol, it can consume the Web services in an understandable format. The HTTP connector transforms the SOAP-based application and AppWorks Platform messages to the format expected by the service and vice versa.

The response from the external HTTP service is added to the Web service response root as XML after the necessary transformations. If a namespace is associated with the HTTP service response, it is preserved when adding it to the Web service response. If no namespace is associated with the HTTP response, the namespace of the root element of the Web service response becomes the namespace of the HTTP response.

When a service is presented to AppWorks Platform, it must follow the following Web service implementation protocol.

Tag	Description
implementation	<p>Required. Root tag of any service executed by the HTTP connector with the following attributes:</p> <ul style="list-style-type: none"> <li>■ type: value is HTTP</li> <li>■ xmlns: <code>http://httpconnector.opentext.com/1.0/implementation</code></li> </ul> <pre data-bbox="409 572 1274 699">&lt;implementation type="HTTP"   xmlns="http://httpconnector.opentext.com/1.0/implementation"&gt;   .... &lt;/implementation&gt;</pre>
connection-id	<p>Required. Connection ID to execute the request. This ID must be present in the HTTP connections configuration stored in the XML store.</p> <pre data-bbox="409 846 1253 1015">&lt;implementation type="HTTP"   xmlns="http://httpconnector.opentext.com/1.0/implementation"   xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/"&gt;   &lt;connection-id&gt;TEST-CONNECTION&lt;/connection-id&gt;   .... &lt;/implementation&gt;</pre>
uri	<p>URI of the external service. This value must be relative to the URI provided in the HTTP connection configuration stored in the XML store. The URI can also be parametrized.</p> <p><b>Web service implementation depicting a static URI</b></p> <pre data-bbox="409 1248 1258 1459">&lt;implementation type="HTTP"   xmlns="http://httpconnector.opentext.com/1.0/implementation"   xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/"&gt;   &lt;connection-id&gt;TEST-CONNECTION&lt;/connection-id&gt;   &lt;uri&gt;/v2/rss/testrest&lt;/uri&gt;   .... &lt;/implementation&gt;</pre> <p><b>Web service implementation depicting a parameterized URI</b></p> <pre data-bbox="409 1586 1258 1733">&lt;implementation type="HTTP"   xmlns="http://httpconnector.opentext.com/1.0/implementation"   xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/"&gt;   &lt;connection-id&gt;TEST-CONNECTION&lt;/connection-id&gt;   &lt;http-method&gt;GET&lt;/http-method&gt;</pre>

Tag	Description
	<pre data-bbox="425 297 1230 608">&lt;uri&gt;/v2/rss/testrest/{0}&lt;/uri&gt; &lt;request-handler class="com.opentext.applicationconnector.httpconnector. impl.RestRequestHandler"&gt;     &lt;uri-parameters&gt;         &lt;parameter type="xpath"&gt;./Username&lt;/parameter&gt;     &lt;/uri-parameters&gt; &lt;/request-handler&gt; ..... &lt;/implementation&gt;</pre>
	<p>While processing the request, the actual URI is constructed by substituting the value of the <code>Username</code> parameter from the request.</p>
http-method	<p>Denotes the HTTP method to use while executing the service. The HTTP connector supports the following methods:</p> <ul data-bbox="409 846 654 1015" style="list-style-type: none"> <li>■ GET</li> <li>■ POST (Default)</li> <li>■ PUT</li> <li>■ DELETE</li> </ul> <pre data-bbox="409 1064 1263 1320">&lt;implementation type="HTTP" xmlns="http://httpconnector.opentext.com/1.0/implementation" xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/"&gt;     &lt;connection-id&gt;TEST-CONNECTION&lt;/connection-id&gt;     &lt;uri&gt;/v2/rss/testrest&lt;/uri&gt;     &lt;http-method&gt;GET POST PUT DELETE&lt;/http-method&gt;     ..... &lt;/implementation&gt;</pre>
ignore-cookies	<p>Set <code>ignore-cookies</code> to <code>true</code> to prevent the HttpClient from sending or receiving cookies. The default value is <code>false</code>.</p> <pre data-bbox="409 1474 1263 1769">&lt;implementation type="HTTP" xmlns="http://httpconnector.opentext.com/1.0/implementation" xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/"&gt;     &lt;connection-id&gt;TEST-CONNECTION&lt;/connection-id&gt;     &lt;uri&gt;/v2/rss/testrest&lt;/uri&gt;     &lt;http-method&gt;GET POST PUT DELETE&lt;/http-method&gt;     &lt;ignore-cookies&gt;true&lt;/ignore-cookies&gt;     ..... &lt;/implementation&gt;</pre>
clean-	<p>If set to <code>true</code>, the response SOAP body content is cleared. This is useful only</p>

Tag	Description
response-body	<p>for methods that cannot return the response method element. The default value is false.</p> <pre data-bbox="414 397 1258 661">&lt;implementation type="HTTP"   xmlns="http://httpconnector.opentext.com/1.0/implementation"   xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/"&gt;   &lt;connection-id&gt;TEST-CONNECTION&lt;/connection-id&gt;   &lt;uri&gt;/v2/rss/testrest&lt;/uri&gt;   &lt;http-method&gt;GET   POST   PUT   DELETE&lt;/http-method&gt;   &lt;clean-response-body&gt;true&lt;/clean-response-body&gt;   .... &lt;/implementation&gt;</pre>
request-handler	<p>Provides various details on request processing. This includes the handlers that process the request and the transformation that must be applied on the request XML.</p> <p>The following attributes are required for the request:</p> <ul style="list-style-type: none"> <li>■ class: Fully qualified class name of the handler that can process the request. The HTTP connector, by default, can access Web services and REST-based services. Each of these require separate handlers. <ul style="list-style-type: none"> <li>• If the service is Web service, use com.opentext.applicationconnector.httpconnector.impl.StandardRequestHandler.</li> <li>• If the service is REST-based, use com.opentext.applicationconnector.httpconnector.impl.RestRequestHandler</li> </ul> </li> </ul> <p>Following are the child elements:</p> <ul style="list-style-type: none"> <li>■ uri-parameters</li> <li>■ xslt</li> <li>■ req-headers</li> <li>■ root-xpath</li> </ul> <p><b>Web service implementation for Non-REST API</b></p> <pre data-bbox="414 1548 1396 1759">&lt;implementation type="HTTP"   xmlns="http://httpconnector.opentext.com/1.0/implementation"   xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/"&gt;   &lt;connection-id&gt;TEST-CONNECTION&lt;/connection-id&gt;   &lt;uri&gt;/v2/rss/testnonrest&lt;/uri&gt;   &lt;request-handler     class="com.opentext.applicationconnector.httpconnector.impl.StandardRe"</pre>

Tag	Description
	<pre> questHandler"&gt;     .... &lt;/request-handler&gt; .... &lt;/implementation&gt; </pre> <p><b>Web service implementation for REST API</b></p> <pre> &lt;implementation type="HTTP" xmlns="http://httpconnector.opentext.com/1.0/implementation" xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/"&gt;     &lt;connection-id&gt;TEST-CONNECTION&lt;/connection-id&gt;     &lt;uri&gt;/v2/rss/testrest&lt;/uri&gt;     &lt;request-handler class="com.opentext.applicationconnector.httpconnector.impl.RestRequestHandler"&gt;         ....     &lt;/request-handler&gt;     .... &lt;/implementation&gt; </pre>
uri-parameters	<p>Details about the list of parameters required by the service. Each parameter is represented by the parameter element. The child element is: parameter.</p> <pre> &lt;implementation type="HTTP" xmlns="http://httpconnector.opentext.com/1.0/implementation" xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/"&gt;     &lt;connection-id&gt;TEST-CONNECTION&lt;/connection-id&gt;     &lt;uri&gt;/v2/rss/testrest&lt;/uri&gt;     &lt;request-handler class="com.opentext.applicationconnector.httpconnector.impl.RestRequestHandler"&gt;         &lt;uri-parameters&gt;             &lt;parameter type="xpath"&gt;./Username&lt;/parameter&gt;             &lt;parameter type="Fixed"&gt;./Tags/Tag&lt;/parameter&gt;         &lt;/uri-parameters&gt;     &lt;/request-handler&gt;     .... &lt;/implementation&gt; </pre>
parameter	<p>Represents one parameter. It has the following attribute.</p> <ul style="list-style-type: none"> <li>■ type: Denotes the type of value entered in the parameter, which can be an XPath, fixed connection-parameter, or connection-uri.</li> </ul> <p><b>Web service implementation illustrating connection-uri and connection-parameter</b></p>

Tag	Description
	<pre> &lt;implementation   xmlns="http://httpconnector.opentext.com/1.0/implementation"   xmlns:c="http://schemas.cordys.com/cws/1.0"   xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/" type="HTTP"&gt;   &lt;connection-id&gt;JENKINS&lt;/connection-id&gt;   &lt;http-method&gt;GET&lt;/http-method&gt;   &lt;uri&gt;/jenkins/job/httpconnector-test-wip-linux/{0}/{1}&lt;/uri&gt;   &lt;request-handler     class="com.opentext.applicationconnector.httpconnector.impl.RestRequestHandler"&gt;     &lt;uri-parameters&gt;       &lt;parameter type="connection-parameter"&gt;firstBuild&lt;/parameter&gt;       &lt;parameter type="connection-uri" /&gt;     &lt;/uri-parameters&gt;   &lt;/request-handler&gt;   &lt;response-handler     class="com.opentext.applicationconnector.httpconnector.impl.RestResponseHandler" /&gt;   &lt;valid-response-code&gt;200&lt;/valid-response-code&gt; &lt;/implementation&gt; </pre>

### XML store connection configuration

```

<configurations
  xmlns="http://httpconnector.opentext.com/1.0/configuration">
  <connections>
    <connection id="JENKINS">
      <url>http://buildmaster-hyd.vanenburg.com/api/json</url>
      <parameters>
        <parameter type="STRING">
          <name>firstBuild</name>
          <value>102</value>
        </parameter>
      </parameters>
    </connection>
  </connections>
</configurations>

```

In this example, the connection configuration from the XML store has a STRING typed parameter with the name `firstBuild` having the parameter value as 102. To read this value and set it as one of the parameter values for the Web service implementation, the following construct is effective.

```
<parameter type="connection-parameter">firstBuild</parameter>
```

Similarly, you can use the following construct to read the relative path of the

Tag	Description
	<p>connection URL for the provided <code>connection-id</code> from the connection configuration stored in the XML store.</p> <pre data-bbox="414 397 910 424">&lt;parameter type="connection-uri" /&gt;</pre> <p>Eventually, for the above example, the effective URI is:  <code>http://buildmaster-hyd.vanenburg.com/jenkins/job/httpconnector-test-wip-linux/102/api/json</code></p> <p>102 is the <code>connection-parameter</code> value while <code>/api/json</code> is the <code>connection-uri</code> parameter.</p>
req-headers	<p>Custom HTTP headers that are set in the request before an outbound call is made. It has the following child element: header.</p> <pre data-bbox="414 804 1393 1353">&lt;implementation type="HTTP"   xmlns="http://httpconnector.opentext.com/1.0/implementation"   xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/"&gt;   &lt;connection-id&gt;TEST-CONNECTION&lt;/connection-id&gt;   &lt;uri&gt;/v2/rss/testrest&lt;/uri&gt;   &lt;request-handler     class="com.opentext.applicationconnector.httpconnector.impl.RestRequestHandler"&gt;     &lt;uri-parameters&gt;       &lt;parameter type="xpath"&gt;./Username&lt;/parameter&gt;       &lt;parameter type="Fixed"&gt;./Tags/Tag&lt;/parameter&gt;     &lt;/uri-parameters&gt;     &lt;req-headers&gt;       &lt;header name="x-ms-version"&gt;2012-03-01&lt;/header&gt;       &lt;header name="Content-Type"&gt;application/xml&lt;/header&gt;     &lt;/req-headers&gt;   &lt;/request-handler&gt;   .... &lt;/implementation&gt;</pre>
root-xpath	<p>The HTTP methods PUT and POST accept payload and denote the XPath to apply to extract the payload from a request and send as part of PUT or POST.</p> <pre data-bbox="414 1516 1393 1748">&lt;implementation   xmlns="http://httpconnector.opentext.com/1.0/implementation"&gt;   &lt;request-handler     class="com.opentext.applicationconnector.httpconnector.impl.RestRequestHandler"&gt;     &lt;root-xpath&gt;payload/GetTasksForUser&lt;/root-xpath&gt;   &lt;/request-handler&gt; &lt;/implementation&gt;</pre>
response-	<p>Provides various details on handling from the service that is accessed. This</p>

<b>Tag</b>	<b>Description</b>
handler	<p>includes the handlers that can process the response and the transformation that must be applied on the response.</p> <p>It has the following attribute.</p> <ul style="list-style-type: none"> <li>■ class: Fully qualified class name of the handler that can process the response. com.opentext.applicationconnector.httpconnector.impl.StandardResponseHandler com.opentext.applicationconnector.httpconnector.impl.RestResponseHandler</li> </ul> <p>It has the following child parameter: xslt.</p> <pre data-bbox="409 699 1396 1300">&lt;implementation type="HTTP"   xmlns="http://httpconnector.opentext.com/1.0/implementation"   xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/"&gt;   &lt;connection-id&gt;TEST-CONNECTION&lt;/connection-id&gt;   &lt;uri&gt;/v2/rss/testrest&lt;/uri&gt;   &lt;request-handler     class="com.opentext.applicationconnector.httpconnector.impl.RestRequestHandler"&gt;     &lt;uri-parameters&gt;       &lt;parameter type="xpath"&gt;./Username&lt;/parameter&gt;       &lt;parameter type="xpath"&gt;./Tags/Tag&lt;/parameter&gt;     &lt;/uri-parameters&gt;   &lt;/request-handler&gt;   &lt;response-handler     class="com.opentext.applicationconnector.httpconnector.impl.StandardResponseHandler"&gt;     &lt;xslt xmlstore="xslts/response.xslt"/&gt;   &lt;/response-handler&gt;   .... &lt;/implementation&gt;</pre>
valid-response-xpath	XPath expression for validating the response.
valid-response-code	HTTP response code for validating the response.
namespace s	Holds the namespace prefix mappings to use. It has the following child element: binding.
binding	Represents one namespace binding with a prefix and URI. It has the following attributes: uri and prefix.
xslt	Path to XSLT in the XML store to transform the request and response. This can be mentioned both for the request handling and the response handling.

Tag	Description
	<pre> &lt;implementation type="HTTP"   xmlns="http://httpconnector.opentext.com/1.0/implementation"   xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/"&gt;   &lt;connection-id&gt;TEST-CONNECTION&lt;/connection-id&gt;   &lt;uri&gt;/v2/rss/testrest&lt;/uri&gt;   &lt;request-handler     class="com.opentext.applicationconnector.httpconnector.impl.RestRequestHandler"&gt;     &lt;xslt xmlstore="xslts/request.xslt"/&gt;    &lt;/request-handler&gt;   &lt;response-handler     class="com.opentext.applicationconnector.httpconnector.impl.StandardResponseHandler"&gt;     &lt;xslt xmlstore="xslts/response.xslt"/&gt;   &lt;/response-handler&gt;   .... &lt;/implementation&gt; </pre>

For an example on how to develop Web services using the HTTP connector, see *Developing Web services using the HTTP connector* in the *AppWorks Platform Advanced Development Guide*.

## Identity

Identity is the relation of a digital user with a real-life person. This digital identity is a combination of identification and credentials. To determine the identity of the user, authentication takes place. During the process of authentication, the user must supply the correct credentials to identify them. This guarantees that users are who they say they are. From that moment, a computer system knows the user's identity.

## Inbox control

The Inbox control displays the AppWorks Platform Inbox in a user interface. To create associations between this control and any required model, you can use the message map. See *Using Message Map* in the *AppWorks Platform API Reference Guide*. The mapping defined between the control and the model is stored in a mapping object. See *Mapping Object* in the *AppWorks Platform API Reference Guide*.

## Schema

AppWorks Platform XForms provides a schema for the Inbox control using which the task data in the Inbox can be mapped with a data source. Using the schema, you can easily access and bind data to the control. See a schema sample.

```

<InboxControl>
    <Control type="target/all">
        <!-- Provide Target types if the control type selected is 'target' -->
        <Target type="user/role/worklist/team"/>
        <!--Need not provide any elements if the control type is 'all'-->
    </Control>
    <Criteria>
        <Query xmlns="http://schemas.cordys.com/cql/1.0">
            <And>
                <LE field="StartDate">
                    <Value>2016-06-04T11:09:32.32</Value>
                </LE>
                <GE field="StartDate">
                    <Value>2015-06-04T11:09:32.32</Value>
                </GE>
            </And>
        </Query>
    </Criteria>
    <TaskIdentifiers>
        <TaskIdentifierType id="" name="" />
    </TaskIdentifiers>
    <OrderBy/>
    <ShowNonWorkableItems/>
</InboxControl>

```

## Schema elements

The following are the schema elements.

Element	Description	Parameters
Control	Type of the control from which to retrieve data	<ul style="list-style-type: none"> <li>■ target: retrieves tasks from a specific target such as a Manager role, a Sales team, a Sales worklist, or a user JDoe. The accepted input values are:           <ul style="list-style-type: none"> <li>• role: DN of the role</li> <li>• team: ID of the organizational unit</li> <li>• worklist: ID of the worklist</li> <li>• user: DN of the user</li> </ul> </li> <li>■ all: retrieves all the tasks related to the current user</li> </ul>
Query	CQL format query to search for the task	And, Or, EQ, NQ, LT, LE, GT, GE, In, Like, Between, IsNull, Not (can be nested and repeated)

Element	Description	Parameters
		<p>The accepted input values are:</p> <ul style="list-style-type: none"> <li>■ TaskId</li> <li>■ ParentTaskId</li> <li>■ SourceInstanceId</li> <li>■ State</li> <li>■ ProcessName</li> <li>■ Activity</li> <li>■ Priority</li> <li>■ Sender</li> <li>■ SourceType</li> <li>■ Assignee</li> <li>■ CompletedByUser</li> <li>■ DelegatedToUser</li> <li>■ DeliveryDate</li> <li>■ StartDate</li> <li>■ DueDate</li> <li>■ StartedOn</li> <li>■ CompletionDate</li> <li>■ IsPriorityFixed</li> <li>■ UITaskId</li> </ul>
TaskIdentifiers	Set of task identifier types to search for the task	<p>The accepted input values are:</p> <ul style="list-style-type: none"> <li>■ id: GUID of the task identifier to query</li> <li>■ name: must be TaskIdentifier</li> </ul>
OrderBy	Parameter by which to sort the search results	<p>The accepted input values are system attributes such as:</p> <ul style="list-style-type: none"> <li>■ Activity</li> <li>■ Assignee</li> <li>■ Sender</li> <li>■ StartDate</li> <li>■ StartedOn</li> <li>■ DueOn</li> <li>■ Priority</li> <li>■ State</li> </ul>

Element	Description	Parameters
		<ul style="list-style-type: none"> <li>■ DeliveryDate</li> <li>■ TaskId</li> </ul>
ShowNonWorkableItems	Whether the non-workable tasks for the user must be displayed	<p>The accepted input value is Boolean.</p> <ul style="list-style-type: none"> <li>■ true: lists the tasks that are not workable by the user along with the other tasks</li> <li>■ false: does not list tasks that are not workable by the current user</li> </ul>

To display details of a location in a window, you must extend the schema by adding elements to the `BusinessObject` element of the schema.

#### To add an element to the schema:

- Open the message map of the model used in the control, right-click the **Business Object** element in the Target pane, and then select **Add element**.
- A new node is added to the schema.  
You can modify the node to define the new element.

## Properties

The properties associated with the Inbox control define its behavior. You can set the ID of the control either in its property sheet or programmatically using APIs.

The following are the list of properties.

Design time	Runtime	Description
ID	id	String that identifies the control on an XForm. If not specified, a unique ID is automatically generated.
	Target type	Target Type can be role, team, worklist, or user.
	Target	<p>Target can be any of the following:</p> <ul style="list-style-type: none"> <li>■ role: DN of the role</li> <li>■ team: ID of the organizational unit</li> <li>■ worklist: ID of the worklist</li> <li>■ user: DN of the user</li> </ul>

## Methods

You can call the following methods to programmatically set the various properties that are available on the Inbox composite control.

Design time	Description
showTasksByTarget (sTargetType, sTargetValue)	<p>Retrieves all the tasks that are sent to the Target of type <code>sTargetType</code> and identified by <code>sTargetValue</code>, and displays them in the Inbox composite control. The accepted values are:</p> <ul style="list-style-type: none"> <li>▪ <code>sTargetType</code>: user/role/team/worklist</li> <li>▪ <code>sTargetValue</code>: <ul style="list-style-type: none"> <li>• role: DN of the role</li> <li>• team: ID of the organizational unit</li> <li>• worklist: ID of the worklist</li> <li>• user: DN of the user</li> </ul> </li> </ul>
showAllMyTasks()	Retrieves all the tasks related to the current user and displays them in the Inbox composite control.
getFilterXml()	Returns an instance of the Inbox composite control schema for the current settings.
setFilterXml (sControlXML)	<p>Retrieves all the tasks that satisfy the query settings provided by <code>sControlXML</code> and displays them in the Inbox composite control. This method takes <code>sControlXML</code>, an XML instance of the Inbox composite control schema defined above. See the following sample Inbox XML.</p> <pre data-bbox="567 1094 1367 1567">&lt;InboxControl   xmlns="http://schemas.cordys.com/notification/workflow/1.0"&gt;   &lt;Control type="target"&gt;     &lt;Target       type="user/role/team/worklist"&gt;DN/ID&lt;/Target&gt;   &lt;/Control&gt;   &lt;Criteria&gt;     &lt;Query&gt;&lt;/Query&gt;   &lt;/Criteria&gt;   &lt;TaskIdentifiers&gt;     &lt;TaskIdentifierType/&gt;   &lt;/TaskIdentifiers&gt;   &lt;OrderBy/&gt;   &lt;ShowNonWorkableItems/&gt; &lt;/InboxControl&gt;</pre>

## Installation role in licensing

The first step in the AppWorks Platform Licensing process is to register the customer with AppWorks Platform. To use AppWorks Platform, a customer must register with AppWorks Platform through an ISV. During registration, the following details are required:

- 
- Registered Customer name
  - Registered Site name
  - Password

After entering the required details, the customer is registered with AppWorks Platform. The customer uses the entered credentials when installing AppWorks Platform.

### **Installation and post installation**

During the AppWorks Platform installation, the installer requires the following information from the customer:

Field	Description
Registered Customer name	Unique name of the customer registered with AppWorks Platform.
Registered Site name	Unique name of the site of the specific customer for which AppWorks Platform is being installed.
Password	Password for the registered site as entered by the customer during registration. If this password is entered, the installer checks its validity, and if valid, obtains the license key from the AppWorks Platform License Service and updates LDAP with the same. If the customer has not registered with AppWorks Platform, and therefore does not possess a password, this field can be left blank during installation. The installer proceeds with the installation, leaving the license key in LDAP blank. In that case, the customer must contact the AppWorks Platform License Service and obtain the license key. After the key is obtained, it must be updated in LDAP using the License Configuration feature of the Management Console. See <a href="#">Modifying AppWorks Platform license information</a> .

You can change the configuration of a computer (Standalone or Contributor) anytime. This configuration is usually defined during the AppWorks Platform installation. However, using the Management Console, the Standalone computer can be made a Contributor or vice versa anytime. This tool has a License Configuration option that enables users to change the license configuration.

**Note:** OpenText recommends not to change the configuration from a Contributor to a Master or vice versa.

When a computer is made a Contributor computer, it is essential to go to the Management Console and specify the Master computer for the same. This ensures that the Contributor computer is registered with the Master, and this facilitates automatic generation and sending of the usage report.

---

Customers cannot use AppWorks Platform unless a valid license key value is entered and stored in LDAP.

When the AppWorks Platform License Site is contacted to obtain the license key, an email is sent to the Site Administrator with the details of the installation (the Registered Customer Name, Registered Site Name, and installation type) and the license key.

For information on types of installations, see [Installation types](#).

## Installation types

AppWorks Platform licensing recognizes two types of installations.

Installation type	Description
Single Install	In this installation, there is only one physical installation of AppWorks Platform in one site (also known as Standalone installation). This installation has a Monitor Service Group, which sends the usage report to the licensing service, and this includes the LicenseManagement Web service interface from the Cordys ESBServer ISV package. The administrator of the standalone computer generates the usage report for the installation. The usage report of this installation is directly sent to the AppWorks Platform Licensing Server.
Multiple Install (deprecated)	In this installation, the site can have more than one AppWorks Platform installation. One of the installations is considered a Master, while the others are Contributors. The Master and Contributors are determined during the installation process. Additionally, users can change the configuration manually anytime. The Contributors send their usage reports (also known as Contributor reports) to the Master. The Master must send its usage report (Contributor report) to itself. The Master consolidates the usage reports and sends the consolidated usage report (called the Master report) to the AppWorks Platform License Service. The AppWorks Platform License Service receives the Master usage report and accordingly sends back the license keys for all the AppWorks Platform installations to the Master. The Master receives the license keys and distributes them to the individual AppWorks Platform installations for use. Alternatively, the Contributors can retrieve their license keys from the Master. The Master and the Contributors have a Monitor Service Group that includes the LicenseManagement Web service interface from the Cordys ESBServer application. The Master additionally includes the LicenseMaster Web service interface of the Cordys ESBServer Application to the same Monitor Service Group. This feature is deprecated.

## Instance-level views

Use an instance-level view to monitor an aspect of a selected process instance.

<b>View</b>	<b>Description</b>	<b>Corresponding Composite Control</b>
GetTopXInstancesByLeadTimeForStatus	Displays the top X instances that have the highest lead time for the selected status. This view primarily identifies bottlenecks. It displays statuses such as Waiting and Aborted on instances that have contributed the most to increased process cycle times.	TopXInstancesByLeadTimeForStatus_SpecificProcess

The time values that display are based on the time zone of the client browser in a 12-hour format.

## Instance-specific properties

When a process is instantiated, a property is instantiated in the message map to provide instance-specific properties. These properties are collectively known as instance properties. They retrieve information about the process instances at runtime and can be used in process definitions.

### XML structure

See the XML structure of the instance-specific properties.

```

<instanceProperties debugMode="true">
    <processName>GetProducts_vcmdemo10.bpm</processName>
    <processDescription>GetProduct</processDescription>
    <bpmn>/Business Processes/BPMN//Business Processes/GetProduct_
vcmdemo10.bpm/11695079978857</bpmn>
    <identifier>{5D8D56E3-523A-47E1-9118-F24520F58201}</identifier>
    <modelSpace>[Organization | ISV]</modelSpace>
    <monitor>true</monitor>
    <startedBy>cn=jdoe,cn=organizational
users,o=system,cn=cordys,o=vanenburg.com</startedBy>
        <currentOwner>cn=jdoe,cn=organizational
users,o=system,cn=cordys,o=vanenburg.com</currentOwner>
            <organization>o=system,cn=cordys,o=vanenburg.com</organization>

    <rootEntityInstanceId>F8B156E3615F11E5FBF273BADAC756DF.16385</rootEntityInstanceId>
        <startTime>1267526084395</startTime>
        <processInstantiationType>Process Debugger</processInstantiationType>
        <priority>3</priority>
        <parentType>PROCESS</parentType>
        <instantiationLocale>en_US</instantiationLocale>
    </instanceProperties>

```

## Elements in the XML Structure

The XML structure of the instance-specific properties contains the following elements.

Element	Description
processName	Name of the process.
processDescription	Description of the process.
bpmn	Information regarding the rendering of the process during runtime. This information is used by Process Instance Manager to display the process instance details in a graphical format.
identifier	Unique identifier of the process instance.
modelSpace	Place from which the process instances are executed and displayed in Process Instance Manager.
monitor	Monitoring attribute set for the process.
startedBy	LDAP DN of the user or role that triggered the process instance.
currentOwner	LDAP DN of the user or role that last triggered the process instance.  <b>Note:</b> Initially, the DN is the same as the value in the startedBy element. However, if the process instance is triggered again by an intermediate user or role, that user or role is the owner for further operations in the process. For example, consider that a process instance sends a message to a participant and is waiting for a response. The participant in turn sends back a response that triggers the same process instance. The owner for all future operations with the process instance is the participant whose response triggered the process instance.
organization	LDAP DN of the user's organization that triggered the process instance.
rootEntityInstanceId	Associated entity instance ID. This is filled in the below scenarios: <ul style="list-style-type: none"> <li>■ When a process is triggered from an entity lifecycle</li> <li>■ When a process is triggered from an entity rule</li> </ul>
startTime	Time when the process instance was instantiated. The format of the startTime is usually in 13 digits, where the last three digits are milliseconds. Consider that the startTime is 1267526084395, where 395 is milliseconds and doesn't need to be converted to a valid YYYY-MM-DDTHH:mm:ss.ms or UTC format.  To convert the format of the startTime of a process instance in a valid YYYY-MM-DDTHH:mm:ss.ms format, do the following, for example:

Element	Description
	<p>3. From Target Process Specific Messages, drag the newly created element on to the Assignments column of the message map.</p> <p>4. Select <b>Replace Content with Expression</b> and use the following expression.</p> <pre data-bbox="621 508 1421 840">concat (sprintf( "%D(%yyyy-%MM-%dd)T%T(%HH:%mm:%ss)", substring( instanceProperties/startTime/text(), 1, 10), substring( instanceProperties/startTime/text(), 1, 10), ".", substring( instanceProperties/startTime/text(), 11, 13) )</pre> <p>The startTime is converted to YYYY-MM-DDTHH:mm:ss.ms format or a valid UTC format.</p> <pre data-bbox="621 931 1421 1121">&lt;TestStartTime xmlns="http://schemas.cordys.com/default"&gt; &lt;Processstarttime&gt;2010-03- 02T16:04:44.395&lt;/Processstarttime&gt; &lt;/TestStartTime&gt;</pre>
processInstantiationType	Source from where the process is triggered.
priority	Execution priority of the process, where priority ranges from 1 to 5. 1 corresponds to very low and 5 corresponds to very high.
parentType	<p>Parent type of the subprocess.</p> <ul style="list-style-type: none"> <li>■ If we invoke a subprocess from a main process, the parentType is PROCESS.</li> <li>■ If we invoke a subprocess from a case, the parentType is CASE.</li> </ul>
callerInstance	<ul style="list-style-type: none"> <li>■ Subprocess instance ID if the parentType is PROCESS</li> <li>■ Case instance ID if the parentType is CASE</li> </ul>
instantiationLocale	Locale information of the instantiated process, for example, en_US or ja_JP.

## Instances by Process Definition interface

The Instances by Process Definition interface is part of the Process Instance Manager.

**Note:** This interface displays the business identifiers associated with the business process model. When the business identifiers are modified, the interface displays only those business identifiers and values associated with the latest published or deployed business process model. To view the modified business identifier values of previous instances, right-click the relevant instance, and then select **Show Business Identifiers**.

The Instances by Process Definition interface contains the following fields.

Field	Description
Process Name	Name of the process model for which you are viewing process instances.
Description	Description of the process model for which you are viewing process instances.
Start Time	Date and time at which the process instance of the business process model was initiated.
Status	Status of the process instance such as Debug, Completed, Terminated, Waiting, and Aborted.
Type	Details from where the process instance was executed, for example, Process Debugger or Run from Process Designer.
Show Subprocesses	Option to view subprocess instance details of a process instance. Click  .
Show Activities	Option to view activity instance details of a process instance. Click  .
Show Original Process	Option to view the initial process instance that was run. When running a process, if an activity fails, you can rectify the error and run the process again. A new instance starts. Click  .
Show Substitute Process	Option to view activity instance details of the substitute process. Click  .
Show Parent Process	Option to show the parent process of the corresponding subprocess, which is enabled only for subprocesses. Click  .
<b>Note:</b> This option is not available for process instances that were migrated from earlier versions of AppWorks Platform but only for process instances that were instantiated in Cordys BOP 4.1.	

To perform an operation in the Instances by Process Definition interface, right-click a process instance, and then select an operation. Alternatively, select a process instance, and then click the necessary icon on the toolbar. You can perform the following operations.

Operation	Description
Suspend	Suspends a process instance.

<b>Operation</b>	<b>Description</b>
	<b>Note:</b> You can suspend process instances that are in the Running or Waiting status only.
Resume 	Resumes the process instance execution. <b>Note:</b> You can resume process instances that are in the Suspended or Debug Ready status only.
Restart 	Restarts a process instance. <b>Note:</b> You can restart process instances that are in the Aborted status only.
Terminate 	Terminates a process instance that is not in the Complete status.
Skip Activity 	Skips running an aborted activity. This ensures that the current activity is skipped and the immediate next activity is run.
Show Graphical View 	Option to view the sequence of activities in a process instance graphically.
Show Process Logs 	Option to view process logs that display the cause for errors for a specific process instance. This saves time without having to view the log message from the file system.
Show Start Message 	Option to view the start message for the process instance.
Show End Message 	Option to view the end message for the process instance.
Show/Edit Message Map 	Option to view or edit the message map for the process instance.
Show Business Identifier Values 	Option to view values of the business identifiers used in the process instance.

<b>Operation</b>	<b>Description</b>
Show Message Queue	Option to view queues for the incoming messages for the selected process instance. In the Queues for Incoming Messages - View the Queues for the process, the List View tab displays individual entries of Receive Message for the process instance whereas the All View tab displays all the entries of Receive Message that are in queue for the selected process instance.
Show Error Text	Option to view the error text for the process instance. 
Debug	Option to run the process instance in the debugger view. 

## Intranet user authentication

Intranet users can be within the domain or outside the firewall, that is, intranet users who are traveling. In the intranet user scenario, the basic authentication and NT challenge response is turned on. For users within the primary domain, the authentication dialog box is not displayed and they can access AppWorks Platform through the NT challenge response authentication. However, in case of traveling intranet users accessing the site from outside the firewall, basic authentication is used, just as in the case of an extranet user.

**Note:** For an intranet user to access the AppWorks Platform Primary edition installed on a computer in the demilitarized zone (DMZ), basic authentication with the domain field is required.

## Java Archive Definition interface

The Java archive definition interface contains two tabs. To enter general information and to store any dependency on other sources.

<b>Tab</b>	<b>Field name</b>	<b>Description</b>	<b>User action</b>
General	Name	Name of the Java Archive Definition file.	Provide a name.
	Description	Description of the Java Archive Definition.	Provide a brief description.
	JAR File Name	Name of the	Provide a name for the JAR file.

<b>Tab</b>	<b>Field name</b>	<b>Description</b>	<b>User action</b>
		JAR file that is created. Contains the build output of all the Java sources (.class files).	
	JAR-Content Folder	Folder from which Java source files are included.	<p>Browse and select the project or folder that contains the Java sources.</p> <p><b>Note:</b> The JAR contains the build output of the Java sources, that is, the .class files. As the documents are published according to the publish logic of Java Archive Definition, the JAR also includes all the folders and documents (including other AppWorks Platform documents) that exist in this folder. By default, the .JAR file is stored at &lt;AppWorks_Platform_install dir&gt;/&lt;instance name&gt;/&lt;qualified name of the root folder&gt; at runtime. If the .JAR file was created on the WS-AppServer content, it is stored at &lt;AppWorks_Platform_installdir&gt;\&lt;instance name&gt;\bsf\runtime\&lt;organization&gt;.</p> <p><b>Note:</b> To exclude files that must be outside the scope of JAR, set the Qualified Name (QN) on a folder. See Using the qualified name in the <i>AppWorks Platform Advanced Development Guide</i>.</p>

<b>Tab</b>	<b>Field name</b>	<b>Description</b>	<b>User action</b>
<p><b>Dependencies:</b> Optional. To include JAR files or Java Archive Sources which share dependencies with the Java Archive Source that you are going to create:</p> <ol style="list-style-type: none"><li>1. Click  (Add) to select a relevant type of dependency. The Java Archive Dependency Wizard opens.</li><li>2. Select one of the options provided.</li></ol>			

<b>Tab</b>	<b>Field name</b>	<b>Description</b>	<b>User action</b>
	Java Archive Source	Adds an existing Java Archive Source available in the workspace as a dependent file.	Select the option and click <b>Next</b> to browse the solution. Then select the relevant Java Archive Source, and click <b>Finish</b> to fill the dependency. This creates a package dependency on the providing application package.
	Internal Archive Source	Adds an existing Java Archive (JAR) available in the workspace as a dependent file.	Select the option and click <b>Next</b> to browse the solution. Then select the relevant Java Archive, and click <b>Finish</b> to fill the dependency. This creates a package dependency on the providing application package.
	External Java Archive	Adds an external JAR to the project and creates the dependency.	<p>Select the option and click <b>Next</b>. On the next page, enter the dependency details. You can enter:</p> <ul style="list-style-type: none"> <li>■ Only the relative or absolute file system path of the Java archive in <b>Enter the path to the external JAR</b> and leave the name of the providing application package in <b>Enter the application package name</b> empty. This does not create any new package dependencies and can be selected when a Java Archive is used in applications that are deployed on the development or production environment outside AppWorks Platform.</li> <li>■ Both the relative or absolute file system path of the Java archive in <b>Enter the path to the external JAR</b> and the name of the providing application package in <b>Enter the application package name</b>. This creates a package dependency on the providing application package and can be selected when a Java Archive is used from a third-party AppWorks Platform application.</li> </ul>

<b>Tab</b>	<b>Field name</b>	<b>Description</b>	<b>User action</b>
			<p><b>Note:</b> The name of the providing application package may be different from the file name of the providing package. See Setting Properties of an Application Package in the <i>AppWorks Platform Advanced Development Guide</i> for more information.</p>

## Key usage options

Key usage is an attribute for a digital certificate that defines the purpose of the certificate. You can verify if the purpose of the certificate is any of the following while validating the certificate:

<b>Key usage</b>	<b>Description</b>
Digital Signature	Used for identifying an entity or person for authentication.
Non Repudiation	Used when a certificate is used with a protocol that encrypts keys.
Key Encipherment	Used when a certificate is used with a protocol that encrypts keys.
Data Encipherment	Used when the public key is used for encrypting user data, other than cryptographic keys.
Key Agreement	Used when the subject public key is used to verify a signature on certificates.
Certificate Signing	Used when the subject public key is used to verify a signature on certificates.
CRL Signing	Used when the subject public key is to verify a signature on revocation information.
Encipher Only	Used only when key agreement is also enabled, ensuring that the public key is used only for enciphering data while performing key agreement.
Decipher Only	Used when key agreement is enabled, ensuring that the public key is used only for deciphering data while performing key agreement.

**Note:** In addition to the key usage options, a digital certificate may have another field called the **extended key usage**. For information on Extended Key Usage, see [Extended key usage options](#).

## Keyboard Shortcuts in XPath Editor

The following table describes the keyboard shortcuts that you can use in the XPath Editor:

Shortcut key	Similar action	Action
+	Expand from context menu.	Expands the selected node.
Alt + U	Use button under Function tree.	Uses the selected function or operator.
Ctrl + /	Use from context menu.	Uses the latest selected node.
Ctrl + Enter	Click <b>OK</b> .	Closes the XPath Editor and returns the XPath to the calling application.
Ctrl + Shift + T	Click <b>Test</b> .	Tests the XPath.
Ctrl + Shift + V	Click <b>Validate</b> .	Validates the XPath.
Ctrl + U	Use button under Source tree.	Uses the selected XML node.
Shift + Esc	Click <b>Cancel</b> .	Closes the XPath Editor without returning the XPath.
	<b>Expand All</b> from context menu.	Expands all children of the selected node.
	-	Collapses all children of the selected node.

## KPI composite control properties interface

Use the Key Performance Indicator (KPI) composite control properties interface to configure the properties of the composite control.

Following are the properties interface fields on the manual KPI definition composite control:

Field name	Description
Label	Description of the value, for example, Total_OrderAmount.
Range	Default ranges defined in the KPI. You cannot edit the ranges. You can provide colors to the ranges from the color picker.

<b>Field name</b>	<b>Description</b>
Render View on load	Render the view while the page loads.
Show Legend	Default option to view the legend information of the ranges and unit of measure. To view only the gauge, clear this option.
Target Value	Editable target value that you specify for the KPI is displayed by default.
Title	Chart title that is not applicable for Angular Gauge and Linear Gauge. You can also use the XForms label to specify the title.
Unit of Measure	Editable unit of measure you selected while defining the KPI is displayed by default.
View	One of the following chart types: Angular Gauge, Linear Gauge, Vertical Bullet, or Horizontal Bullet.

Fields on the properties interface of the KPI composite control with attached business measure:

<b>Field name</b>	<b>Description</b>
Time-frame	By default, the defined time-frame is displayed in the Business Measure. It is an editable field. <b>Note:</b> For time-frame details, see Selecting a Web Service and a Time-Frame for a Business Measure in the <i>AppWorks Platform Advanced Development Guide</i> .
Label	Description of the value. For example, Total_OrderAmount.
Render View on load check box	Render the view while the page loads.
Show Legend	View the legend information about ranges and the unit of measure. Clear this check box to view only the gauge. Legends are selected by default.
Title	Chart title. The title property is not applicable for Angular Gauge and Linear Gauge. Use the XForms label to specify the same.
View	Chart types: Angular Gauge, Linear Gauge, Vertical Bullet, or Horizontal Bullet.
Range	Ranges defined in the KPI are displayed by default. You cannot edit the ranges. You can provide colors to the ranges from the color picker.
Target Value	Editable target value that you specify for the KPI is displayed by default.
Unit of	Editable unit of measure that you select while defining the KPI is displayed by

<b>Field name</b>	<b>Description</b>
Measure	default.
Input Parameter	<p>Value of the attribute for which you need dynamic input while designing the dashboard. This is specified in the business measure. Values defined during the KPI definition are displayed here by default. You can edit the default values depending on the parameter and its data type.</p> <p><b>Note:</b> There is one scenario in which the default values defined are not displayed. KPIs with multiple namespaces do not carry forward the default values in the Web Service Definition Language (WSDL) for the input parameters. Therefore, the default values are not displayed in the user interface, for example, when a BPM consumes a message to start running it with a namespace that is different from the BPM namespace. The Web service generated on this BPM has multiple namespaces in the associated WSDL. When this Web service is used in the BM used to build the KPI, the default values are not displayed. The composite control created on such KPI, if used in a user interface, do not display the default values that are defined for the input parameters.</p>

Fields on the properties interface of the KPI composite control with schedule (KPI Trends):

<b>Field name</b>	<b>Description</b>
Title	Chart title. The title property is not applicable for the angular gauge and linear gauge. You can use the XFORMS label to specify the title.
View	Read-only field that displays the line chart by default.
Label	Description of the value, for example, Total_OrderAmount.
Target Value	Editable target value that you specify for the KPI is displayed by default.
Unit Of Measure	Editable unit of measure that you select while defining the KPI is displayed by default.
Range	Ranges defined in the KPI are displayed by default. You cannot edit the ranges. You can provide colors to the ranges from the color picker.
X-Axis Caption	X-axis caption.
X-Axis Label Style	Select one of the following for the x-axis label style: <ul style="list-style-type: none"> <li>■ <b>Truncate:</b> Displays the truncated label.</li> <li>■ <b>Rotate:</b> Displays the rotated label vertically.</li> <li>■ <b>Slant:</b> Displays the slanted label at a 45-degree angle.</li> </ul>

<b>Field name</b>	<b>Description</b>
	<ul style="list-style-type: none"> <li>■ <b>Stagger:</b> Displays the staggered labels in multiple lines.</li> <li>■ <b>Wrap:</b> Displays the wrapped label. If there isn't enough space on the chart, the text of the labels is trimmed and an ellipsis (...) is added at the end. Tooltips are displayed for these labels.</li> </ul>
Y-Axis Caption	Y-axis caption.
Render View on load check box	Render the view while the page loads.
Time-frame	Editable field that is displayed by default in the Business Measure.
Input Parameter	Value of the attribute for which you need dynamic input while designing the dashboard as specified in business measure. Type in the input values, depending on the parameter and its data type.
Show Trend	<p>Trend Analysis refers to the analytical comparison of monitored KPIs and identifying their patterns over a period of time. Trends can be plotted over a period of time based upon the processed values of the expression. Select this check box to view the trend as a graph. Clear this option to view latest processed KPI value as a gauge view. Show Trend is selected by default if the KPI has a Schedule attached to it. If you clear the Show Trend option, only scenario 2 properties will appear. The time values displayed will be based on the time zone of the client's browser in 12-hour format.</p> <p><b>Note:</b> While configuring the drill-down option from a Time Frame Composite Control, mapping is possible only between similar time frames. For instance, mapping is possible from a Rolling Time Frame Composite Control, only if the Business Measure on which the KPI is built is having Rolling time frame; similarly, mapping is possible from a Static Time Frame Composite Control, only if the Business Measure on which the KPI is built is having Static time frame.</p>
Show Legend	Select this check box to view the legend information regarding ranges and unit of measure. Clear this option to view only the gauge. Show Legend is selected by default.

#### APIs for the KPI Composite Control

<b>API Name</b>	<b>Description</b>
setAsynchronous	Valid values are True and False. By default, the request is sent in the

<b>API Name</b>	<b>Description</b>
(String)	asynchronous mode. To change the mode to synchronous, invoke the API with value False and invoke Refresh API to redraw the chart.
refresh	Use the API to redraw the chart.
getChartObject	Returns the chart object on which the APIs of Chart are invoked.

Events for the KPI composite control:

<b>Event</b>	<b>Description</b>
ondataclick	Sent when clicked on a data point on the chart. For example:  <pre>function Form_InitDone(eventObject) { &lt;kpiid&gt;.addListener(ondataclicklistenerFn); } function ondataclicklistenerFn(eventObject) { //Can do any operation here }</pre>

## LDAP cache statistics

The following table contains the LDAP cache statistics:

<b>Field name</b>	<b>Description</b>
Total requests	Number of requests to LDAP.
Number of hits	Number of successful queries to LDAP.
Number of misses	Number of queries that did not yield a successful response from LDAP.
Number of flushes	Number of times the LDAP cache was flushed.
Number of cached results	Number of results from LDAP that are cached at the Web gateway.

**Note:** To clear the contents of the LDAP cache stored at the Web gateway, click **Clear LDAP Cache**.

## LDAP logon interface

The following table contains the details of the LDAP logon interface:

<b>Field</b>	<b>Description</b>
LDAP Server is running in SSL	Connect to the LDAP running in Secure Sockets Layer (SSL) mode.

Field	Description
Use as default on this computer	Use the default LDAP server, port, and credentials. As a result, all service containers running on this computer use the same LDAP server details.
Search Root	LDAP root.
Server	Name of the LDAP server.
Password	Password to connect to LDAP.
Port	Port to connect to.
Name	User name for authentication to LDAP.

## Leveraging XQY extensions in the WS-AppServer

The WS-AppServer uses the XQY layer for database persistence. XQY has extensions that provide specific functionalities to the WS-AppServer. This topic describes the attributes that the WS-AppServer provides to leverage these extensions. These attributes can be set both at the method implementation stage, and in the SOAP requests.

Following are the options provided by the WS-AppServer:

Option	Valid values	Default value	Operations	Databases
batchUpdate	Yes No	No	<ul style="list-style-type: none"> <li>■ Insert</li> <li>■ Update</li> <li>■ Delete</li> </ul>	All databases
preserveSpace	True False	False	<ul style="list-style-type: none"> <li>■ Insert</li> <li>■ Update</li> <li>■ Delete</li> <li>■ Query</li> </ul>	All databases
reply	Yes No	Yes	<ul style="list-style-type: none"> <li>■ Insert</li> <li>■ Update</li> <li>■ Delete</li> </ul>	All databases

### Using 'reply'

This attribute is used to read the database again for a given operation and send the response.

For an insert request with `reply='yes'`, XQY sends a select query with the given primary key, reads the database after the insert operation, and sends the response.

---

If this attribute is set to **no** (`reply='no'`), XQY does not read the database after the insert operation, thus improving performance. The WS-AppServer sends an empty response. Setting this attribute to **no** helps in cases where there are bulk inserts or updates, and you do not want to retrieve the inserted row information for every request.

The default value is **yes**.

#### To set the attribute to no:

- In a SOAP request, add an attribute **reply** with the value **no** (`reply='no'`).
- In method implementation, add an attribute **reply** with value **no** to the update method stored in LDAP.

Examples of using this attribute when WS-AppServer is in SOAP mode:

<b>Reply</b>	<b>SOAP request</b>	<b>SOAP response</b>
Yes	<pre>&lt;SOAP:Envelope   xmlns:SOAP="http://schemas.xmlsoap.org/so   ap/envelope/"&gt; &lt;SOAP:Body&gt;     &lt;UpdateCustomer reply="yes"       xmlns="http://schemas.cordys.com/Ws-       AppServer/Testing"&gt; &lt;tuple&gt; &lt;new&gt;         &lt;Customer&gt; &lt;id&gt;1&lt;/id&gt;         &lt;name&gt;PARAMETER&lt;/name&gt; &lt;/Customer&gt; &lt;/new&gt;       &lt;/tuple&gt; &lt;/UpdateCustomer&gt; &lt;/SOAP:Body&gt;     &lt;/SOAP:Envelope&gt;</pre>	<pre>&lt;data&gt; &lt;UpdateCustomer   reply="yes"   xmlns="http://schemas.cordy   s.com/Ws-   AppServer/Testing"&gt; &lt;tuple&gt;     &lt;new&gt; &lt;Customer&gt; &lt;id&gt;1&lt;/id&gt;     &lt;name&gt;PARAMETER&lt;/name&gt; &lt;/Customer&gt; &lt;/new&gt;   &lt;/tuple&gt; &lt;/UpdateCustomer&gt; &lt;/data&gt;</pre>
No	<pre>&lt;SOAP:Envelope   xmlns:SOAP="http://schemas.xmlsoap.org/so   ap/envelope/"&gt; &lt;SOAP:Body&gt;     &lt;UpdateCustomer reply="no"       xmlns="http://schemas.cordys.com/Ws-       AppServer/Testing"&gt; &lt;tuple&gt; &lt;new&gt;         &lt;Customer&gt; &lt;id&gt;1&lt;/id&gt;         &lt;name&gt;PARAMETER&lt;/name&gt; &lt;/Customer&gt; &lt;/new&gt;       &lt;/tuple&gt; &lt;/UpdateCustomer&gt; &lt;/SOAP:Body&gt;     &lt;/SOAP:Envelope&gt;</pre>	<pre>&lt;data&gt; &lt;UpdateCustomer   reply="no"   xmlns="http://schemas.cordy   s.com/Ws-   AppServer/Testing"/&gt; &lt;/data&gt;</pre>

Examples of using this attribute when WS-AppServer is embedded in standalone mode:

<b>Reply</b>	<b>Code</b>
Yes	<code>BSF.getObjectManager().setNoReply(false);</code>
No	<code>BSF.getObjectManager().setNoReply(true);</code>

**Note:** JDBC connectors do not support insertion into tables with an auto-generated field as the primary key. In this case, setting `reply='no'` enables insertions.

### Using 'preserveSpace'

This attribute is used to preserve the trailing white space in the text, while rendering it in the SOAP response. It is used by the XQY layer and applies only to the response. It does not affect the actual information that is stored in the database. Therefore, if the original data has a trailing white space, that is stored in the database. The response may or may not contain the trailing white space depending upon the attribute setting.

The default value for this attribute is **false**, which truncates the white space at the end of the data (trailing space) in the response. If this attribute is set to **true**, the white space at the end of the data is preserved in the response.

**Note:** If you are testing this functionality using the AppWorks Platform SOAP Node Test tool, observe that when the attribute is set to **true**, irrespective of the number of spaces given in the request, the preserved trailing space in the response is restricted to a single space. This is because the Microsoft XML parser used in AppWorks Platform truncates the additional trailing spaces, even when the value is set to **true**. If you use any other program to test this functionality, the number of spaces shown in the response tally with the spaces that get stored in the database.

#### To set this attribute to true:

- In a SOAP request, add an attribute **preserveSpace** with the value **true** (`preserveSpace='true'`).
- In the method implementation, add an attribute **preserveSpace** with value **true**.

Examples of using this attribute when the WS-AppServer is in SOAP mode:

	If <b>preserveSpace=false</b>	If <b>preserveSpace=true</b>
Insert SOAP Request	<pre>&lt;SOAP:Envelope   xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/"&gt; &lt;SOAP:Body&gt;     &lt;UpdateCustomer       preserveSpace="false"       xmlns="http://schemas.cordys.com/Ws-AppServer/Testing"&gt; &lt;tuple&gt;         &lt;new&gt; &lt;Customer&gt; &lt;id&gt;1&lt;/id&gt; &lt;name&gt;           abc &lt;/name&gt; &lt;/Customer&gt; &lt;/new&gt;       &lt;/tuple&gt; &lt;/UpdateCustomer&gt;     &lt;/SOAP:Body&gt; &lt;/SOAP:Envelope&gt;</pre>	<pre>&lt;SOAP:Envelope   xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/"&gt; &lt;SOAP:Body&gt;     &lt;UpdateCustomer       preserveSpace="true"       xmlns="http://schemas.cordys.com/Ws-AppServer/Testing"&gt; &lt;tuple&gt;         &lt;new&gt; &lt;Customer&gt; &lt;id&gt;1&lt;/id&gt; &lt;name&gt;           abc &lt;/name&gt; &lt;/Customer&gt; &lt;/new&gt;       &lt;/tuple&gt; &lt;/UpdateCustomer&gt;     &lt;/SOAP:Body&gt; &lt;/SOAP:Envelope&gt;</pre>
Insert SOAP Response	<pre>&lt;data&gt; &lt;UpdateCustomer   preserveSpace="false"   xmlns="http://schemas.cordys.com/Ws-AppServer/Testing"&gt; &lt;tuple&gt;     &lt;new&gt; &lt;Customer&gt; &lt;id&gt;1&lt;/id&gt; &lt;name&gt;       abc&lt;/name&gt; &lt;/Customer&gt; &lt;/new&gt;</pre>	<pre>&lt;data&gt; &lt;UpdateCustomer   preserveSpace="true"   xmlns="http://schemas.cordys.com/Ws-AppServer/Testing"&gt; &lt;tuple&gt;     &lt;new&gt; &lt;Customer&gt; &lt;id&gt;1&lt;/id&gt; &lt;name&gt;       abc &lt;/name&gt; &lt;/Customer&gt; &lt;/new&gt;</pre>

	<b>If preserveSpace=false</b>	<b>If preserveSpace=true</b>
	</tuple> </UpdateCustomer> </data>	</tuple> </UpdateCustomer> </data>
Query SOAP Request	-	<GetCustomerObject preserveSpace="true" xmlns="http://schemas.cordys.com/W s-AppServer/Testing"> <id>1</id> </GetCustomerObject>
Query SOAP Response	-	<data> <GetCustomerObjectResponse preserveSpace="true" xmlns="http://schemas.cordys.com/W s-AppServer/Testing"> <tuple> <old> <Customer> <id>1</id> <name> abc </name> </Customer> </old> </tuple> </GetCustomerObjectResponse> </data>

Examples of using this attribute when WS-AppServer is in embedded mode:

<b>preserveSpace</b>	<b>Code</b>
Yes	BSF.getObjectManager() .setPreserveSpace(true);
No	BSF.getObjectManager() .setPreserveSpace(false);

### Using 'batchUpdate'

Batch update enables users to update the same set of columns for multiple records in a database table. Using batch update, you can send multiple updates (insert, update or delete requests) in a single update request, to be run as a batch rather than sending these requests separately. This helps improve the performance of the WS-AppServer.

To enable batch update, set the batchUpdate attribute to **yes** in the implementation of the specific update (<objectname>update) Web service operation. The default value is **no**.

**Note:** You cannot run batch update for the generic Web service operation update .

#### To set the attribute to yes:

- In the SOAP request, add an attribute **batchUpdate** with the value **yes**  
(batchUpdate='yes')
- In the method implementation, add an attribute **batchUpdate** with the value **yes**, to the update method stored in LDAP.

Examples of using this attribute when WS-AppServer is run in SOAP mode:

<b>batchUpdate</b>	<b>Yes</b>
Soap request	<SOAP:Envelope xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/"> <SOAP:Body> <UpdateCategories batchUpdate="yes" commandUpdate="no" preserveSpace="no" reply="yes" xmlns="http://schemas.cordys.com/db12313"> <tuple> <new> <Categories qAccess="0" qConstraint="0" qInit="0" qValues=""> <CategoryName>ships</CategoryName> <Description>shipDesc</Description> </Categories> </new> </tuple> <tuple> <new> <Categories qAccess="0" qConstraint="0" qInit="0" qValues=""> <CategoryName>Car</CategoryName> <Description>CarDesc</Description> </Categories> </new> </tuple> </UpdateCategories> </SOAP:Body> </SOAP:Envelope>
SOAP response	<data> <UpdateCategoriesResponse batchUpdate="yes" commandUpdate="no" preserveSpace="no" reply="yes" xmlns="http://schemas.cordys.com/db12313" xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/"> <tuple xmlns="http://schemas.cordys.com/db12313" xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/"> <new xmlns="http://schemas.cordys.com/db12313" xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/"> <Categories xmlns="http://schemas.cordys.com/db12313"> <CategoryName>ships</CategoryName> <Description>shipDesc</Description> </Categories> </new> </tuple> <tuple xmlns="http://schemas.cordys.com/db12313" xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/"> <new xmlns="http://schemas.cordys.com/db12313" xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/"> <Categories xmlns="http://schemas.cordys.com/db12313"> <CategoryName>Car</CategoryName> <Description>CarDesc</Description> </Categories> </new> </tuple> </UpdateCategoriesResponse> </data>

To use this attribute when WS-AppServer is in the standalone, embedded mode, use the following code:

```
BSF.getObjectManager().setBatchUpdate(true);
```

With `batchUpdate`, you can also use the attribute `batchSize` (`batchSize="number"`). However, this attribute can be used only in the embedded mode or by customizing the Java classes according to the following code:

```
BSF.getObjectManager().setBatchSize(100);
```

**Note:** For details, refer to the Java API.

### Using options in method implementation

---

You can set the attributes in method implementation in the following ways:

- To set the option for an update request, use the following code:

```
<implementation type="BsfUpdate"> <update commandUpdate="yes|no"
reply="yes|no" preserveSpace="true|false" batchUpdate="yes|no"/>
</implementation>
```

- To set the option for a query request (GetXXXXObjects), use the following code:

```
<implementation type="BsfJavaCall" preserveSpace="true">
<class>com.myapp.Customer</class> <method scope="out"
dt="java:com.myapp.Customer[]" ct="tuples"
wt="true">getCustomerObjects</method> <parameters> <fromId dt="int"/> <toId
dt="int"/> </parameters> </implementation>
```

**Note:** The attributes set in SOAP requests take precedence over the method implementation.

## License configuration interface

UI Element	Description
License Mode	The type of AppWorks Platform installation, that is Single or Multiple.
Treat this Computer as Master	Select this option to consider the host computer as the master. Clear the selection to treat this computer as a contributor.
Registered Customer Name	Name of the customer who holds the AppWorks Platform license.
Registered Site Name	Any string assigned by AppWorks Platform.
Master Computer Name	Name of the master computer. <b>Note:</b> This field is enabled only if the <b>License Mode</b> is <b>Multiple</b> and the <b>Treat this Computer as Master</b> option is not selected.
Master Web Site Port Number	Port number of the computer that is treated as master compute <b>Note:</b> This field is enabled only if the <b>License Mode</b> is <b>Multiple</b> and the <b>Treat this Computer as Master</b> option is not selected.
Key	The license key currently being used. <b>Note:</b> The validity period of the current key is displayed below the box.
Send Usage Report Automatically	Whether the usage report has to be sent automatically to the AppWorks Platform License Service every month.

# Lightweight Directory Access Protocol

Lightweight Directory Access Protocol (LDAP) is a networking protocol that enables you to query and modify a directory service based on TCP/IP. An LDAP directory enables you to locate organizations, individuals, and other resources, such as files and devices on the Internet or on a corporate intranet.

AppWorks Platform LDAP serves as a central repository of information on organizations, roles, applications, middleware, and URIs. AppWorks Platform LDAP is functionally known as OpenText CARS. This information is stored in a specific format in the LDAP schema installed by the OpenText CARS installer.

## Loading a certificate

### To load a certificate:

1. To navigate to the Load New Certificate dialog box, click  (Lookup Certificate).
2. Select **Upload Certificate** or **Paste Base64 Encoded Certificate** to attach the certificate.
3. Depending on the option you selected, do one of the following:

Option	Step
Upload Certificate	<ol style="list-style-type: none"><li>1. To select the certificate from the saved location, click  (Lookup) beside <b>Path</b>.</li><li>2. Select the type of certificate. It can be a Binary or a Base64 Encoded certificate.</li><li>3. Click <b>OK</b>.</li></ol>
Paste Base64 Encoded Certificate	<ol style="list-style-type: none"><li>1. Paste the Base64 encoded form of the certificate in <b>Text</b>.</li><li>2. Click <b>OK</b>.</li></ol>

## Localizing an email model

You can localize email models to display messages to users in different languages. For example, you can localize the models to display messages in French for users who have selected French as their preferred language.

### To localize an email model:

1. From the **/Cordys/notification/emailmodels/models** of the XML store, open the XML file containing the email model.
2. Create a copy of the XML file in the same directory.

- 
3. Open a copy of the XML file, make the required changes, and save it.
  4. Rename the file by adding the locale as the suffix, for example, \_fr\_FR for French.

**Note:**

- The name of the email model must be the same as the name of the localized email model. For example, if the name of the email model is **Order**, the name of the localized model must be **Order\_fr\_FR**.
- The browser of the receiver must support the character set of the selected locale.

The email model is localized and can be invoked by providing the locale in the LANGUAGE tag of the Deliver Message SOAP request in the *AppWorks Platform API Reference Guide*.

If the locale specified in the message does not exist in the XML store, the AppWorks Platform default locale setting is used while installing AppWorks Platform. It is stored as a property in the `wcp.properties` file.

## Manage case instances interface

This topic describes the tabs on the Case Instance Manager UI:

### Overview

The Overview tab depicts a graphical view of all the available case instances for a case model. It depicts the total case instances, model status, status or state of the case instances. You can filter using the **Published At** filter criteria. Data is displayed in the form of a bar chart for ease of understanding. You can perform the following tasks:

Field	Description
Published At	When there is a large number of case instances, and you want to view all the instances published at different times, filter using the published time. Select a published time to view a case instance, or select <b>All</b> .
Filter Instances by > Status	When there is a large number of case instances, and you want to view all the instances based on their different statuses as a graph, filter using the status. For example, view all the case instances whose status is new or completed.
Filter Instances by > State	When there is a large number of case instances and you want to view all the instances based on the state as a graph, filter using the state. For example, view all the case instances that are in final state.

### Detail View

The Detail View tab displays all the available case instances of a specific case model in a table. You can filter the case instances using the following options:

Field	Description
Published At	When there is a large number of case instances and you want to view all the instances published at different times, filter using the published time. Select a published time to view a case instance, or select <b>All</b> to view instances published at different times.
Filter Instances by > State	When there is a large number of case instances and you want to view, monitor, or perform operations on selected instances with a specific state, filter using their state. Select this option to view all the case instances by their state. For example, view all the case instances that are in final state. You can select from: All, Final State, or Default State.
Filter Instances by > Status	<p>When there is a large number of case instances and you want to view, monitor, or perform operations on only certain case instances with a specific status, filter the case instances using their status. Select this option to view all the case instances based on their statuses. For example, view all the case instances whose statuses are new or completed.</p> <p>You can select from:</p> <ul style="list-style-type: none"> <li>■ <b>All:</b> All the available case instances are displayed.</li> <li>■ <b>New:</b> A case is new when its instance is created and no activities are released. However, the initial activities and automatic followups for the state entry and case start are released. All the newly-created case instances are displayed.</li> <li>■ <b>In Progress:</b> A case is in progress only when a few activities of the case are complete and the remaining activities are awaiting completion. All the case instances that are in progress or are being monitored are displayed.</li> <li>■ <b>Overdue:</b> All the case instances that are overdue are displayed.</li> <li>■ <b>Completed:</b> A case is completed when all the activities in the case are completed as defined. All the case instances that are completed are displayed.</li> <li>■ <b>Aborted:</b> A case is aborted when an activity fails in the case flow, the activity is aborted, and in turn, the case is also aborted. All the aborted case instances are displayed.</li> </ul>

Field	Description
	<ul style="list-style-type: none"> <li>■ <b>Suspended:</b> A case is suspended when an administrator closes the case in the midst of execution, and all the activities that are in progress are suspended. The case can be resumed from where it is suspended. All the suspended case instances are displayed.</li> <li>■ <b>Terminated:</b> A terminated case is one that has been closed in the midst of execution and all its activities that are in progress are stopped. All the terminated case instances are displayed.</li> </ul>

### Detail View: table description

The following describes the fields that appear in the Detail View table.

Field	Description
Status	Current business status of the case instance.
Current Functional State	<p>Can also be a milestone. Business status of the case can be a part of this milestone. For example, a case can be closed or completed, but the business status can be approved or rejected.</p> <ul style="list-style-type: none"> <li>■ For a terminated case, <b>Current Functional State</b> displays the state where the case was terminated.</li> <li>■ For a suspended case, the <b>Current Functional State</b> displays the state from where the case was suspended.</li> <li>■ For a closed case, the <b>Current Functional State</b> displays the state where the case came to an end.</li> </ul> <p><b>Note:</b></p> <ul style="list-style-type: none"> <li>■ <b>Default State</b> is displayed for the activity that is not associated with any state, unless a specific value for the same is provided in the related model.</li> <li>■ <b>Case Identifier</b> columns are always listed in the same position. Even if you move or reorder these columns in the grid, the order is restored to their original position after any operation. This is applicable only for the columns that are related to the case identifiers.</li> </ul>
Started On	Displays the start day, date, and time at which the Case instance was started. For example, Tuesday, November 11, 2008 3:48:33 PM.

Field	Description
Started By	Name of the Case worker who started the Case instance. For example, Luca Raggi.
Acted On	Day, date, and time at which the case instance was acted upon, for example, Tuesday, November 11, 2008 3:48:33 PM.
Acted By	Name of the case worker who acted upon the case instance, for example, Luca Raggi.
Due On	Day, date, and time at which the case instance is due to be closed, for example, Monday, November 14, 2008 3:48:33 PM.

## MDM data entity interface

Use the MDM data entity interface to fill the fields on the MDM data entity interface.

The following table describes the Other fields on the MDM data entity interface:

Field name	Description
Name	Specify name of the data entity.
Description	Provide description for the data entity.
Entity Type	<p>Options for the Entity Type describe the use of entity for the flow of data to and from it. Select from:</p> <ul style="list-style-type: none"> <li>▪ <b>Send and Receive Data:</b> Default option to send and receive data between the hub and the spoke entities.</li> <li>▪ <b>Send Data only:</b> Send data.</li> <li>▪ <b>Receive Data only:</b> Receive data.</li> </ul> <p><b>Note:</b> <b>Send and Receive Data</b> is the only option for a hub data entity.</p>
Business Object Lifecycle	<p>This check box is enabled by default to manage the lifecycle of a business object. For simple synchronization, clear this check box. The <b>Sniffer</b> tab opens in place of the Business Object Lifecycle.</p> <p><b>Note:</b> The <b>Enable Business Object Lifecycle</b> check box opens only for a hub data entity.</p>
Business Object	<p>Each business object is represented as a schema fragment in the AppWorks Platform environment, on which the rest of the modeling activities are based, for example, it is the basis for defining data transformation, and business rules.</p> <p><b>To select a business object:</b></p>

Field name	Description
	<ol style="list-style-type: none"> <li>1. Click  (Lookup Record) corresponding to the Business Object. The Select a Schema Fragment Object dialog box opens.</li> <li>2. Select the schema fragment object. The selected schema fragment object opens in the text box corresponding to Selected Document, and OK is enabled.</li> <li>3. Click <b>OK</b>. The selected schema fragment opens in the Business Object text box.</li> <li>4. Click  (Select Keys) to select the fields as primary keys. The Select Key Fields dialog box opens with a list of fields.</li> <li>5. Select the check boxes corresponding to the fields under the Key column as the primary keys, enabling MDM to uniquely identify an instance of the selected schema fragment object.</li> <li>6. Under the Data Type column click <b>Select the Data Type</b> for the check boxes that you have selected under the Key column to view the lists. The data type list opens with the list of data types under the Data Type column.</li> <li>7. Select the relevant data type and click <b>OK</b>.</li> </ol>

The following table describes the fields on the Business Object Lifecycle tab:

Field name	Description
State Model	Graphical depiction of the lifecycle management model defined for the hub entity. Specify a state model. Click  (Lookup Record) to display a list of state models that were already created. For information on designing state models, see Designing a State Model in the <i>AppWorks Platform Advanced Development Guide</i> .
Event Triggers	Communication channels with which the behavior of a state model is controlled. Inside a state model, you can model to control the behavior of the object reacting to different events leading to different execution paths. Select the trigger type that you have specified while designing the state model: <ul style="list-style-type: none"> <li>■ <b>Insert Trigger:</b> When a record at the spoke entity is inserted, it enters the first state of the state engine. If this record already exists in the state engine, select the event name using which this record moves to another state.</li> <li>■ <b>Update Trigger:</b> When a record at the spoke entity is updated, select the event name using which this record moves to another state. If this record does not exist in the state engine, it enters the first state of the state model.</li> </ul>

Field name	Description
	<ul style="list-style-type: none"> <li>▪ <b>Delete Trigger:</b> When a record at the spoke entity is deleted, select the event name using which this record moves to another state. If this record does not exist in the state engine, it enters the first state of the state model.</li> </ul>
Enable Registry	<p>Registry is a collection of entries. In MDM, a registry holds a collection of links that point to the corresponding records in different systems for a master data record.</p> <p><b>To enable registry:</b></p> <ol style="list-style-type: none"> <li>1. Select the <b>Enable Registry</b> check box. The ID Generator opens.</li> <li>2. Click  (Lookup Record) next to the ID Generator. The Select a sequence generator dialog box opens.</li> <li>3. Select the relevant <b>Sequence Generator</b> that is already created. The Sequence Generator dialog box opens if you have selected a default option. For information on the fields on the Sequence Generator, see the <a href="#">Sequence Generator Properties interface</a>.</li> </ol> <p><b>Note:</b> If you have selected a customized sequence generator, the property sheet opens with the specified properties. You can edit these properties.</p>
Enable Data Quality	<p>Refine the quality of master data that enters the hub. MDM contains pure and accurate master data with data quality tool. Select this check box to enable data quality. The Data Quality tab with the following fields appear on selecting the check box:</p> <ul style="list-style-type: none"> <li>▪ <b>Data Quality Plan:</b> Select the data quality plan that is already created or create a new data quality plan. For details see Creating a Data Quality Plan in the <i>AppWorks Platform Advanced Development Guide</i>.</li> <li><b>Note:</b> Select a data quality plan if you have to configure a hub entity to use a third-party data quality management tool, and integrate the third-party data quality management tool with MDM.</li> <li>▪ <b>Match Object Life Cycle:</b> Select a state model using  (Lookup Record). Select the state model for the lifecycle of the candidate object, if the selected data quality plan has a <b>Match</b> operation defined and returns a set of candidate objects as a response to this operation. Each candidate object undergoes the lifecycle of the selected state model.</li> <li>▪ <b>Match Object Table Name:</b> Type the table name that saves details of the candidate object changes for a business object. View this in the Advanced tab.</li> </ul>

The following table describes the fields on the Web Services tab:

Field name	Description
Web Service Type	<p>In the MDM synchronization model, the data integration frameworks work with Web services to communicate with different systems. The type of Web service indicates whether the Web service is an in-house AppWorks Platform Web service or an externally hosted Web service.</p> <ul style="list-style-type: none"> <li>■ Select from:</li> </ul> <p><b>Transactional:</b> If you have a transactional Web service that conforms to the AppWorks Platform standard protocol.</p> <p><b>Note:</b> The Transactional option is selected for a hub entity by default, and the non-transactional option is disabled. For details, see <i>Transactional and Non-transactional Web Service</i> in the <i>AppWorks Platform Advanced Development Guide</i>.</p> <ul style="list-style-type: none"> <li>■ <b>Non-Transactional:</b> If you have a non-transactional Web service that does not conform to the AppWorks Platform standard protocol.</li> </ul> <p><b>Note:</b> The Web Service Transformations tab opens on selecting the Non-transactional option. For details, see <i>Transactional and Non-transactional Web Service</i> in the <i>AppWorks Platform Advanced Development Guide</i>.</p>
Use Generic Update Web Service	<p>A generic update service can perform all the operations on a business object (insert, update, and delete). Select the Use Generic Update Web Service check box to insert, modify, and delete operations using a single Web service, such as the <b>Update</b> and <b>UpdateBusinessObject</b> Web services generated in AppWorks Platform over database tables. Clear the check box to select three distinct Web services to insert, modify, and delete data. You can select the Web services to read data on the Read tab, and update data on the Modify tab.</p> <p><b>Note:</b></p> <ul style="list-style-type: none"> <li>■ The Use Generic Update Web Service check box is selected by default, on selecting the Transactional option, and is disabled on selecting the Non-transactional type of Web service.</li> <li>■ The Use Generic Update Web Service check box does not open if you choose the Send Data only option of the Entity Type, while creating a data entity.</li> </ul>
Read tab	Click <b>Read to select a Web service to read data</b> , and specify Web service request parameters.
Modify tab	Click <b>Modify</b> , to select a Web service to update data.
Web Service Transformations tab	Each Web service requires input parameters in a specific structure as defined in the WSDL. You have to map the business object to the input structure of the Web service using data transformations. Click Web Service Transformations to specify the Read, Update, Insert, and Delete

Field name	Description
	<p>transformations to transform the non-transactional Web service into a transactional Web service that conforms to the AppWorks Platform standard protocol.</p> <p><b>Note:</b> The Web Service Transformations tab opens only if you choose the non-transactional option of the entity type while creating a data entity.</p>

The following table describes the fields on the Sniffer tab:

Field name	Description
Sniffer Type	<p>Sniffers are components in data integration framework of MDM. They capture the changes that occur in the source system and publish them to the subscribing clients. Select from:</p> <ul style="list-style-type: none"> <li>▪ <b>Default:</b> Select one of the default sniffers that AppWorks Platform provides. The default Sniffer appears with a list .</li> <li>▪ <b>Custom:</b> Specify the sniffer that you have already configured. The custom Sniffer appears with  (Lookup Record).</li> </ul> <p><b>Note:</b> Hub data entity is configured to trigger Sniffer.</p>
Default Sniffer	Select one of the default sniffers that AppWorks Platform provides. See <a href="#">MDM default sniffer fields</a> for more information on the properties and fields of the default sniffers.
Custom Sniffer	<p>To specify the sniffer:</p> <ol style="list-style-type: none"> <li>1. Click  (Lookup Record). The Select a sniffer dialog box opens.</li> <li>2. Select the sniffer if you have already created one or create a Custom MDM Sniffer. See Configuring an MDM Sniffer in the <i>AppWorks Platform Advanced Development Guide</i>.</li> <li>3. Click <b>OK</b>.</li> </ol> <p><b>Note:</b> The fields for a custom sniffer depend on the parameters you define.</p>
Enable Schedule For Sniffer	<p>There may be business needs where you have to synchronize data realtime or at the scheduled time. For some business needs, it is mandatory to synchronize data after business hours. Hence, you can use the scheduling option to meet the purpose. Select the check box if you want to watch the changes at regular intervals. The <b>Schedule Details</b> tab opens. See the <a href="#">Schedule interface fields</a> for information on fields on the <b>Schedule Details</b> tab.</p> <p><b>Note:</b> Configuring the Schedule Details tab is optional for all sniffer types except the <b>Comparator</b> sniffer.</p>

The following table describes the fields on the Advanced tab:

Field name	Description
Read Web Service Cursor	<p>Reads data from the source system. The database may contain a lot of data. The data integration framework pulls this data in smaller batches to process them in chunks. You can configure the batch size. Type the number of records you want to retrieve in the text box next to the Read Web Service cursor.</p> <p><b>Note:</b> Read Web Service cursor appears only when you select the Use Generic Update Web Service check box. The default value is 1000. You can change the number of records to be retrieved.</p>
Update Web Service Batch Size	<p>Updates data from the source system. The database may contain a lot of data. The data integration framework must pull this data in smaller batches to process them in chunks. You can configure the batch size. Type the number of records you want to retrieve to update data in the text box next to Update Web Service Batch Size.</p> <p><b>Note:</b> Update Web Service Batch Size appears only when you select the Use Generic Update Web Service check box. The default value is 1000. You can change the number of records to be retrieved.</p>
Insert Web Service Batch Size	<p>Inserts data from the source system. The database may contain a lot of data. The data integration framework must pull this data in smaller batches to process them in chunks. You can configure the batch size. Type the number of records you have to fetch to insert data in the text box next to Insert Web Service Batch Size.</p> <p><b>Note:</b> The Insert Web Service Batch Size appears only for a spoke data entity. The default value is 1000. You may change the number of records to be retrieved depending on the requirement.</p>
Delete Web Service Batch Size	<p>Deletes data from the source system. The database may contain a lot of data. The data integration framework must pull this data in smaller batches to process them in chunks. You can configure the batch size. Type the number of records you want to retrieve to insert data.</p> <p><b>Note:</b> Delete Web Service Batch Size appears only for a spoke data entity. The default value is 1000. You can change the number of records to be retrieved.</p>
Sniffer Audit Table Name	<p>Sniffers work with an audit table. Every sniffer in the synchronization framework stores the changes in the audit table for further asynchronous processing. The database table name that stores all the audit information appears by default.</p>
State Instance Table Name	<p>State management or life cycle management of the master data object is done in a staging area, which is different from the final master data repository where this object has to be updated. State instance table is the staging area for this hub object. The specified database table name appears in the Advanced tab by default. The database table saves the details of state changes for a business</p>

<b>Field name</b>	<b>Description</b>
	<p>object.</p> <p><b>Note:</b> State Instance Table Name appears only for a hub data entity.</p>
Registry Table Name	<p>Collection of cross linkages of master data records. These linkages are stored in this registry table. Specify a name for the database table, that stores the registry details.</p> <p><b>Note:</b> Registry Table Name appears in the Advanced tab only if you have selected the <b>Enable Registry</b> check box for a hub data entity.</p>
Match Object Table Name	<p>In life cycle management of master data object, the data steward has to deal with duplicate objects that match the incoming object. Data steward must sometimes manually resolve these data conflicts and accept the incoming record as distinct or mark it as duplicate. This table is the repository to hold such match objects. Type the table name that saves details of the candidate object changes for a business object.</p> <p><b>Note:</b> Match Object Table Name appears in the Advanced tab only if you have selected the <b>Enable Data Quality</b> check box for a hub data entity.</p>

## MDM data store interface

The following table contains the fields on the data store interface:

<b>Field name</b>	<b>Description</b>
Automatically receive data	<p>Automatically synchronizes the data store with the hub, once the MDM model containing this data store is published. If not selected, this data store must be manually synchronized.</p> <p><b>Note:</b> This field is disabled for a hub data store.</p>
Description	Brief description of the data store.
Download Database Scripts	<p>When the model containing this data store is published, certain changes such as creation of audit tables and triggers are made to the actual database that this data store represents. Select this check box to:</p> <ul style="list-style-type: none"> <li>■ Prevent these changes.</li> <li>■ Save the database scripts that affect those changes in a file.</li> </ul> <p>The Database Administrator (DBA) can review the scripts and then run them manually on the database. This is useful to control each change made to the database. See <a href="#">MDM admin settings</a> for more information on downloading database scripts.</p>
Name	Name of the data store.
Publisher	Select a service group from the list of publishers that can be used by the

Field name	Description
Service Group	<p>data store. This property is automatically saved on selection. Each data store must have its own unique publisher. For information on creating a publisher service group, see the <a href="#">MDM publisher connector configuration interface</a>.</p> <p><b>Note:</b></p> <ul style="list-style-type: none"> <li>▪ The selected MDM publisher service groups and the associated MDM service group must not be renamed after you publish the MDM model, with which the data store is associated.</li> <li>▪ For a hub data store, the type of the publisher service group selected must be <b>Hub Publisher</b>.</li> </ul>

## MDM default sniffer fields

### Trigger Sniffer

A change in spoke data that uses trigger-based sniffing triggers a change request to the hub. The hub also uses trigger-based sniffing. Therefore, any change in its data triggers a change request to all the subscribed spokes. Any data update to a spoke returns to it after being propagated to the hub, and consequently results in an infinite loop. This is prevented by providing the hub with a mechanism to identify the individual spokes that use trigger-based sniffing to ensure that the hub does not return a change request to the spoke from where it came.

To facilitate identification, you must create a new database user, the MDM database User, with appropriate access rights for each spoke that uses trigger-based sniffing. The hub uses the username for its transaction with the spoke.

**Note:** Ensure that the MDM User has Select, Insert, Update, and Delete permissions on any type of data repositories.

The following fields are displayed when the Trigger sniffer is selected.

Field	Description
MDM User	<p>User who has access to the database. The user's access permission on the database must be the same as the type of spoke defined.</p> <ul style="list-style-type: none"> <li>▪ If the spoke is send only, the user must have read permission on the database.</li> <li>▪ If the spoke is receive only, the user must have write permission on the database.</li> <li>▪ If the spoke is send and receive or if it is a hub, the user must have read and write permissions on the database.</li> </ul> <p>This user is required only for trigger based sniffers on the entities.</p>

<b>Field</b>	<b>Description</b>
Record Type	<p>Options to select:</p> <ul style="list-style-type: none"> <li>■ All Changes: The audit table must record all changes to the data entity.</li> <li>■ Last Change: The audit table must record only the last change made to the data entity.</li> </ul>

### **Generic Timestamp Sniffer**

The following fields are displayed when the Generic Timestamp sniffer is selected.

<b>Field</b>	<b>Description</b>
Column Name	Column name in the data store that contains the time stamp.
Correction Factor (in ms)	Time for the correction factor. The data entities can be changed immediately after or during data integration. The next integration must accommodate this change. Therefore, a correction factor ensures that the next integration starts earlier than the specified time.
Sniffer Start Date	Start date when sniffing must begin. When the date is selected from the calendar, the current time is also selected. You can manually modify the time according to your needs.

### **Oracle10g Sniffer**

The following fields are displayed when the Oracle10g sniffer is selected.

<b>Field</b>	<b>Description</b>
Correction Factor (in ms)	Time for the correction factor. The data entities can be changed immediately after or during data integration. The next integration must accommodate this change. Therefore, a correction factor ensures that the next integration starts earlier than the specified time.
Record	<p>Options to select:</p> <ul style="list-style-type: none"> <li>■ All Changes: The audit table must record all changes to the data entity.</li> <li>■ Last Change: The audit table must record only the last change made to the data entity.</li> </ul>
Start Time	Start date when sniffing must begin. When the date is selected from the calendar, the current time is also selected. You can manually modify the time according to your needs.

### **Message Queue**

The following fields are displayed when the Message Queue sniffer is selected.

Field	Description
JMS Vendor	Type of JMS vendor, for example, Sun, IBM, or Open JMS.
JNDI Parameters tab	Location where the JMS administered objects can be found (Local file system, LDAP, or any other location). For any location, type the Class Name and URL. If required, type the User Name and Password for authentication.
Receiver Details tab	On the Receiver Details tab, select the type of JMS administered objects you are using (Queue type or Topic type). For each, type the Factory Lookup Name and the Topic Lookup Name. Select the message processing mechanism to use. For the default process, type the Event Name and the conditions for Insert, Update, and Delete operations. For a custom process, type the class name to use.

For more information about the sniffer fields, see [Message Queue sniffer example](#).

### Comparator Sniffer

The Comparator sniffer compares the data entity at different time slots and checks for data changes. There are no parameters to configure for this sniffer. However, it is mandatory to configure the sniffer schedule. See [Schedule interface fields](#).

The Comparator sniffer is active only once in the time period of the entire schedule. After the Comparator sniffer has sniffed in that interval, it does not sniff again until the schedule is triggered. If the sniffer has already sniffed once in that interval, it does not pick the latest changes even if the schedule is active and there are new changes coming in. It must wait until the next schedule is triggered.

The Comparator sniffer enables data comparison using snapshots. A snapshot is a set of data, which is used to compare different data sets to determine the modified data set. The pull method defined on a particular entity is used to read the data and create a data snapshot. If a custom Web Service is used, the Web Service must ensure that data is always pulled in the same order such as a sorted order. If a different set of data is read each time data is pulled, different snapshots are created. In this case, comparison of the data sets yields unknown results.

## MDM-specific fields

Each data entity in the master data repository must maintain some MDM-specific information. For each data entity in the master data repository, add seven MDM-specific additional columns. To create scripts to add columns you can:

- Run the scripts.
- Create the scripts manually.

## To run scripts to create the columns:

**Note:** Use the utility-stored procedure creation script, available at <<Cordys\_HOME>>\components\mdm\dbscripts\applicationlocation.

1. Open a database client tool, depending on the database used for the MDM-hub application database configuration.
2. Go to the file system location <<Cordys\_HOME>>\components\mdm\dbscripts\application and choose the required .sql file.
3. Run the .sql script from the tool.  
A Stored Procedure **ADDMDMHUBFIELDS** is created in the database.
4. Run the newly created stored procedure with comma-separated table names.

### Example:

If your master data repository is using the MS SQL server, you must:

- i. Connect to the MS SQL server using the SQL-server client.
- ii. Connect to the corresponding database.
- iii. Run the scripts in components\mdm\dbscripts\application\MDM\_SQLSERVER\_UTILSCRIPTS.sql.  
This creates the **ADDMDMHUBFIELDS** procedure in the selected database.
- iv. Run the stored procedure using **EXEC ADDMDMHUBFIELDS 'TABLE1, TABLE2'** where TABLE1 and TABLE2 are the table names in the database where you want to add the MDM-specific fields.

## To create the scripts manually:

Based on the database used, do one of the following in the table schema:

### Oracle Database

If you are using an Oracle database as the master database, create the following columns:

Field	Data type	Length	Null option
RCORDELETED	NUMBER	1	NULL
RCOREXPIRY	DATE		NULL
RCORGUID	VARCHAR2	50	NULL
RCORLASTMODIFIED	DATE		NULL
RCORLASTMODIFIEDBY	VARCHAR2	50	NULL
RCOROPERATION	NUMBER	1	NULL
RCOROWNER	VARCHAR2	1000	NULL

### MS SQL Database

If you are using an MS SQL database as the master database, create the following columns:

<b>Field</b>	<b>Data type</b>	<b>Length</b>	<b>Null option</b>
RCORGID	VARCHAR	50	NULL
RCORDELETED	INT		NULL
RCOROWNER	NVARCHAR	1000	NULL
RCORLASTMODIFIED	DATETIME		NULL
RCOREXPIRY	DATETIME		NULL
RCORLASTMODIFIEDBY	NVARCHAR	50	NULL
RCOROPERATION	INT		NULL

### **DB2 Database**

If you are using a DB2 database as the master database, create the following columns:

<b>Field</b>	<b>Data type</b>	<b>Length</b>	<b>Null option</b>
RCORDELETED	INT		NULL
RCOREXPIRY	TIMESTAMP		NULL
RCORGID	VARCHAR	50	NULL
RCORLASTMODIFIED	TIMESTAMP		NULL
RCORLASTMODIFIEDBY	NVARCHAR	50	NULL
RCOROPERATION	INT		NULL
RCOROWNER	NVARCHAR	1000	NULL

### **PostgreSQL Database**

If you are using a PostgreSQL database as the master database, create the following columns:

<b>Field</b>	<b>Data type</b>	<b>Length</b>	<b>Null option</b>
RCORGID	VARCHAR	50	NULL
RCORDELETED	INT		NULL
RCOROWNER	VARCHAR	1000	NULL
RCORLASTMODIFIED	TIMESTAMP	0	NULL
RCOREXPIRY	TIMESTAMP	0	NULL
RCORLASTMODIFIEDBY	VARCHAR	50	NULL
RCOROPERATION	INT		NULL

## Memory status interface

The following table describes the parameters in the memory status interface:

Parameter	Description
Virtual Memory Usage	Virtual memory allocated to the service container.
Resident Memory Usage	Physical memory currently allocated to each service container.
NOM Nodes Memory	Memory for NOM nodes.
NOM Memory	NOM memory details.

## Message debugger

Use the message debugger to view and validate SOAP messages being sent to or received from the back end. The function of the icons that are displayed on the message debugger toolbar are as follows:

Interface element	Function
Application manager	Displays the applications or libraries being run.
Arrow	Enables the back and forward buttons. It is green when enabled and black when disabled.
Auto Format	Select this check box to remove formatting for SOAP messages. It is selected by default.
Back	Navigates to the previous page in the cache.
Continue	Resumes debugging.
Forward	Navigates to the next page in the cache.
Pause	Stops debugging temporarily.
Stop	Stops debugging.

## Message Queue sniffer example

Let us assume that the JMS provider is Sun Microsystems and the JMS Administrator has created the following administered objects of type Queue with lookup names and stored them in a file system with path C:\Bindings:

- MDMQueue
- MDMQueueConnectionFactory

## Configuring the JNDI Parameters tab

To locate and access these administered objects, the user must provide values for the necessary Java Naming Directory Interface (JNDI) parameters like Class Name and Provider URL. The repository is a file system and hence, the corresponding values for Class Name and URL are:

Type of repository	Class name	URL
File System	com.sun.jndi.fscontext.RefFSContextFactory	file:///C:/Bindings
LDAP	com.sun.jndi.ldap.LdapCtxFactory	ldap://cin0707-vm01:6717/ou=javaobjects,dc=vanenburgh,dc=com

The following table lists the Repository name and its respective Class name used by JNDI to locate the resources:

Repository name	Class name
File System	com.sun.jndi.fscontext.RefFSContextFactory
LDAP	com.sun.jndi.ldap.LdapCtxFactory
RMI	com.sun.jndi.rmi.registry.RegistryContextFactory
CORBA	com.sun.jndi.cosnaming.CNCtxFactory
DNS	com.sun.jndi.dns.DnsContextFactory

## Configuring the Receiver Details tab

Specify the values for the Queue Type in the Receiver details tab:

Field name	Value
Factory Lookup Name	MDMQueueConnectionFactory
Queue Lookup Name	MDMQueue

### Message Processing Area

Message Processing is a mechanism to handle the incoming messages from the JMS server and perform the necessary event and business object translation. The following are the types of incoming messages that a JMS provider supports:

- 
- TextMessage
  - ByteMessage
  - MapMessage
  - ObjectMessage
  - StreamMessage

However, the input to the sniffer must be an XML associated with a database action like insert, update, or delete. So, once a client receives a message from a JMS server, some extra processing must be performed to convert that message in to a format expected by the sniffer. There are two types of Message processing:

- **Default**

In default MDM processing the following are the assumptions:

- Incoming message is a TextMessage
- Its payload contains XML in text format
- Database specific events like insert, update, and delete are specified in the TextMessage application properties like (key, value) pair in which the value of the Event Name is the key and the value of insert, update, or delete is the value

Therefore, the key name is database\_action and its value can be new/new\_old/old.

- **Custom**

If the user wants to perform a different kind of business object or event translation other than the default MDM process then the user can provide a custom class which does the necessary business object and event translation. This custom class must implement `com.cordys.rcor.publisher.sniffer.messaging.IMessageHandler` interface.

Therefore, the MapMessageHandler is the custom class given in the class name of the custom process.

## Message repository

A message repository is a collection of all the localized log messages defined in the system. All log messages are now a part of the AppWorks Platform Message Code Framework (MCF).

A message in a message repository has a `MessageID` associated with it. This identifies the message text and description. All messages are defined under a `<MessageBundle>`, which denotes the context of all messages under it. Each `<Message>` contains the details of a particular message. Specify the Message ID to enable the administrator or the developer identify such log messages.

Sample message:

```
<MessageBundle id="Cordys.ESBServer.Messages">
    Message id="methodInvoked">
```

```

<MessageText>Invoked the method.</MessageText>
<Description/>
<Annotations>
    <DocumentationURL/>
</Annotations>
</Message>
<Message id="methodInvokeError">
    <MessageText>Exception while invoking the
method.</MessageText>
    <Description/>
    <Annotations>
        <DocumentationURL/>
    </Annotations>
</Message>
</MessageBundle>

```

The following attributes are used in the message repository file:

<b>Attribute</b>	<b>Description</b>
Annotations	Optional. Link to a detailed description of the log message.
Description	Optional. Describes the log messages.
ID	Associated with the message for defining and identifying log messages.
Message Text	Description of the log messages. Provides a solution to fix the error.

## Method properties interface

Use these properties determine the behavior of the method of the model to which it is being added.

**Note:** General and Other tabs are displayed only when you right-click an existing WS-AppServer model, and select **Add > Method**. If you select a method from the Methods pane in the Method Properties interface, only the Property fields are displayed in the Method Properties pane on the right.

---

<b>Tab</b>	<b>Property</b>	<b>Description</b>
General	Name	Name of the method.

Tab	Property	Description
	Return type	<p>Return data type of the related object. Choose from the following data types:</p> <ul style="list-style-type: none"> <li>▪ Unsigned Byte (ui1): One-byte unsigned integer</li> <li>▪ Unsigned Short (ui2): Two-byte unsigned integer</li> <li>▪ Unsigned Integer (ui4): Four-byte unsigned integer</li> <li>▪ Unsigned Long (ui8): Eight-byte unsigned integer</li> <li>▪ Short Integer (i2): Two-byte short integer</li> <li>▪ Integer (i4): Four-byte integer</li> <li>▪ Long Integer (i8): Eight-byte long integer</li> <li>▪ Float (r4): Floating point number with four-byte encoding</li> <li>▪ Double (r8): Floating point number with double precision</li> <li>▪ Currency (cy): Floating point number that represents currency data types</li> <li>▪ Select one of the following data types to represent XML as an integer in Java: <ul style="list-style-type: none"> <li>• NOM xml node (xml)</li> <li>• DOM Element</li> </ul> </li> </ul> <p><b>Note:</b> When either of the data types is used, the Java class for that WS-AppServer model is generated with the following code, facilitating addition of related custom logic to process XML data. For details, see Extension Class in the <i>AppWorks Platform Advanced Development Guide</i>.</p> <pre>public static org.w3c.dom.Element DOM (org.w3c.dom.Element Dom) {     // TODO implement body     return null; }</pre> <p>To add custom logic to this Java code, you can use the <code>getObjectData()</code> API as shown in the following code snippet:</p> <pre>/* Returns the BusObject Data as a DOM Element  * @return Document DOM Element containing the</pre>

Tab	Property	Description
	<ul style="list-style-type: none"> <li>■ Unsigned Byte (ui1): One-byte unsigned integer</li> <li>■ Unsigned Short (ui2): Two-byte unsigned integer</li> <li>■ Unsigned Integer (ui4): Four-byte unsigned integer</li> <li>■ Unsigned Long (ui8): Eight-byte unsigned integer</li> <li>■ Short Integer (i2): Two- byte short integer</li> <li>■ Integer (i4): Four-byte integer</li> <li>■ Long Integer (i8): Eight- byte long integer</li> <li>■ Float (r4): Floating point number with four-byte encoding</li> <li>■ Double (r8): Floating point number with double precision</li> </ul>	

Tab	Property	Description
	<ul style="list-style-type: none"> <li>■ Currency (cy): Floating point number that represents currency data types</li> <li>■ Select one of the following data types to represent XML as an integer in Java:           <ul style="list-style-type: none"> <li>• NOM xml node (xml)</li> <li>• DOM Element</li> </ul> </li> </ul> <p><b>Note:</b> When either of the data types is used, the Java class for that WS-AppServer model is generated with the following code, facilitating addition of related custom logic to process XML data. For details, see Extension Class in the <i>AppWorks Platform</i></p>	

Tab	Property	Description
	<p><i>Advanced Development Guide.</i></p> <pre>public static org.w3c.dom.Ele ment DOM (org.w3c.dom.E lement Dom)  {  //  TODO impl emen t body  return null ;  }</pre> <p>To add custom logic to this Java code, you can use the <code>getObjectData()</code> API as shown in the following code snippet:</p> <pre>/* Returns the BusObject Data as a DOM Element  * @return Document DOM Element containing the BusObject data */ public Element getObjectData() {</pre>	

Tab	Property	Description
	<pre>return m_ objectState.getCu rrentObjectDataEl ement(); }</pre> <ul style="list-style-type: none"> <li>■ string: String</li> <li>■ boolean:       <p>Value of either 0 to represent <b>false</b> or 1 to represent <b>true</b></p> </li> <li>■ dateTime:       <p>Date with an optional time data</p> </li> <li>■ WS-       <p>AppServer always works with the GMT/UTC date and time format. It is therefore recommended to follow the same format during application development. For example, in the soap method of implementation type <b>BsfJavaCall</b>, the arguments of type <b>Date</b></p> </li> </ul>	

Tab	Property	Description
	<p>must be handled appropriately.</p> <ul style="list-style-type: none"> <li>■ Base64 encoded BLOB (bin.base64): MIME-style Base64 encoded binary large object (BLOB).</li> <li>■ BusObject: Any data of type <b>BusObject</b></li> <li>■ &lt;classtype&gt;: Existing class in the same package or a different package that can be a data type.</li> </ul>	
Package	<p>Contains the name of the package to which the method is being added. Click  (Lookup record) to browse and select the WS-AppServer Package.</p>	
Occurrence	<p>Determines the occurrence of the method. Valid values are:</p> <ul style="list-style-type: none"> <li>■ 1 for single occurrence</li> </ul>	

Tab	Property	Description
		<ul style="list-style-type: none"> <li>■ * for multiple occurrences</li> </ul>
SOAP		Method can be exposed as a web service.
Cursor		Method can accommodate cursor details.
Static		Static method or an instance method.
Other	Soap Result	Format of the response as a set of tuples or without tuples.
	Transaction	<p>Method participates in an automatic transaction or does not automatically participate in any transaction and the application takes care of starting and completing the transaction. Valid values are:</p> <ul style="list-style-type: none"> <li>■ automatic: WS-AppServer controls the activity of starting and committing the transaction.</li> <li>■ none: Application controls the activity of starting and committing the transaction. When this attribute is left blank (default value), the transaction is considered as automatic and is controlled by the WS-AppServer.</li> </ul>
	Implementation	Implementation type of the method. It can be the default or a signature method.
	Parameters	<p>Parameters of the method.</p> <p><b>Note:</b> A newly created method does not contain any parameter. To make it operational, add parameters.</p>

## Monitoring and crash recovery interface

The following options are available in the Monitoring and Crash Recovery Settings pane to modify monitoring and crash recovery settings:

Field name	Description
Process Name	Selected model. When multiple models are selected, this field is not displayed.
Enable Crash Recovery	<p>Select to enable crash recovery for the published process.</p> <p><b>Note:</b> The crash recovery options set at this level override the options set during design time. You must enable it first in the Process Engine tab of the Business Process Management Service properties. For details see the <i>AppWorks Platform Advanced Development Guide</i>.</p>
Enable Monitoring	<p>Enable or disable monitoring of the published process. If you select this check box, the Monitoring Level list is enabled.</p> <p><b>Note:</b> The monitoring options set at this level override the monitoring options set during design time. You must enable it first in the Process Engine tab of the Business Process Management Service properties. For details see the <i>AppWorks Platform Advanced Development Guide</i>. Options are:</p> <ul style="list-style-type: none"> <li>■ <b>Monitor Level:</b> Enable or disable monitoring of the published process.</li> <li>■ <b>Process status:</b> Basic process information such as status, timestamp, and user information.</li> <li>■ <b>Process input and output messages:</b> Basic process information, input and output messages of the business process.</li> <li>■ <b>Process status, input, output messages and message-map:</b> Basic process information, input and output messages, and message map of the business process.</li> <li>■ <b>Store complete process information when process aborts:</b> Basic process information always. Records input and output messages, and message map of a business process only if the process aborts.</li> <li>■ <b>Store complete process information when process aborts and basic information when the process completes:</b> Basic process information unless process aborts. If process aborts, the complete information related to the input and output messages, and message map of a business process are recorded.</li> <li>■ <b>Store complete process information when process aborts and basic information when the process completes:</b> Basic process information unless process aborts. If process aborts, the complete information related to the input and output messages, and message map of a business process are recorded.</li> <li>■ <b>Store complete process information when one or more exceptions are handled for aborted activities:</b> Complete process information when exceptions are handled.</li> </ul>

# Monitoring user activity in AppWorks Platform

The AppWorks Platform API contains events that are triggered whenever a user logs into or out of the computer. Subscribe to these events to receive updates about users logging into and out of Explorer.

## To monitor user activity:

1. From the Windows Start menu, click **Programs > OpenText AppWorks Platform <version> > <instance name> > Tools > Management Console**.  
The Management Console window opens.
2. On the Management Console window, click **Event Viewer**.  
The Event Viewer window opens and you are subscribed to the event service for monitoring user activity.

# Naming rule actions

Consider the suggestions when naming rule actions.

Rule actions	Suggested rule action name
Trigger Business Process	Name of the process to run.
Send Notification	Name that begins with verbs such as notify, intimate, and alert, followed by the subject of the message to deliver.
Invoke Web Service	Name of the SOAP method to invoke.
Fire Rule	Name of the passive rule to trigger.
Assign	For simple assignment, name in the format SetX where X is the object property being assigned a value.
Abort Transaction	Name that contains a brief description of the condition that causes the action.

# New entity configuration interface

This topic describes the new entity configuration interface.

The following table describes the entity information:

Field name	Description
Entity Name	Optional. Name of the new entity.
Description	Optional. Description of the entity.

The following table describes the contact information:

<b>Field name</b>	<b>Description</b>
Address	Optional. Communication address of the contact.
Description	Optional. Description of the contact.
E-mail	Optional. Email address of the contact.
Name	Optional. Name of the contact.
Phone	Optional. Phone number of the contact.
Use Type	Optional. Designation of the contact.

**Note:** To add new contact information, select  and enter details in the Contact tab created. To delete a contact, select the contact tab and click .

The following table describes the category bag element that enables you to categorize the business entities according to any of the available taxonomy-based classification schemes.

<b>Field name</b>	<b>Description</b>
tModel Key	Optional. Information about common forms of categorizing the business entity.
Key Value	Optional. Identifier within the categorization.
Key Name	Optional. Further describes the key value.

**Note:** To add a new category bag, select  and enter details in the new row created. To delete a category bag, select the check box next to the category bag name and click .

The following table describes the identifier bag that enables business entities to include information about common forms of identification, such as the D-U-N-S number.

<b>Field name</b>	<b>Description</b>
Key Name	Optional. Further describes the key value.
Key Value	Optional. Identifier within the identification.
tModel Key	Optional. Identification.

**Note:** To add a new identifier bag, select  and enter details in the new row created. To delete a category bag, select the check box for the identifier you want to delete and click .

## New service configuration interface

This topic describes the new service configuration interface.

The following table describes the fields in Service Information:

<b>Field name</b>	<b>Description</b>
Description	Optional. Describes the entity.
Service Name	Optional. Name of the new entity.

The identifier bag enables business services to include information about common forms of identification, such as the D-U-N-S number. The following table describes the fields in the Identifier Bag section:

<b>Field name</b>	<b>Description</b>
Key Name	Optional. Further describes the key value.
Key Value	Optional. Identifier within the identification.
tModel Key	Optional. Identification.

**Note:** To add a new identifier bag, select  and enter details in the new row created. To delete a category bag, select the check box for the identifier you want to delete and click .

## New toolbar configuration interface

This topic describes the new toolbar configuration interface.

The following table describes the fields in the new toolbar configuration interface:

<b>Field</b>	<b>Description</b>
caption	Short description of the toolbar.
description	Complete description of the toolbar that appears on the title bar.
name	Unique name of the toolbar.

## Routing Web service requests

When the Web Gateway receives a request from an external client, the request must be passed on to a specific Service Container for execution. This is called routing of Web service requests.

AppWorks Platform consists of different components that work together and use the SOA grid to exchange information through Web services. Understanding how AppWorks Platform routes Web services between the Web Gateway and Service Containers helps in deploying, testing, and monitoring AppWorks Platform with the deployed solutions.

---

## Routing logic

Any message within the SOA grid is a SOAP message. From this SOAP message, only the root element name, operation name, and namespace within the SOAP body are used to route the message to a specific Service Container that can handle the SOAP message. Using a SOA grid client (also known as a bus-client), the Web gateway and the Service Container can send SOAP messages to other Service Containers.

Following are the three levels in routing:

- **Service Group routing**

The target Service Group is identified as follows:

1. The bus-client retrieves the root element name X and namespace Y from the SOAP body of the SOAP message.
2. The bus-client retrieves the Service Groups from LDAP that support Web Service Interfaces with the namespace Y.
  - a. If no Service Group is found in step 2, an error message is returned to the caller.
  - b. If only a single Service Group is found in step 2, routing is continued with Service Container routing.
  - c. If multiple Service Groups are identified in step 2, the bus-client retrieves Service Groups with Web Service Interfaces with the namespace Y and containing a Web Service X. Routing is continued with Service Container routing with the first Service Group that is found.

- **Service Container routing**

After a Service Group that can handle the SOAP message is identified, a Service Container is identified as follows:

1. The bus-client retrieves all Service Containers for the selected Service Group.
2. Based upon the routing algorithm selected on the Service Group, a Service Container is selected.
  - a. In case of Simple Loadbalancing, the SOAP request is routed to the next Service Container in a round robin manner.
  - b. In case of Simple Failover, the SOAP request is routed to the first Service Container on the list that is active and running.
  - c. In case of Classic Loadbalancing, the SOAP request is randomly routed to one of the Service Containers.

The bus-client continues with the Connection Point routing to the specific Service Container.

- **Connection Point routing**

After a Service Container that can handle the SOAP message is identified, a Connection Point is identified as follows:

1. The bus-client selects all the Connection Points for the selected Service Container.
  - a. If there is only a single Connection Point, the SOAP request is sent to that specific Connection Point.
  - b. If there are multiple Connection Points, the following options are available:
    - If the SOAP request requires a reliable connection, a queuing Connection Point is preferred. For example, an activity in a Business Process does not require a response (fire and forget).
    - A direct socket Connection Point is preferred, for example, when an activity in a Business Process requires the output of the Web service before continuing on the execution path.
2. The client waits for a response synchronously (holding on to a thread) or waits for a response asynchronously (giving up the thread).

## **Routing optimization to keep requests inside an OS Process or TomEE**

With the introduction of running Service Containers in TomEE, an optimization is made to the routing logic. To a large extent, the SOA grid maintains the SOAP message and cascading requests within the same OS Process or TomEE (also an OS Process). When the Web gateway or Service Containers that run in TomEE receive a SOAP message, the routing logic identifies the Service Groups and Service Containers within the same TomEE instance to route the SOAP message. If the routing logic cannot identify a Service Group and Service Containers within the same TomEE instance, the request is sent to the matching Service Groups and Service Containers on another Platform node of the cluster.

There is an exception to this rule. If a matching Service Group and Service Container is identified in the same TomEE instance that can handle the

## **Schedule interface fields**

The Schedule interface contains the following fields.

<b>Field</b>	<b>Description</b>
Schedule for every	Frequency to sniff for changes. Select the period, such as Months, Weeks, Days, Hours, or Minutes during which the schedule must be triggered and type the number of times to sniff for changes.
Start time (HH:MM)	Start time to begin the sniffing during the set period. This is applicable for all the period types except Minutes.

<b>Field</b>	<b>Description</b>
Duration HH:MM	Duration of the time during which the sniffing must be performed.
Select days to run the schedule	This is applicable only when the period selected is of the type Weeks or Months. Select the appropriate duration based on the requirement.

## Searching for Web services in Web service Interface Explorer

The Web service Interface Explorer enables you to manage the Web service interfaces and operations in a single interface. To perform any activity such as testing the operations, setting access controls, and publishing to UDDI registry, you must search for a Web service interface or operation using the search parameters provided in this interface.

### Before you begin:

You must have the role of systemAdmin or organizationalAdmin to search for Web services.

### To search for Web services:

- On the Welcome page, click  (Web service Interface Explorer). Alternatively, click  on the toolbar of the System Resource Manager window.  
The Web Service Interface Explorer window opens with a few default search query options.
  - In Keyword, type the relevant term.
- Tip:** To search for content that begins with a specific prefix, select **StartsWith**, and then type the prefix in Keyword. To search for content that ends with a specific suffix, select **EndsWith**, and then type the suffix in Keyword.
- Set the appropriate Search Criteria, and then click **Find**.

<b>Search criteria</b>	<b>Search result displayed in Search Results</b>
Web Service Operations	Web service operations with the keyword.
Web Service Interfaces	Web service interfaces with the keyword.
Service Group	Web service interfaces or operations within the service group.
Namespace	Web service interfaces or operations associated with the namespace. Ensure to specify the exact namespace in Keyword.

<b>Search criteria</b>	<b>Search result displayed in Search Results</b>
	<b>Note:</b> The StartsWith and EndsWith options are not applicable for this criterion.
Application	The Web service interfaces or operations within the application.

## Security settings and options

By default, all the fields on the Code Signing tab of Security Administration are set to Disallow. OpenText recommends retaining the default settings. However, you can modify the default security settings for verifying applications.

The Code Signing tab contains the following fields.

### Security settings

<b>Security setting</b>	<b>Description</b>
Unsigned applications	Applications not signed using a certificate.
Tampered applications	Signed applications that are altered. This includes addition, removal, and update of the files in the application.
Certificate not trusted	Applications signed using certificates that are not in the trust store. The signing certificate and its corresponding certificate chain are not configured in the trust store.
Trusted certificate without valid key usage	Applications signed by a certificate that does not have 'code signing' extended key usage. This certificate might be trusted but is not intended for signing applications.

### Security options

Each setting that can be assigned to any of the fields on the tab is assigned a priority. The options are listed in the order of their decreasing levels of security. This order of priority is chosen while alerting the user of the status of the applications that are being installed. The order of priority for each option is listed in the Security level column.

<b>Security option</b>	<b>Security level</b>	<b>After option selection</b>
Disallow	High	The application installation is stopped.
Prompt	Medium	During application installation, the user is prompted for approval to continue with the installation.
Allow	Low	The application is installed without any prompts.

The application verification status displayed during installation depends on the combination of security settings and the security options associated with them. Consider that the security settings Tampered applications and Certificate not trusted are set to Disallow, Unsigned Applications is set to Allow, and Trusted Certificate without valid key usage is set to Prompt. If the application to install is tampered and also signed by a certificate (not in the trust store), the user is notified that the application is tampered and will not be installed.

## SendMail

This Web service operation sends a mail to the mail server.

### SOAP request

```
<SendMail xmlns="http://schemas.cordys.com/1.0/email">
  <to>
    <address>
      <displayName>Steve Buchanan</displayName>
      <emailAddress>sbuchanan@vanenburg.com</emailAddress>
    </address>
    <address>
      <displayName>Michael Suyama</displayName>
      <emailAddress>msuyama@cordys.com</emailAddress>
    </address>
  </to>
  <cc>
    <address>
      <displayName>Nancy Davolio</displayName>
      <emailAddress>ndavolio@cordys.com</emailAddress>
    </address>
  </cc>
  <bcc>
    <address>
      <displayName>Laura Callahan</displayName>
      <emailAddress>lcallahan@cordys.com</emailAddress>
    </address>
  </bcc>
  <subject><![CDATA[Mail subject]]></subject>
  <body><![CDATA[There is a meeting scheduled for 14:00 hrs today. Make it convenient to attend the same.
Enclosed are some important documents to be discussed at the meeting today.]]>
</body>
<attachments>
  <attachment encoded="true" name="word.doc">
    SGkgZnJvbSBDaGlydXZvbHUGc3VkaGFrYXIu
  </attachment>
  <attachment encoded="false" name="text.txt">
    some plain ASCII content
  </attachment>
</attachments>
<from>
```

```

<displayName>pamela</displayName>
<emailAddress>pworth@cordys.com</emailAddress>
<replyTo>pworth@cordys.com</replyTo>
</from>
<headers>
    <header name="header1">header value1</header>
    <header name="header2">header value2</header>
</headers>
</SendMail>

```

## Request parameters

Parameter	Description
to	Address to send the mail. It contains the address tag.
cc	Addresses to send the mail. It contains the address tag.
bcc	Addresses to send the mail. The recipient details are invisible to other message recipients. Contains the address tag.
address	Address to send the mail. It contains two tags: <ul style="list-style-type: none"> <li>▪ displayName: Name to display in the address</li> <li>▪ emailAddress: Actual email address to send the mail.</li> </ul>
body	Body of the message being sent. It takes type (optional attribute) to use when the text being sent is in HTML format. When this attribute is not specified, the default text type is normal.
attachments	Attachments to send with the message. It contains the attachment tag.
attachment	Attachment file being sent with the mail. It contains two attributes: <ul style="list-style-type: none"> <li>▪ name: Name of the file sent as an attachment</li> <li>▪ encoded: Boolean value that specifies whether the attachment is encoded. If true, the contents of the attachment are Base64 encoded. If false, the attachment is sent as a plain ASCII text file.</li> </ul>
from	Details of the message sender. These details depend on the mail provider. If the <from> tag is not present, the sender details are taken from the email profile stored in LDAP. If both the <from> tag and the email profile are not present, a SOAP error is displayed. If the profile is not yet configured, set the profile using the SetProfile API.
includeHeaders	Option to display the header information of the mail when set to true.

## SOAP response

```
<SendMailResponse xmlns="http://schemas.cordys.com/1.0/email">
  <to>
    <address>
      <displayName>Steve Buchanan</displayName>
      <emailAddress>sbuchanan@vanenburg.com</emailAddress>
    </address>
    <address>
      <displayName>Michael Suyama</displayName>
      <emailAddress>msuyama@cordys.com</emailAddress>
    </address>
  </to>
  <cc>
    <address>
      <displayName>Nancy Davolio</displayName>
      <emailAddress>ndavolio@cordys.com</emailAddress>
    </address>
  </cc>
  <bcc>
    <address>
      <displayName>Laura Callahan</displayName>
      <emailAddress>lcallahan@cordys.com</emailAddress>
    </address>
  </bcc>
  <subject><![CDATA[Mail subject]]></subject>
  <body type="html"><![CDATA[There is a meeting scheduled for 14:00 hrs today.  
Make it convenient to attend the same.  
Enclosed are some important documents to be discussed at the meeting  
today.]]>
  </body>
  <attachments>
    <attachment encoded="true" name="word.doc">
      SGkgZnJvbSBDaGlydXZvbHUGc3VkaGFrYXIu
    </attachment>
    <attachment encoded="false" name="text.txt">
      some plain ASCII content
    </attachment>
  </attachments>
  <from>
    <displayName>pamela</displayName>
    <emailAddress>pworth@cordys.com</emailAddress>
    <replyTo>pworth@cordys.com</replyTo>
  </from>
</SendMailResponse>
```

## Sequence Generator Properties interface

The Sequence Generator Properties interface contains the following fields.

Field	Description
Seed Value	Initial integer that begins the sequence. Type the seed value, for example, 100.
Increment	Integer by which the sequence increases. Type the integer value. For example, if you specified Seed Value as 100 and Increment as 1, the next sequence is Seed Value + Increment, that is, $100 + 1 = 101$ .

The Sequence Creator interface contains the following fields.

Field	Description
Type	<p>Types of registry key fragments.</p> <p>Click  (Add) under Sequence Creator to define a registry key fragment.</p> <p>A fragment (row) appears with a list under the Type column with the default value, Source Key. You can select a combination of more than one type of registry key fragments, which results in a unique value.</p> <p>Type has the following options:</p> <ul style="list-style-type: none"> <li>■ Source key: Select to replace the fragment with the primary key of the incoming record.</li> <li>■ Static token: Select to specify a set of characters as a token to the fragment.</li> <li>■ Next sequence value: Select to retrieve the fragment value depending on the specified Seed Value and Increment.</li> </ul> <p><b>Note:</b> Click  (Delete) to delete the fragment. Select the check box corresponding to the fragment, and then click  to move the fragment up or click  to move the fragment down.</p>
Value	<p>Value of the registry key fragment type. You can specify values only for the Static token registry key fragment. The values for the Source Key and Next sequence value are generated automatically during runtime.</p>

### Example for specifying field names to generate sequence registry entries

To integrate data from applications, for example, Spoke1 and Spoke2 to the hub Hub 1, specify Seed Value as 100 and Increment as 1. The next sequence generated is Seed Value + Increment, that is,  $100 + 1 = 101$ .

If you select Source Key type as the first registry key fragment, select Static token as the second registry key fragment, specify its value as HUB REGISTRY, and then select Next sequence value as the third registry key fragment.

The entries in the registry table are generated as follows: HUB REGISTRY 101, HUB REGISTRY 102, and HUB REGISTRY 103.

## Task example

The Application Registry task supports the concept of a task. It helps in performing the following activities, which are classified into task parts and related tasks. The Application Registry task is configured to different roles allowing specific activities based on the role.

Activity	Task type	Roles that can perform the activity
Display application details	Task part	<ul style="list-style-type: none"><li>■ System administrator</li><li>■ Organizational administrator</li></ul>
Display installation details	Task part	<ul style="list-style-type: none"><li>■ System administrator</li><li>■ Organizational administrator</li><li>■ Developer</li></ul>
Display application content	Task part	<ul style="list-style-type: none"><li>■ System administrator</li><li>■ Organizational administrator</li></ul>
Load applications	Related task	<ul style="list-style-type: none"><li>■ System administrator</li></ul>
Unload applications	Related task	<ul style="list-style-type: none"><li>■ System administrator</li><li>■ Organizational administrator</li></ul>
Upgrade applications	Related task	<ul style="list-style-type: none"><li>■ System administrator</li><li>■ Organizational administrator</li></ul>

## tModel Details interface

The tModel Details interface contains the following sections and fields.

### tModel information

The tModel Information section contains the following fields.

Field	Description
tModel Name	Optional. Name of the tModel.
Description	Optional. Description of the tModel.

## Category bag

The Category Bag element categorizes tModels according to any of the available taxonomy-based classifications schemes.

Field	Description
Key Name	Optional. Description of the key value.
Key Value	Optional. Identifier within the tModel categorization.
tModel Key	Optional. Information about common forms of categorizing the tModel.

## Identifier bag

The Identifier bag allows tModels to include information about common forms of identification such as D-U-N-S number.

Field	Description
Key Name	Optional. Description of the key value.
Key Value	Optional. Identifier within the tModel identification.
tModel Key	Optional. Identification of the tModel.

# Universal Discovery Description Integration (UDDI)

Universal discovery description integration (UDDI) is a core Web service that represents a set of protocols and a public directory for registration and real-time look up of Web services and other business processes. It is an XML and SOAP based specification or protocol, and performs the following functions:

- Defines the standard for communication by a client or a business to publish their Web services to a directory service called the UDDI Business Registry (UBR).
- Defines APIs that can be used by other software applications.
- Contains APIs that can replicate the business registry.

UDDI provides a framework that enables business entities to describe and classify their services, and provide the technical details about the interfaces of their Web services. It also allows organizations to search for and locate the required Web services.

## UDDI Explorer interface

The UDDI Explorer interface contains the following fields.

Field	Description
Inquiry URL	Mandatory. URL to connect to the details of the UBR. Click  (Browse) to display the Master UBR List window, and then click <b>Inquiry URL</b> to create an inquiry URL.
Find	Mandatory. Business entities or business services depending on what to search.
Name	Mandatory. Name to search. By default, the UBR is searched for entered values such as keys.

## UDDI Explorer tool

The UDDI Explorer tool allows users to browse through any UDDI business registry.

You can use the UDDI Explorer to search the UDDI Business Registry (UBR) for business entities based on the following search criteria:

- Name
- Categories
- Identifiers

You can use the UDDI Explorer to search the UBR for business services based on the following search criteria:

- Name
- Business key
- Categories

The UDDI Explorer also allows adding a UBR and searching for content dynamically.

## Unconditional ACL

In an unconditional ACL, the ACL is evaluated based on the object metadata. The prepared request is sent to the ACL engine for evaluation based on which, the data is retrieved and updated.

ACL can be either open or blocked. All permissions are open by default.

See the following sample ACL object tree, which blocks the `GetEmployee` Web service operation in the `Northwind` Web service interface in the `Northwind` service group.

```
<object id="cn=NorthwindService,cn=soapnodes,o=system,cn=cordys,o=vanenburg.com">
    <object id="http://schemas.cordys.com/1.0/demo/northwind">
        <object acl="blocked" id="GetEmployee"/>
    </object>
</object>
```

When specifying ACL settings for an object, the `cn=object` of the object is specified. However, in the case of a Web service interface, the namespace of the Web service interface is specified. For example, to set ACL for the Northwind Web service interface, the `http://schemas.cordys.com/1.0/demo/northwind` namespace is used in the object settings instead of the name of the Web service interface.

For anonymous users to access database related methods, ACLs must be set at the database level or the table level. See the following sample to set database level permissions for an object.

```
object id="cn=NorthwindService,cn=soap nodes,o=system,cn=cordys,o=vanenburg.com">
<object id="testdb">
    <object id="testtablename">
        <method id="read" acl="open" />
    </object>
</object>
<object id="http://schemas.cordys.com/testdb" acl="open">
    <object id="GettesttablenameObject" acl="open" />
</object>
</object>
//Where testdb is the database name and testtablename is the name of the table in
the database//
```

ACL settings at the metadata services level can be set at the object level or at the Web service operation level. When set at the object level, the hierarchy of permissions are open or blocked. Every child object inherits the settings from its parent object unless specified otherwise.

See the following sample ACL setting for a parent object and its child.

```
<object acl="blocked" id="parentObject">
    <object id="childObject"/>
    <object acl="open" id="childObject2"/>
</object>
```

In the above code sample, the permission for `childObject` is set to blocked because nothing is specified on this child object. However, for `childObject2`, the permission is set to open.

When set at the Web service operation level, the permissions are hierarchical in the following order:

- read
- update
- insert
- delete

See the following sample to set Web service operation level permissions for an object.

```
<object id="parentObject">
    <object id="childObject1">
        <method acl="open" id="delete"/>
    </object>
    <object id="childObject2">
        <method acl="blocked" id="insert"/>
    </object>
</object>
```

The above object tree indicates that the ACL permissions are open on `childObject1` because `delete` is open. For `childObject2`, the `insert` permissions are blocked, and as per the hierarchy, the `insert` and `delete` permissions are blocked.

When read permission is blocked for a user, the user is completely blocked from accessing the object. ACL settings are strictly hierarchical in nature and therefore, when multiple Web service operation settings are set for an object, the object with the highest priority (topmost hierarchy) is considered as the setting.

**Note:** When an update operation is tried on a blocked field, it results in an empty tag for that field, implying that it is not possible to update that field.

ACL settings can be defined for system administrators, who are authorized to administer the entire system, and organizational administrators, who administer an organization in AppWorks Platform.

You can define roles in an organization and at the system level. Organizational roles are defined in an organization and can be assigned to the users in an organization. A user in an organization must be assigned at least one role. The roles can have subroles and the subroles can again contain roles under them. You can specify ACL settings on organizational roles that belong to a particular organization.

## Updating the WS-AppServer package with database details

This topic describes the procedure to update a WS-AppServer package with database content.

### Before you begin:

Create a Database Metadata or Reload the existing Database Metadata before updating the package, to ensure that you get the latest database sources into the package.

**Note:** See Creating a Database Metadata in the *AppWorks Platform Advanced Development Guide* and [Reloading the database metadata](#) for details.

To make the WS-AppServer package functional, load it with database sources so that you can generate the Java code and perform Web service operations on them.

### To update the package:

1. Double click the  (WS-AppServer Package) from either of the following locations:

- Workspace Documents (My Recent Documents) window.
- Workspace Documents (Explorer) > <solution> > <project> .

The WS-AppServer Package window opens, displaying the data models and their corresponding attributes, methods, and relations.

2. On the **WS-AppServer Package** toolbar, click  (Load from Database).  
The Update Package from Database Metadata dialog box opens.
3. Select **Table**, or **View** or both, to view the database contents for the selection.  
The database objects are listed in rows.
4. Click the individual check boxes to select a specific Table or View, or click the database object check box to select all.

**Tip:** Enter the name of the Table or View in the Filter field to narrow the search and display matching results, thereby enabling quicker selection.

5. Click **Update**.

**Note:** If you update the WS-AppServer package with the same database tables or views that currently exist in the package, all the existing models are displayed with an option to overwrite the tables. If you have not selected any existing table for the update, the package is directly updated with the newly selected tables, without listing any existing model, or without showing the option to overwrite. Depending upon the requirement, perform the necessary action.

The WS-AppServer Package is loaded with the selected database models and displays the details of each as Attributes, Methods, and Relations.

For each model, it displays a:

- **Mapped Name:** The name that corresponds to the actual database table name
- **Class Namespace:** The name of the Package Namespace.

You can edit the default names.

## Upload and download properties interface

Use the upload or download properties interface to configure the properties of the upload or download functionality.

The following table describes the fields in the properties interface of an entity:

Field name	Description
Sync With Spokes	Select this check box to synchronize data from the hub with all the subscribed spokes, after uploading the data.

<b>Field name</b>	<b>Description</b>
Enable State Flow	When a state engine is defined on a hub entity, select this check box in the property sheet of Upload, to ensure that the data to be uploaded first goes through the state flow before being updated in the hub.
Use Lookup	Select this check box to register conflicting data, instead of overwriting it in the Upload function.
Threads	Upload data in batches. Names of the thread specified in the Web service.
Parameter	Input parameter names to specify the range of values in the batch.
Xpath	Xpath of the input parameter in the request. For example, if / is displayed, it indicates that it is the immediate child of the request.
Value	Specify the range values. For example, specify the range values such as range1: 0-100, and range2: 101-200.
Use Filter	Select this check box to filter specific records based on a rule in the rule filter window.

## Using a digital certificate

The various aspects within AppWorks Platform where digital certificates are used are:

<b>Feature</b>	<b>Description</b>	<b>Store</b>
OpenText CARS	Self-signed certificate to secure LDAP connection. This certificate is used only when OpenText CARS is being used in the SSL mode (ldaps://).	<OpenText CARS_installdir>/certificates/<MACHINE_NAME>-cert.cer. See <a href="#">SSL options on OpenText CARS</a> for more information.
Application Signer	Certificates used for signing	This certificate must be provided while signing. See <a href="#">Application signing</a> for more information.

<b>Feature</b>	<b>Description</b>	<b>Store</b>
	applications to ensure integrity of the application.	
UDDI Connector	Certificates used for server authentication (SSL/TLS).	Trust anchors are stored in the Certificate store in the Security Administration task. Backwards compatibility: Trust anchors can be stored in Java keystore provided with <code>uddi.keystore</code> property.
UDDI Connector	Certificates used for client authentication (SSL/TLS).	See <a href="#">Using SSL in Platform connectors</a> for more information.
HTTP Connector	Certificates used for server authentication (SSL/TLS).	Trust anchors are stored in the Certificate store in the Security Administration task.
HTTP Connector	Certificates used for client authentication (SSL/TLS).	See <a href="#">Using SSL in Platform connectors</a> for more information.
Enterprise Service Bus (ESB)	Certificates used for signing SOAP messages. These messages are exchanged between service groups to ensure secure transfer.	Java keystore. See <a href="#">Enabling SSL communication</a> for more information.
Trusted publisher certificate	Certificate-issued guarantee application integrity. Used for signing application packages released by software vendors such as OpenText.	Code Signing Certificate store in Security Administration task.
AppWorks Platform Monitor Certificate	Every installation has a self-signed certificate called monitor	<code>&lt;AppWorks_Platform_installdir&gt;\certificates\keystore\&lt;MACHINE_NAME&gt;_monitor.p12</code>

Feature	Description	Store
	certificate, that is unique for every installation. It acts like a Certification Authority (CA) in AppWorks Platform.	
SSO	Monitor certificate used for signing SAML assertions issued by AppWorks Platform.	

## Viewing access permissions granted for a role

This topic describes the procedure to view ACLs that are granted for a role.

### Before you begin:

Required role: System Admin

In CWS , a developer can set access controls on Web service interfaces, operations and tables of Database Metadata. You can view those access controls through the User Manager.

1. On the Welcome page, click  (User Manager).  
The User Manager window opens.
2. Select Roles - Roles, Roles - Users, or Roles - Tasks view in the User Manager window.  
See [Roles - Roles view](#), [Roles - Users view](#), and [Roles - Tasks view](#) for details.  
The selected view opens.
3. In the Roles pane, right click a role and select **Security**.  
The ACL Explorer dialog box opens, displaying the existing access controls assigned for that role.
4. Click an access control.  
The Access Control Set - <Name> pane opens, displaying the details. See the ACL Explorer Interface in the *AppWorks Platform Advanced Development Guide* for details.

## Viewing log messages at service container level

1. On the Welcome page, click  (System Resource Manager).  
The System Resource Manager window opens and lists the available service containers.

- Note:** Alternatively, you can click  to view the existing service containers.
2. Right-click <Service Container> and click the **View Log** option.  
The **Log Viewer - <Service Container>** dialog box opens displaying the log messages. See [Log viewer interface for a service container](#).
- The event details are published to the file specified during log configuration. In the absence of log configuration, the log messages are read from the file called `ProcessNamedDailyRollingFileAppender`, which is located at `<AppWorks Platform_installdir>\Logs`. The naming convention for this depends on the process to which the log messages are written.

For instance:

- Service container - `<Organization Name>#<Service Group DN>#<SOAP Processor DN>.xml` For example, `system#xml_store_service#xml_store_processor.xml` (spaces are replaced with an underscore).
- OS process - `<OS Process Name>.xml`  
For example, `MyOSProcess.xml` is created for an OS process named `MyOSProcess`.
- Monitor - `Monitor.xml` and by default, this is available in the `<AppWorks Platform_installdir>\Logs` directory.
- Gateway - It is `Gateway.xml` and this is available in the `<AppWorks Platform_installdir>\Logs` directory.

If log messages to be published do not fall in any of the four categories mentioned above, they are published to `General.xml` in the `<AppWorks Platform_installdir>\Logs` directory.

## Viewing log messages using Apache Chainsaw

The Apache's Chainsaw is an open source GUI-based tool used to view the log messages.

1. In the Provide Log Configuration Details screen, in the Logger Consumers frame, do the following:
  - a. Select the **Publish to Remote Host** option.
  - b. In **Host Name**, type the machine number.
  - c. Type the port number as 4445. This is the default port for Chainsaw. For further information on changing the configuration, see the Chainsaw documentation.
2. Run the following command from the command prompt:

```
java org.apache.log4j.chainsaw.Main
```

**Note:** Apache Chainsaw cannot open files which have the hash (#) character in the filename. AppWorks Platform log files have this character. Rename it to some other character to view the files in Chainsaw.

## Web gateway statistics interface

Table 1. Web Gateway statistics

Field	Description
Number of requests	The total number of requests passed through the gateway.
Total processing time	The total time taken for processing the requests.
Average processing time per request	The average time taken to process a single request.
Total back end wait time	Time delay at the back end while waiting for a response.
Invalid messages	Total number of invalid messages passed through the gateway.
Unknown Web users	Total number of unknown users accessing the gateway.
Invalid organization requests	Total number of invalid organization requests to the gateway.
Messages sent to Server	Total number of messages sent to the gateway.
Response from Server	Total number of replies received from the gateway.
Timed out requests	Total number of timed out requests (replies not received due to time out).
LDAP errors	Total number of LDAP errors encountered.
Errors	Total number of errors occurred.

## Web service operation interface

A Web service operation interface is described in the Web Services Description Language (WSDL) format .

WSDL contains the description about the following:

- Request and response objects
- Parameters passed for the request and their data types
- Response type
- Parameters of the response and their data types

WSDL contains the information about the input and output of the Web service operations in LDAP. Any browser-based application can use such information to query information or inputs that must be supplied to the back end.

WSDL is stored as the bus method signature parameter of the LDAP Web service operation and has the following structure:

```
<definitions name="method name" targetNamespace="namespace"
  xmlns="http://schemas.xmlsoap.org/wsdl/" xmlns:tns="namespace">
  <types/>
  .....
  <message/>
  .....
  <portType/>
  .....
  <binding/>
  .....
  <service/>
  .....
</definitions>
```

In the structure:

- types - contains definitions of XML schemas
- message - contains descriptions of documents that comply with XSDs from the types
- portType - contains the description of a Web service operation signature
- binding - defines message format and protocol details for operations and messages.
- service - represents a service, which groups related ports to provide some functionality.

The definition contains the attributes such as `name` (Web service operation name), `targetNamespace` (labeledURI of the Web service interface). Each `portType` contains an operation. This is a Web service operation that is specifically defined (for example, `GetEmployees`). The `portType` structure is defined as follows:

```
<portType name="Method Set Name">
  <operation name="MethodName">
    <input message="Methodrequestreference"/>
    <output message="Methodresponserference"/>
  </operation>
</portType>
```

Each operation within the `portType` has input and output. These inputs and outputs are messages and must be defined in one of the `<message>` elements. As a result, the contents of these messages are defined in the schemas that go under the `<types>` element.

The message contains a part. These are the request and the response names that the `<input/>` and the `<output/>` tags are referencing in the operation. The `<message/>` structure is as follows:

```
<message name="message request/response name">
<part name="body" element="request/response name">
</message>
```

The `types` tag contains the schema definition and the element definitions of the request and response involved in the Web service operation.

```
<types>
  <schema targetNamespace="namespace" xmlns="http://www.w3.org/2001/XMLSchema">
    <element name="request">
      <complexType>....</complexType>
    </element>
    <element name="response">
      <complexType>....</complexType>
    </element>
  </schema>
</types>
```

The elements for request and response contain the definitions for the input and output parameters. The parameters can be of a `complexType` (that contains subnodes and data under them) or a `simpleType` (that is self descriptive and does not expand). The structure for the `simpleType` and `complexType` are as follows:

```
<element name="complex-element">
  <complexType>
    <all>
      <element name="simple-element">
        <simpleType>
          <restriction base="type definition">
            <length value="length"/>
          </restriction>
        </simpleType>
      </element>
    </all>
  </complexType>
</element>
```

XSDs can be created as an LDAP Web service operation of object class `busmethodtype` in addition to `busmethod`. Front end or back end tools can use these XSDs to display data and parameters for the return type. An XSD has the following structure:

```
<schema xmlns="http://www.w3.org/2001/XMLSchema">
  <complexType>
    <all>
      <element/>
      ...
      ...
    </all>
  </complexType>
</schema>
```

```

    </all>
  </complexType>
</schema>

```

In this structure:

- schema - contains the XML schema definition of a business object
- element - contains the element definitions involved in the business object return type

The `<element>` definition can have the following structures:

```

<element>
  <complexType>
    <all>
      <element>
        <simpleType>
          ....
        </simpleType>
      </element>
    </all>
  </complexType>
</element>

```

In this structure, there can be two types of elements:

- Complex Type - Items that can contain children nodes under them and are expandable. These can have more elements under them.
- SimpleType - Items that cannot contain child nodes under them (last level of an element).

## Web service properties interface

The Web service Properties Interface helps in configuring properties of a Web service. Select a relevant Web service to create, read, update and delete data. These Web services are used during data synchronization and while performing bulk operations like upload.

Table 1. Fields on the Web Service Properties Interface

Field Name	Description
<b>Cursor</b>	The number of records to be fetched for data integration. When you click the <b>Read</b> Web service under <b>Web Services</b> , by default, <b>1000</b> is displayed in the text box under <b>Cursor</b> . You may change the number of records to be fetched depending upon the requirement.

Field Name	Description
	<p><b>Note:</b> The <b>Cursor</b> along with <b>1000</b> is applicable for only <b>Read</b> and <b>Update</b> Web services Properties panes, when you select the <b>Use Generic Update</b> check box.</p>
<b>Read Thread Parameters &gt; Parameter</b>	<p>When you select a <b>Read</b> Web service, the WSDL is read and its input parameters are displayed. The input parameter name to specify the range of values in the batch is displayed under the <b>Parameter</b> column.</p>
<b>Read Thread Parameters &gt; Threads</b>	<p>Threads facilitate to upload data in batches. <b>Thread1</b> appears by default, under the <b>Threads</b> column on selecting the Web Service. Click <b>+</b> (<b>Insert</b>) to add more threads.</p> <p><b>Note:</b> The Read Thread Parameters along with Threads, Parameter, XPath and Value columns are applicable only for the Read Web Service Properties pane.</p>
<b>Read Thread Parameters &gt; Value</b>	<p>Specify the range values. For example, specify the range values as range1: 0-100, range2: 101-200 and so on. If the range is very large, for performance improvement, you can add more than one thread. The threads run in parallel to improve performance.</p>
<b>Read Thread Parameters &gt; XPath</b>	<p>The XPath of the input parameter in the request is displayed. For example, if '/' is displayed, it indicates that it is the immediate child of the request.</p>
<b>Use Generic Update</b> check box	<ul style="list-style-type: none"> <li>■ Select the <b>Use Generic Update</b> check box to perform Insert, Update and Delete operations using a single Web service. For example, <b>Update</b> and <b>UpdateBusinessObject</b> Web services generated in AppWorks Platform over database tables.</li> <li>■ Clear the check box to select 3 distinct Web services for Insert, Update and Delete. The <b>Read</b>, <b>Insert</b>, <b>Update</b> and <b>Delete</b> Web services are displayed under</li> </ul>

Field Name	Description
	<p><b>Web Services</b> on clearing the check box.</p> <p><b>Note:</b> By default, the <b>Use Generic Update</b> check box is selected, when you select the <b>Database Table</b> option. This check box is disabled, when you select the <b>Others (External Web Service, etc)</b> type of business object.</p>
<b>Web Service</b>	<p>Select a Web service read data.</p> <p><b>To select a Web service:</b></p> <ol style="list-style-type: none"> <li>1. Click  under <b>Web Service</b>. The <b>Select a Webservice Operation</b> dialog box opens.</li> <li>2. Click the relevant Web service to select it. The selected Web service is displayed in the text box under the <b>Selected Document</b> and the <b>OK</b> button is enabled.</li> <li>3. Click <b>OK</b>. The selected Web service is displayed in the text box under <b>Web Service</b>.</li> </ol> <p>The Web services is selected.</p> <p><b>Note:</b></p> <ol style="list-style-type: none"> <li>1. Click  to clear the selected Web service.</li> <li>2. The <b>Web Service</b> is applicable for <b>Read, Insert, Update, and Delete</b> Web services Properties panes.</li> </ol>

## Working with State SyncUp

State SyncUp supports distributed applications. When more than one instance of the same application runs on a single computer or on different computers, synchronization among these instances is essential. AppWorks Platform State SyncUp synchronizes the data across instances and provides it to the user as if it were available locally.

State SyncUp maintains the system state, the state of service containers and application connectors, to use this information by the clients to route requests intelligently. For example, if a new instance of a service is added to a group, this information is provided by the State SyncUp to all the clients, who can start sending requests to this newly added service. This balances the overall load on the system, thereby, making it scalable.

---

Similarly, when the service is down, either crashed or is in the maintenance phase, clients can use this information to re-route the messages to the other available service instances. This is referred to as high availability. See [Overview of state SyncUp](#) and [State Syncup properties](#).

This section covers the following topics:

- State SyncUp Architecture - Explains the architecture of State SyncUp. See [State SyncUp architecture](#).
- Components of a Default Cluster - Describes the various SyncUp types and differentiates between them. See [Components of a default cluster](#).
- Configuring State SyncUp - Describes the procedure to configure State SyncUp. See [Modifying the SyncUp configuration of monitor service group](#).
- Synchronizing Service Containers - Describes the procedure to synchronize service containers. See [Synchronizing service containers](#).

## Overview of state SyncUp

This topic provides an overview about the members involved in a ring.

### ***Communication between the members of the default ring***

- The AppWorks Platform (<instance name>)s form the ring.
- The Master SyncUp (the SyncUp instance within the monitor) receives and sends messages and acts as a member of the ring.
- The communication between the members happen using TCP.

### ***Role of the default ring***

- LDAP cache invalidation and service container state such as start, stop, or in problem are communicated using the ring.
- AppWorks Platform provides a smart routing mechanism based on SyncUp. Using this mechanism, a client can identify the service containers that are down, and accordingly, route messages to another service container. This information is available locally with the client and such a routing decision is made instantly.

### ***Role of the master SyncUp***

- The master SyncUp instance in the monitor communicates with the SyncUp in the service containers. The SyncUp in the service container is known as the local SyncUp. The communication between the monitor (master SyncUp) and service container (local SyncUp) occurs through the TCP protocol and is one-to-one. The local SyncUp instances communicate with the master SyncUp on the same machine.
- The master SyncUp at the monitor updates the state registry with the content for the individual service container on that machine, using the local SyncUp instance.

- Broadcast messages are sent to the broadcast module and state registry messages are sent to all associated state registries, at master SyncUp and local SyncUp, over TCP/IP.
- Each AppWorks Platform (<instance name>) may again have multiple remote clients that communicate from another machine. These constitute the remote SyncUp instances. The communication between the monitor and remote clients occurs through the TCP protocol and is one-to-one.
- Remote SyncUp follows both Broadcasting and State Registry communication types. However, it cannot modify the content of the State Registry.
- State Registry messages (Stateful messages) are dispatched to the local state registry, and to state registries attached to it, through the local and remote syncUp instances.
- Broadcast messages (stateless messages) are published to people who subscribe to them.

## State SyncUp architecture

The AppWorks Platform State SyncUp concept is based on the group membership protocol.

- All members in the group can exchange information with each another.
- Message transfer among members can take place only when the member of the group (monitor) has the token.
- The token is forwarded to the other group member through this ring.

AppWorks Platform provides a default ring that consists of all AppWorks Platform monitors in the network.

## State SyncUp features

State SyncUp facilitates scalability and high availability for AppWorks Platform, with the following features:

- State SyncUp defines a logical ring of application instances.
- Reliable communication among the SyncUp instances within the ring can be carried out in either of the following ways:

Mechanism	Features
Broadcast Mechanism (Stateless Transfer or Publish-Subscribe mechanism)	<ul style="list-style-type: none"> <li>■ The developer can register an object (application) for a specific topic.</li> <li>■ The registered object receives messages that are published on the topic.</li> <li>■ Such an object can publish messages for the topic.</li> </ul>
State Registry	<ul style="list-style-type: none"> <li>■ All the members in the group have access to the common data in XML format.</li> </ul>

Mechanism	Features
	<ul style="list-style-type: none"> <li>■ A member can perform certain operations on the data such as read, insert, delete, or update.</li> <li>■ Listeners can be registered for a particular content so that changes to the XML are notified to the listener.</li> </ul>

## State Syncup properties

The following table describes the properties for State Syncup.

Property name	Description
com.eibus.transport.groupmembership .udp.networkipaddress	This property sets the IP address for the monitor on the current system for multicast. Set this property when AppWorks Platform runs behind a VPN connection or outside the network. The State Syncup feature provided by the monitor Service Container synchronizes the data with other monitors. A multicast IP is used for broadcasting the data to the other monitors. When the system is out of the network, it cannot connect to the IP. This property has to be set to enable OpenText AppWorks Platform (<instance name>) outside the network.
com.eibus.transport.middleware.routing	This property sets the default routing class. It defaults to com.eibus.transport.routing.ClassicRouting . This routing class is used only when the configuration of the Service Container does not contain the routing algorithm. This implies, if you are sending a message to an anonymous connection point, it will ignore the StateSyncUp information.
ldap.multicast.timeout	This property sets the timeout for LDAP multicasting.

**Note:** By default, multicast is no longer used. When multicast is enabled it only is used in the bootstrap phase.

## Components of a default cluster

SyncUp instances are of three types: master, local, or remote. These are described in the following table.

A SyncUp instance type is determined based on the protocol.

<b>Master SyncUp</b>	<b>Local SyncUp</b>	<b>Remote SyncUp</b>
OpenText AppWorks Platform (<instance name>)s constitute the members of the cluster.	Each instance of local SyncUp is associated with an instance of master SyncUp on that machine.	An instance of remote SyncUp is arbitrarily associated with the instance of master SyncUp, on a different machine.
The messages between these instances are communicated over TCP.	Communication between the master SyncUp and local SyncUp is one-to-one, and is through TCP.	Communication between the master SyncUp and remote SyncUp is one-to-one and is through the TCP protocol.
Each instance in the cluster has a unique identifier.	The unique identifier of master SyncUp is used by the local SyncUp.	-
Instances can either use state registry or broadcast communication types, to communicate with each other.	Instances can either use state registry or broadcast communication types to communicate with each other.	A remote SyncUp instance can use only the stateless transfer of data mechanism to send messages across the cluster.
Each instance can either insert, update or delete only a designated part of the state registry (shared data in XML).	Each instance can either insert, update or delete only a designated part of the state registry (shared data in XML).	Remote SyncUp can read the information within the state registry but cannot modify its contents.
Each instance of the master SyncUp can subscribe to any topic or publish message on any topic.	Each instance can subscribe to any topic or publish message on any topic.	Each instance can subscribe to any topic or publish message on any topic.

## Modifying the SyncUp configuration of monitor service group

By default, the Monitor Service Group is enabled to participate in the SyncUp with default settings.

### To change the SyncUp configuration:

1. Access the **Properties App Palette** of Monitor Service Group Properties.
2. In the **State Syncup Details** section, provide the following details.
  - Machine Name or IP Address- Type the Machine name or the IP address. This IP synchronizes the SyncUp instances among the monitors. This value can be changed and subsequently when the page is accessed, the new IP address is updated on all the monitors in the ring.
  - Port Number - Type the port number

- 
3. Click  on the toolbar.

Your machine is configured to use a SyncUp instance.

Alternatively, you can configure State SyncUp while creating a new Monitor Service Group.

## Synchronizing service containers

While developing distributed applications, data across the service containers needs to be synchronized so that all the members of a group have up-to-date information. Do this by forming a ring and sharing the same machine name or IP address address and port number.

### To synchronize service containers:

1. On the Welcome page, click **Synchronize Service Containers**.  
The Choose Target dialog box opens. By default, the first component in the list of components is selected and the target related to that component is displayed.
2. If you want to synchronize the target that is displayed, click **OK** in the Choose Target dialog box, and proceed to step 5 of this task.  
Click **Cancel** to select a different target. The Synchronize Service Containers page opens, displaying the list of components along with the machine name or IP address and port number in the Machine Name\IP address and Port Number text boxes.
3. Select the required service (for example, CoBOC Service or Rule Service) from the list of components.  
The Choose Target dialog box opens.
4. Click the required service and then click **OK**.  
A list of DNs of the service container that embeds CoBOC or Rule Engine, machine name or IP address, and the port number of the selected service is displayed in the respective SyncUp Identifiers, Machine Name\IP address, and Port Number text boxes.
5. Select the required SyncUp identifier from the SyncUp Identifier list box.
6. Make the necessary changes to the Machine Name\IP address and Port Number and click **Save**.

**Note:** The Save button is enabled only when you modify any of the content. For the changes to take effect on the instances configured to the corresponding database, restart each service container.

The service containers are now synchronized and share or update the data when the data is updated in any of the service containers . To synchronize all the service containers of a particular service group, repeat steps 5 and 6.

### **Changing CoBOC service container SyncUp configuration**

When more than one instance of the same application runs on a single machine or on different machines, synchronization among these instances is essential. AppWorks Platform State SyncUp synchronizes data across instances and provides it to the user, as if it were available locally. A reliable multicast protocol for IP provides reliability and atomicity to application programs. Such a protocol is useful for developing distributed applications,

---

particularly when it must be ensured that all members of the group receive the same information.

1. On the Welcome page, click  (Synchronize Service Containers).  
The Choose Target dialog box opens.
2. Click  CoBOC to select the rule management service that is configured to the database for which you want to change the SyncUp configuration and click **OK**.  
The **Synchronize Service Containers** window opens, displaying **List of Components** and the related Service Properties pane with the Multicast IP and port number in the Multicast IP and Port Number text boxes respectively, by default.
3. Make the necessary changes in Multicast IP and Port Number and click  on the toolbar.

For the changes to take effect on the instances configured to the corresponding database, restart each CoBOC service.

The syncup configuration of the CoBOC service container is changed and all the processors sharing the same Multicast IP address will be in sync now.

For the changes to take effect on the instances configured to the corresponding database, restart each CoBOC service container.

### ***Changing rule SyncUp configuration***

When more than one instance of the same application runs on a single machine or on different machines, synchronization among these instances is essential. AppWorks Platform State SyncUp synchronizes data across instances and provides it to the user, as if it were available locally. A reliable multicast protocol for IP provides reliability and atomicity to application programs. Such a protocol is useful for developing distributed applications, particularly when it must be ensured that all members of the group receive the same information.

1. On the Welcome page, click  (Synchronize Service Containers).  
The Choose Target dialog box opens.
2. Click  Rule Management to select the rule management service that is configured to the database for which you want to change the SyncUp configuration and click **OK**.  
The Synchronize Service Containers window opens, displaying List of Components and the related Service Properties pane with the Multicast IP and port number in the Multicast IP and Port Number text boxes respectively, by default.
3. Make the necessary changes in the Multicast IP and Port Number text boxes and click  on the toolbar.

For the changes to take effect on the instances configured to the corresponding database, restart each Rule Management service.

The syncup configuration of the Rule Management service container is changed and all the service containers sharing the same Multicast IP address will be in sync now.

For the changes to take effect on the instances configured to the corresponding database, restart each Rule Management service container.