



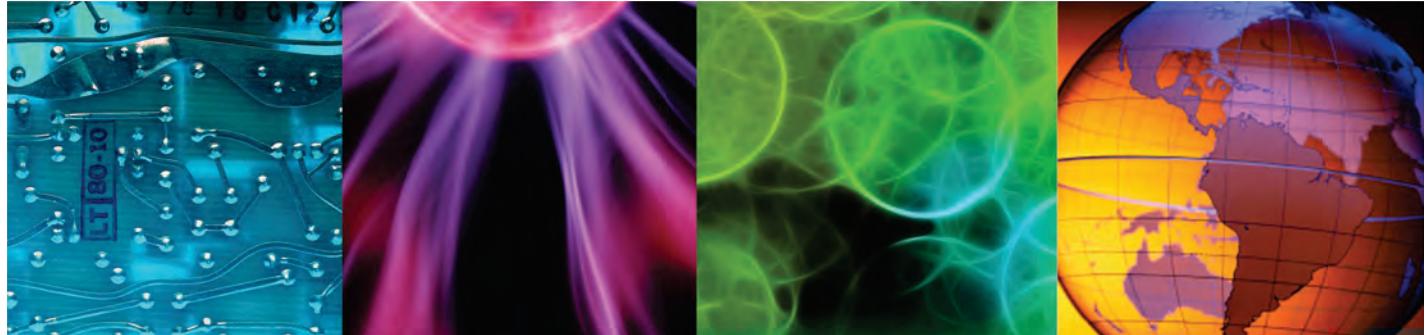
IBM Training

IBM Tivoli Workload Scheduler for z/OS 9.2 Scheduling and Operations

Student Notebook

Course code TM405 ERC 1.0

October 2015



All files and material for this course are IBM copyright property covered by the following copyright notice.

© Copyright IBM Corp. 2015. All Rights Reserved.

US Government Users Restricted Rights: Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

IBM, the IBM logo, and ibm.com are trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at "Copyright and trademark information" at www.ibm.com/legal/copytrade.shtml.

Trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.

IT Infrastructure Library is a registered trademark of the Central Computer and Telecommunications Agency which is now part of the Office of Government Commerce.

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

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

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

ITIL is a registered trademark, and a registered community trademark of the Office of Government Commerce, and is registered in the U.S. Patent and Trademark Office.

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

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

Cell Broadband Engine is a trademark of Sony Computer Entertainment, Inc. in the United States, other countries, or both and is used under license therefrom.

Linear Tape-Open, LTO, the LTO Logo, Ultrium, and the Ultrium logo are trademarks of HP, IBM Corp. and Quantum in the U.S. and other countries.

The information contained in this publication is provided for informational purposes only. While efforts were made to verify the completeness and accuracy of the information contained in this publication, it is provided AS IS without warranty of any kind, express or implied. In addition, this information is based on IBM's current product plans and strategy, which are subject to change by IBM without notice. IBM shall not be responsible for any damages arising out of the use of, or otherwise related to, this publication or any other materials. Nothing contained in this publication is intended to, nor shall have the effect of, creating any warranties or representations from IBM or its suppliers or licensors, or altering the terms and conditions of the applicable license agreement governing the use of IBM software.

References in this publication to IBM products, programs, or services do not imply that they will be available in all countries in which IBM operates. Product release dates and/or capabilities referenced in this presentation may change at any time at IBM's sole discretion based on market opportunities or other factors, and are not intended to be a commitment to future product or feature availability in any way. Nothing contained in these materials is intended to, nor shall have the effect of, stating or implying that any activities undertaken by you will result in any specific sales, revenue growth, savings or other results.



Contents

About this course	ix
About the student	x
Learning objectives	x
Course agenda	xi
1 Introduction.....	1
Objectives	2
Lesson 1 IBM Tivoli Workload Automation suite	3
Tivoli Workload Automation products	5
Lesson 2 Architecture	6
z/OS -only architecture	7
Fault-tolerant capable end-to-end architecture	8
z-centric end-to-end architecture	10
Tivoli Workload Scheduler Distributed agent	11
Tivoli Workload Scheduler for z/OS address spaces: Sample configuration	12
Lesson 3 Concepts and terminology	13
Workstations	14
Long-term plan and current plan	16
Scheduling objects	17
Applications	19
Workload Service Assurance (critical path management)	22
Lesson 4 Version 9.1.0 differences overview	24
Run cycle groups	26
Application Description dependency criteria	27
Lesson 5 Version 9.2.0 differences overview	28
Version 9.2.0 differences overview (continued 1)	31
Version 9.2.0 DWC and mobile device differences	33
Version 9.2.0 DWC and mobile device differences (continued)	35
Version 9.2.0 DWC and mobile device differences (continued)	37
Review questions	38
Review answers	39
Summary	40
2 Workstations.....	41
Objectives	42
Lesson 1 Workstation overview	43
Workstation types	45
Workstation reporting attributes	46
Workstation options	47

Computer, remote, and general workstations	49
Lesson 2 Creating workstation definitions	52
Parallel servers	55
Workstation activity	56
Workstation ready list	57
Student exercises	59
Summary	60
3 Calendars, periods, and run cycle groups.....	61
Objectives	62
Lesson 1 Introduction to calendars	63
Calendar usage	65
Creating Calendars	66
Lesson 2 Periods	67
When work is scheduled to run	69
Examples of periods	70
Using period offsets	72
Lesson 3 Run cycle groups	74
Specifying run cycle group list criteria	75
Creating a run cycle group	76
Run cycle group fields	76
Defining rules in run cycle groups	77
Run cycle group rule types and sub-sets	78
Free day rules 1,2,3, and 4	79
Free day rule E	80
Specifying rules	81
Using the Every option	82
Using the GENDAYAS command	83
Student exercises	86
Summary	87
4 Applications	88
Objectives	89
Lesson 1 Applications and operations	90
Application structure	92
Application groups	93
Lesson 2 Creating applications	95
General information for applications	96
Creating an Application in ISPF	97
Lesson 3 Timing workloads	98
Run command panel	99
Defining run cycles	100
Specifying run cycle rules	102
Using the Every option	103
Using the GENDAYAS command	104
Creating unique application occurrences	105
Lesson 4 Defining operations	106
Entering operations information	107
Operations panel	109

GRAPH command example display	110
More on predecessor dependencies	111
Operations details panel	113
Predecessors panel	114
Dependency resolution panel	115
Conditional dependency components	118
Conditions list panel	119
Condition dependency definition panel	120
Condition related status codes	122
Condition dependency example	123
Specifying other operation details	124
Operation automatic options	126
Considerations for operations	129
Operation-related data sets	130
Lesson 5 Job descriptions	132
Job description example	133
Student exercises	136
Summary	137
5 Operation submission, throughput, and monitoring	138
Objectives	139
Lesson 1 Operation submission and throughput	140
Planning the operation run order	141
Operation throughput management options	143
Lesson 2 Dynamic feedback overview	144
Dynamic feedback flow: duration feedback	145
Dynamic feedback initialization parameters	146
Operation feedback limit and smoothing factor	148
Feedback and smoothing formula terminology	150
Operation duration feedback example	151
Smoothing factor example	152
Getting started capturing operation durations	153
Lesson 3 Operation priority	154
Lesson 4 Critical operations	155
Defining critical operations	156
Running the critical path	158
Monitoring the critical path	159
Additional critical path considerations	160
Running the critical path reports	161
Critical path report example	162
Lesson 5 Workload Manager scheduling environments	163
Supported environments	164
Workload Manager scheduling environment availability	165
Single sysplex example	166
Setting up a Workload Manager scheduling environment	167
Using the SCHENV JCL keyword	169
Monitoring scheduling environment operations	170
Lesson 6 Advanced ISPF panels	171

Advanced ISPF panels example	172
Advanced panels graph view	173
Student exercises	174
Summary	175
6 Long-term and current plans.....	176
Objectives	177
Lesson 1 Data bases and plans overview	178
Long-term plan status	179
General CP information	180
Lesson 2 Long-term planning overview	182
LTP build process	184
Maintaining the long term plan panel	185
Contents of the long-term plan	187
Long-term plan extension	188
LTP dependencies	190
Lesson 3 The current plan	192
Daily planning batch functions	194
The daily planning process	195
Current plan batch job parameters	196
Creating and extending the current plan.....	198
Student exercises	199
Summary	200
7 Restart and cleanup	201
Objectives	202
Lesson 1 Introduction to restart and cleanup	203
Restart and cleanup overview	204
Joblog retrieval data sets	205
Starting restart and cleanup	206
Lesson 2 Restart and cleanup options	208
The RC row command	210
Operation restart and cleanup panel	211
Restart and cleanup operations detail panel	213
Data set selection eligibility	216
When cleanup actions are initiated	217
Requesting notification of cleanup	218
Exceptions to immediate cleanup actions	219
Lesson 3 Restarting the operation	220
Selecting step restart	221
Step restart selection list panel	223
Modifying cleanup actions panel	224
Editing JCL from restart	226
Confirm restart panel	227
Cleanup results	228
Lesson 4 Browsing the job log	229
Job log view (1 of 2)	230
Job log view (2 of 2)	231

Student exercises	232
Summary	233
8 Special resources.....	235
Objectives	236
Lesson 1 Special Resource Overview	237
Special resource characteristics	238
Special resource database properties	239
Lesson 2 Creating special resources	241
Special resource ON COMPLETE example	243
Special resources MAX USAGE example	245
Creating resources dynamically	246
SRSTAT LIFESPAN parameter	248
Special resource LIFESPAN example	249
Lesson 3 Using special resources in operations	250
Specifying operation resource requirements	251
Assigning a special resource to operations	252
Special resource monitoring	253
Special resource monitor: Waiting queue	254
Modifying a special resource	255
Student exercises	256
Summary	257
9 Automated job tailoring.....	258
Objectives	259
Lesson 1 Automated job tailoring overview	260
JCL directives overview	262
JCL variables overview	263
Lesson 2 JCL directives	264
Identifying directive statements	266
Dynamic inclusion and exclusion	267
The FETCH directive	268
Begin and End directives	269
Data manipulation directives	270
The SETFORM directive	271
The SETVAR directive Syntax	273
Search-related directives	275
Searching for variables examples	276
Lesson 3 JCL Variables	277
Using variables in a job	279
Variable concatenation	281
Using compound variables	282
Using tabular variables	283
Operator-prompted variables	284
Tivoli Workload Scheduler for z/OS supplied variables	285
User-created variable tables	286
Defining variables in tables	287
Assigning variable tables to applications	288

Scanning order for variable resolution	289
Summary	290
10 Automatic recovery	291
Objectives	292
Lesson 1 Automatic recovery	293
Automatic recovery diagram	295
Implementing automatic job recovery	296
Recovery actions	297
Recovery statements: conditional criteria	298
Recovery statements: JCL rebuild and actions	299
Automatic job recovery example JCL	300
The ARC row command	301
Automatic recovery messages and comments	302
Logging and error reporting	303
Summary	304
11 Managing unplanned work	305
Objectives	306
Lesson 1 Managing unplanned work overview	307
Lesson 2 OPSTAT and SRSTAT commands	308
The OPSTAT command	309
Issuing the OPSTAT command	310
OPSTAT example	311
The SRSTAT command	312
Issuing the SRSTAT command	313
SRSTAT example	314
Lesson 3 Event-triggered tracking	315
Event types	316
ETT occurrence-related JCL variables	317
ETT variable substitution overview	319
ETT event type J: Sample JCL job	320
Specifying ETT criteria	321
Special resource event triggers	323
Job name event trigger	324
Lesson 4 Data set triggering	325
Creating a data set triggering table	327
Data set triggering table: Sample entries	329
Loading the data set triggering table	330
Data set triggering life span values	331
Lesson 5 XML, an alternative method	332
Student exercises	334
Instructor demonstration	335
Summary	336

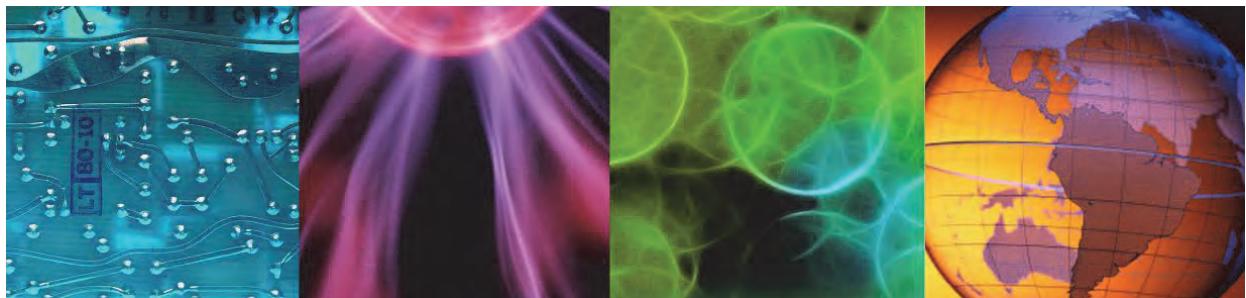


About this course

IBM Tivoli Workload Scheduler for z/OS version 9 release 2 modification 0



IBM Tivoli Workload Scheduler for z/OS 9.2.0 Scheduling and Operations



© Copyright IBM Corporation 2015

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

In this 4-day instructor-led course, you learn how to manage batch workloads in z/OS with IBM Tivoli Workload Scheduler for z/OS. During the course, you use ISPF panels to create scheduling definitions and perform operational tasks. There is also a brief overview of the end-to-end environment.

The course covers creating and controlling planning objects, such as workstations, applications, calendars, and resources. You learn how to schedule these objects into a daily planning cycle and then monitor and manage them.

For information about other related courses, visit the education training paths website:

ibm.com/software/software/tivoli/education/

Details	
Delivery method	Classroom or instructor-led online (ILO)
Course level	ERC 1.0
This course is an update of TM404 ERC 1.0	

Details	
Product and version	BM Tivoli Workload Scheduler for z/OS 9.2
Duration	4 days
Skill level	Intermediate

About the student

This course is designed for schedulers and operators who will schedule and manage jobs in a z/OS environment using IBM Tivoli Workload Scheduler for z/OS 9.2.0.

Before taking this course, make sure that you have the following skills:

- A basic understanding of the z/OS operating system
- The ability to move around in ISPF dialogs
- Basic JCL

Learning objectives

Objectives

After completing this course, you should be able to perform the following tasks:

- Describe the components of Tivoli Workload Automation
- Define the different types of workstation
- Create scheduling objects: calendars, periods, and run cycle groups
- Build and schedule Application Descriptions containing multiple operations
- Managing critical batch
- Create the Long Term and Current plans and manage the batch to completion
- Implement restart and cleanup to simplify the rerun and restart of failed jobs
- Create special resources to further manage and control batch jobs
- Automate JCL editing with JCL variables and directives
- Code automatic recovery statements in JCL to restart failed work
- Take control of unscheduled batch

Course agenda

The course contains the following units:

1. [Introduction](#)

This unit reviews the architecture, components, and main features of Tivoli Workload Scheduler for z/OS. It also describes the main feature differences available in version 9.1.0 and version 9.2.0.

There are no exercises for this unit.

2. [Workstations](#)

In this unit, you learn how Tivoli Workload Scheduler for z/OS workstations function, and how to create such workstations.

3. [Calendars, periods, and run cycle groups](#)

In this unit, you learn about Tivoli Workload Scheduler for z/OS calendars and calendar periods. You learn to describe calendar and period functions, and how to create them.

4. [Applications](#)

In this unit, you learn how to create and schedule Tivoli Workload Scheduler for z/OS applications, application groups, and job descriptions.

These exercises give you the opportunity to create a Tivoli Workload Scheduler for z/OS application group, an application with multiple operations, and two job descriptions by using the scheduler panels. These exercises are the first of several where you create Tivoli Workload Scheduler for z/OS applications. You schedule and run these two applications during the [Unit 6, “Long-term and current plans exercises” on page 42](#).

5. [Operation submission, throughput, and monitoring](#)

In this unit, you learn when and how applications and operations are selected for submission in Tivoli Workload Scheduler for z/OS. The unit also covers how operations are managed to help maximize throughput by using several Tivoli Workload Scheduler for z/OS options.

6. [Long-term and current plans](#)

In this unit, you learn the relationship between the Tivoli Workload Scheduler for z/OS plans, and how to create and maintain the plans.

In these exercises, you use the Tivoli Workload Scheduler for z/OS panels to review the Long-Term and current plans and work with and monitor a scheduled unit of work. You work with an application in the current plan (CP) and edit job statements in operations in the application. You also correct failed operations and monitor the application to a successful completion.

7. [Restart and cleanup](#)

In this unit, you learn how to use the IBM Tivoli Workload Scheduler for z/OS restart and cleanup functions.

In this exercise, you use the Tivoli Workload Scheduler for z/OS panels to define the cleanup types for operations in the application database. You then restart failed operations at job level and step level with different cleanup attributes.

8. [Special resources](#)

This unit describes the implementation and use of special resources.

9. [Automated job tailoring](#)

In this unit, you learn about automated job tailoring, to automatically change jobs by using Tivoli Workload Scheduler for z/OS Job Control Language (JCL) variables and directives.

There are no exercises for this unit.

10. [Automatic recovery](#)

In this unit you learn about the Tivoli Workload Scheduler for z/OS automatic recovery function and how to code the control statement for it.

There are no exercises for this unit.

11. [Managing unplanned work](#)

In this unit, you learn about TSO-authorized Tivoli Workload Scheduler for z/OS commands.

You also learn about event-triggered tracking and the data set triggering functions. You can use TSO commands, event-triggered tracking, and data set triggering to handle unplanned work.

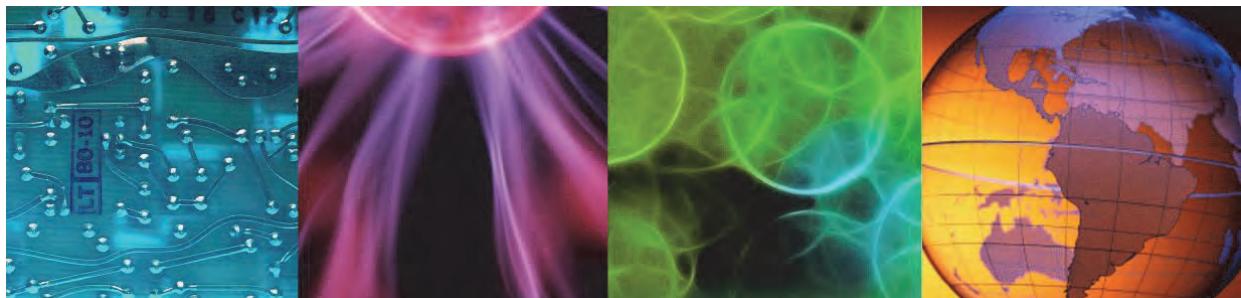
The exercises for this unit combine what you learned in Units 9, 10, and 11. You create an application with operations that use automatic recovery, automated job tailoring, and event-triggered tracking. You issue IBM Tivoli Workload Scheduler for z/OS SRSTAT and OPSTAT commands from a batch job and TSO.



Tivoli Workload Scheduler for z/OS 9.2.0

1 Introduction

1 Introduction



© Copyright IBM Corporation 2015

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

This unit reviews the architecture, components, and main features of Tivoli Workload Scheduler for z/OS. It also describes the main feature differences available in version 9.1.0 and version 9.2.0.

References

- SC32-1264 IBM Tivoli Workload Scheduler for z/OS Planning and Installation
- GC27-4088-01 IBM Tivoli Workload Scheduler for z/OS 9.2.0 Memo to Users
- GC27-4088-00 IBM Tivoli Workload Scheduler for z/OS 9.1.0 Memo to Users

Objectives

- Describe the IBM Tivoli Workload Automation suite
- Describe these properties of IBM Tivoli Workload Scheduler for z/OS:
 - Structure
 - Major functions and components
 - Relationship to single-host and multiple-host systems
- Describe the feature differences provided in IBM Tivoli Workload Scheduler for z/OS 9.1.0 and 9.2.0

© Copyright IBM Corporation 2015

Objectives

The Tivoli Workload Automation Family of Scheduler products is a mature collection of products with components that can be used to manage work on many platforms. This unit looks at all the possible configurations that can be deployed. After completing this unit, you can describe Tivoli Workload Automation and its workload management functions, and provide introductory information about its products:

- IBM Tivoli Workload Scheduler for z/OS
- IBM Tivoli Workload Scheduler
- IBM Tivoli Workload Scheduler for Applications
- Tivoli Dynamic Workload Console

Furthermore, you can describe the following items:

- The structure of IBM Tivoli Workload Scheduler for z/OS
- Where it fits in single-host and multiple-host systems
- The major functions of Tivoli Workload Scheduler for z/OS

Tivoli Workload Scheduler for z/OS, version 9.2.0 was used to develop this course, however, feature differences and details that are related to version 9.1.0 are pointed out and emphasized during the course.

Lesson 1 IBM Tivoli Workload Automation suite

Lesson 1 IBM Tivoli Workload Automation suite

- IBM Tivoli Workload Scheduler for z/OS
- IBM Tivoli Workload Scheduler
- IBM Tivoli Workload Scheduler for Applications
- Tivoli Dynamic Workload Console

© Copyright IBM Corporation 2015

In this lesson, you review the product suite that is associated with Tivoli Workload Automation. After completing this lesson, you should be able to list the products available in the Tivoli Automation product suite.

References: SC32-1264 *IBM Tivoli Workload Scheduler for z/OS Planning and Installation*

The IBM Tivoli Workload Automation suite consists primarily of four components:

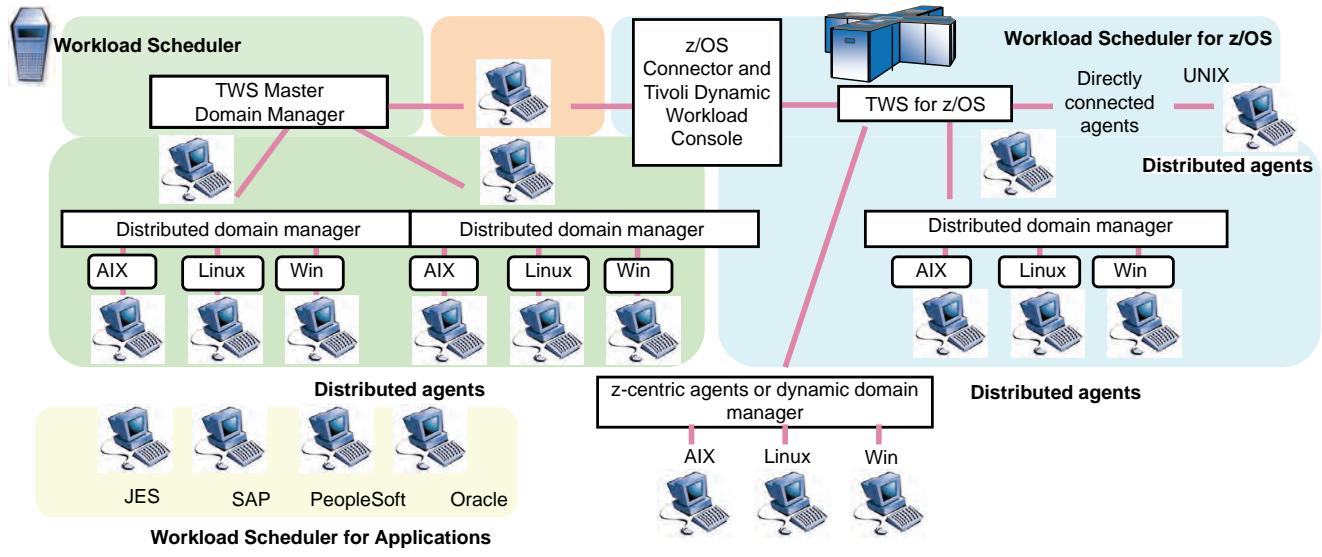
- **IBM Tivoli Workload Scheduler for z/OS** is the Tivoli Workload Automation solution for mainframe-resident workloads that run primarily on z/OS. Tivoli Workload Scheduler for z/OS also extends to other platforms such as Linux, UNIX, and Windows (LUW).
- **IBM Tivoli Workload Scheduler**, primarily for workloads that are on Linux, UNIX and Windows platforms, offer a single console, real-time alerts, dynamic resource-aware workload brokering, reporting, and self-healing capabilities.
- **IBM Tivoli Workload Scheduler for Applications** extends automation features to Oracle e-business Suite, PeopleSoft®, SAP R/3, and SAP Business Warehouse. Tivoli Workload Scheduler for Applications is SAP Solution Manager Ready certified.
- The **Tivoli Dynamic Workload Console** is a component of IBM Tivoli Workload Scheduler for z/OS and IBM Tivoli Workload Scheduler. Tivoli Dynamic Workload Console is a light, single point of control for the scheduling network. Tivoli Dynamic Workload Console provides single

1 Introduction

Lesson 1 IBM Tivoli Workload Automation suite

sign-on and authentication to one or many schedulers, is scalable and provides real-time monitoring, management, and reporting of enterprise workloads.

Tivoli Workload Automation products



© Copyright IBM Corporation 2015

Tivoli Workload Automation products

This slide depicts the Tivoli Workload Automation products and components in a single architecture. Many of the components, such as the distributed agents and the Tivoli Dynamic Workload Console are used in both Tivoli Workload Scheduler and Tivoli Workload Scheduler for z/OS installations.

Lesson 2 Architecture

Lesson 2 Architecture

- z/OS only architecture
 - Can have cross dependencies with other z/OS or distributed engines
- End-to-end architecture
 - z-centric controller based
 - Classic fault-tolerant capable server based

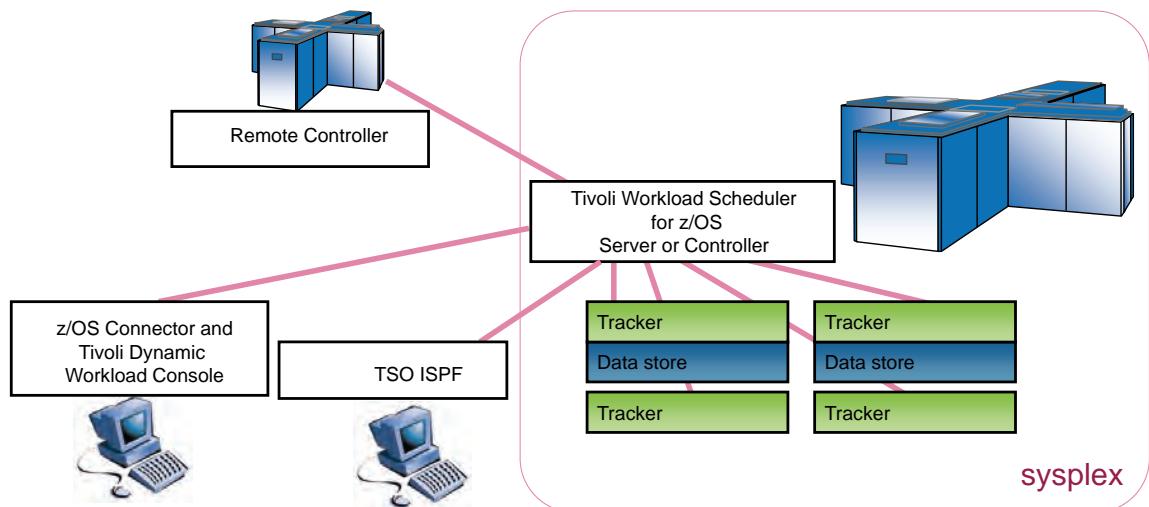
© Copyright IBM Corporation 2015

This lesson discusses several different architectures that are possible in a Tivoli Workload Scheduler for z/OS environment. The purpose is to help provide the student with the knowledge of how to implement the various features and components of Tivoli Workload Automation with z/OS being the central system.

After completing this lesson, you should be able to describe various architectures and implementations possible with Tivoli Workload Scheduler for z/OS.

References: SC32-1264 *IBM Tivoli Workload Scheduler for z/OS Planning and Installation*

z/OS-only architecture



© Copyright IBM Corporation 2015

z/OS -only architecture

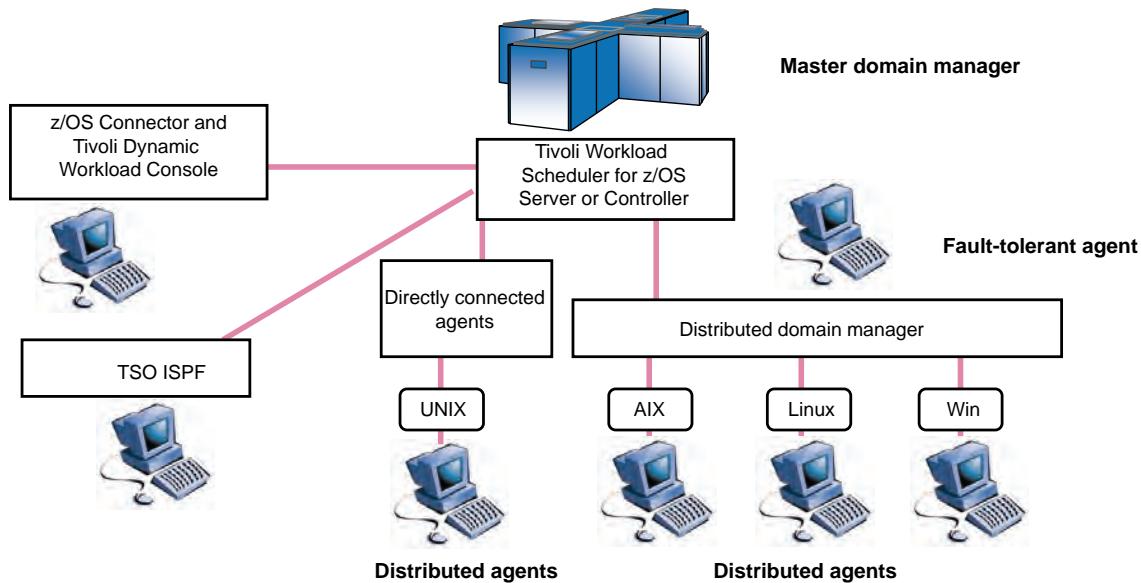
In the z/OS-only architecture, all work occurs on the mainframe, within z/OS. The base component in the Tivoli Workload Scheduler for z/OS system is the **tracker**. A tracker handles the submission of jobs and tasks onto the system. It monitors the workloads using standard interfaces with Job Entry System (JES) and system management facilities (SMF). The tracker records relevant information about the workload and generates event records. The event records are communicated to the controller for processing.

Optionally, a **data store** is installed for each JES spool in a system. Its function is collecting information about steps and data sets of jobs. The Restart & Cleanup functions (restart at job or step level, data set cleanup, job log retrieval) uses the collected information. In a simple JES configuration, each tracker has one data store. In systems with shared spools, each spool can have one data store.

The **controller** is the heart of the Tivoli Workload Scheduler for z/OS system. It contains all the controlling functions, user panels, databases, and plans. Each production environment has at least one controller. You can install more controllers, for example to provide hot standby capabilities or for business reasons, but only one controller is required. The controller communicates with all the trackers for managing the entire scheduling system. TSO (Time Sharing Option) and ISPF provide the primary interface for scheduling and monitoring jobs.

The **Tivoli Dynamic Workload Console** is the web-based user interface for the entire Tivoli Workload Automation suite of products. The Console is the strategic user interface for the suite. It includes support for the latest functions and enhancements that are featured in the scheduler. It uses a z/OS connector feature to connect to the z/OS system. The z/OS connector is automatically installed with version 9.1.0 and later of the Dynamic Workload Console.

Fault-tolerant capable end-to-end architecture



© Copyright IBM Corporation 2015

Fault-tolerant capable end-to-end architecture

The **Tivoli Workload Scheduler for z/OS Connector** accompanies the product and provides connectivity between the Tivoli Dynamic Workload Console and the controller. One connector can communicate with multiple Tivoli Workload Scheduler for z/OS controllers, and can serve multiple servers that run the Tivoli Dynamic Workload Console. One server can have both the Tivoli Dynamic Workload Console and connector installed. Tivoli Dynamic Workload Console and Tivoli Workload Scheduler for z/OS connector are components that you install and run on Linux, UNIX, and Windows (LUW) platforms.

Specify **remote engine workstations** if you want dependencies that are associated with batch activities managed by another Tivoli Workload Scheduler environment. This type of dependency is called a **cross-dependency**. The diagram shows the remote engine as a z/OS controller, but it might also be a Tivoli Workload Scheduler distributed Master Domain Manager (MDM).

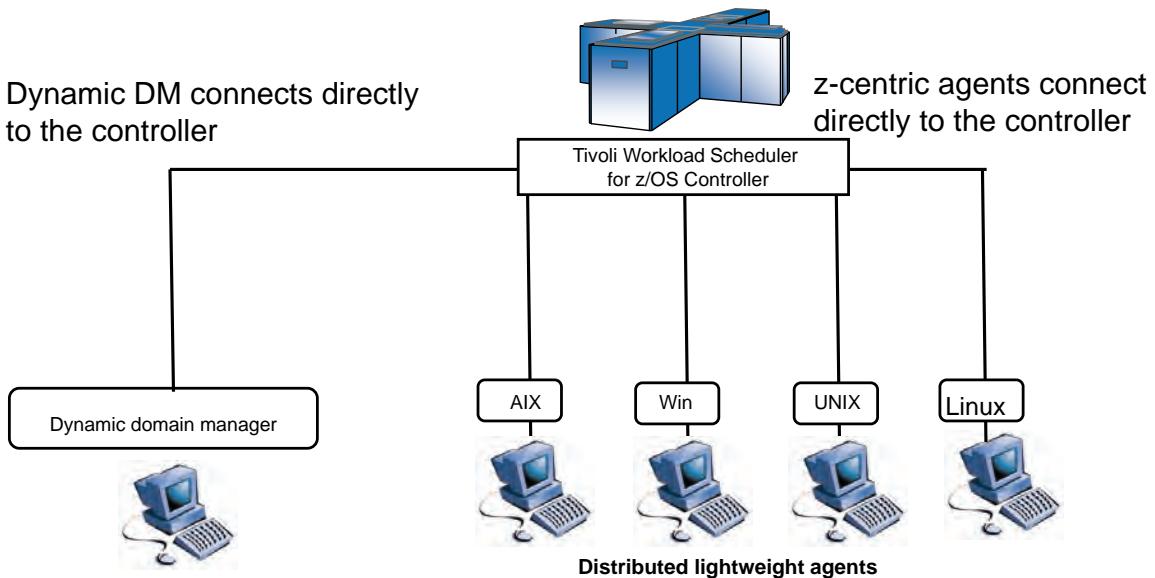
IBM Tivoli Workload Scheduler for z/OS can also drive workloads on other platforms, in two end-to-end configurations, classic fault-tolerant capable end-to-end, and zCentric end-to-end.

In the classic end-to-end architecture, the Tivoli Workload Scheduler for z/OS controller connects to Tivoli Workload Scheduler fault-tolerant agents (FTAs). You can manage workloads on LUW and iOS from either ISPF panels or through the Tivoli Dynamic Workload Console. End-to-end configurations provide extra choices for designing workloads that span physical and organizational boundaries.

In the end-to-end with fault tolerance architecture, you can have an unlimited number of linked Tivoli Workload Scheduler agents and subdomains. By passing each agent a copy of the daily

workload, each agent can independently manage its own work. It can manage the work even if it encounters a system or network outage and automatically recover workloads after the outage ends.

z-centric end-to-end architecture



© Copyright IBM Corporation 2015

z-centric end-to-end architecture

The newest Tivoli Workload Scheduler for z/OS topology option is the zCentric end-to-end. In this architecture, a lightweight distributed agent is deployed in the LUW environment. The Tivoli Workload Scheduler for z/OS controller connects directly to the agent when it detects that a job is ready to run there. The controller then sends the job information to that agent.

The Controller can assign jobs to the workstation that best meets both the hardware and software requirements, by using a Dynamic Domain Manager (DDM). The zCentric agents are assigned to fixed or dynamic pools in the DDM. The jobs are assigned to the workstation that represents the DDM and the DDM decides which agent/server best suits the needs of the job.

Refer to the *IBM Tivoli Workload Scheduler: Planning and Installation Guide* for a detailed explanation on how to install a dynamic domain manager for a z/OS controller. Product documentation can be found at:

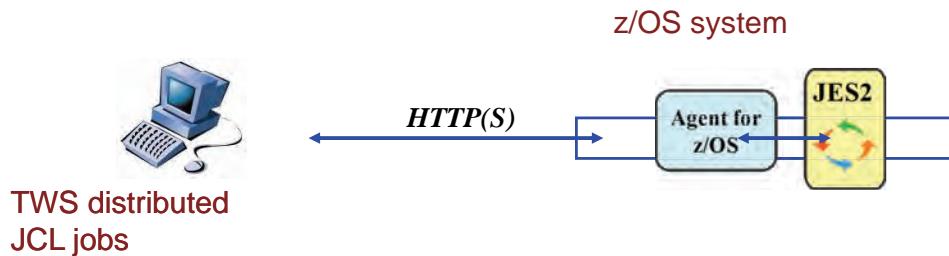
<http://publib.boulder.ibm.com/infocenter/tivihelp/v3r1/index.jsp?toc=/com.ibm.tivoli.itwszos.doc/toc.xml>



Note: A single enterprise can have combinations of all of the listed architectures. Tivoli Workload Scheduler for z/OS offers the flexibility to adapt the deployment to the specific needs of the business.

Tivoli Workload Scheduler Distributed z/OS agent

- Installs on z/OS
- Run JCL jobs from a Tivoli Workload Scheduler distributed engine
 - Define, submit, and monitor JCL jobs inside a Tivoli Workload Scheduler distributed plan
 - Get the JES job log of the completed JCL jobs in the Tivoli Workload Scheduler distributed user interface

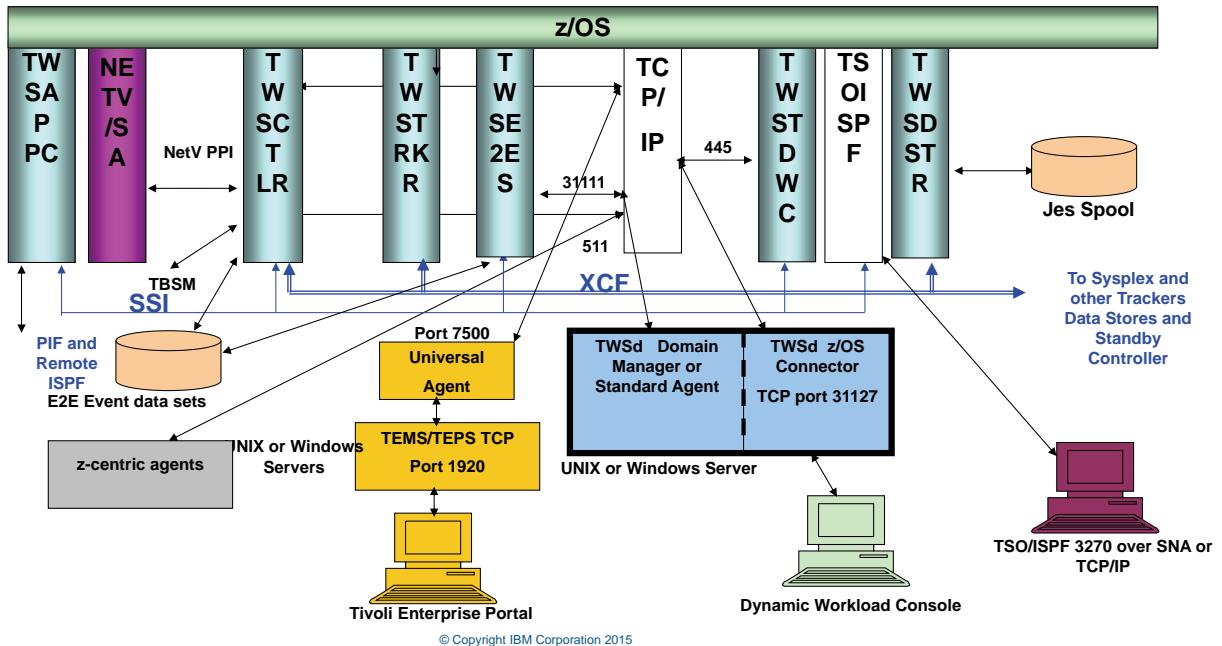


© Copyright IBM Corporation 2015

Tivoli Workload Scheduler Distributed agent

This agent is based on z/OS and runs on z/OS. Tivoli Workload Scheduler distributed controls and manages it. With this agent, the Tivoli Workload Scheduler distributed environment can run JCL jobs on z/OS.

Tivoli Workload Scheduler for z/OS address spaces: Sample configuration



Tivoli Workload Scheduler for z/OS address spaces: Sample configuration

This slide shows a possible address space configuration for the Tivoli Workload Scheduler for z/OS. In this example, all three architectures are implemented. This slide also shows the use of the Tivoli Enterprise Portal as a monitor.

Lesson 3 Concepts and terminology

Lesson 3 Concepts and terminology

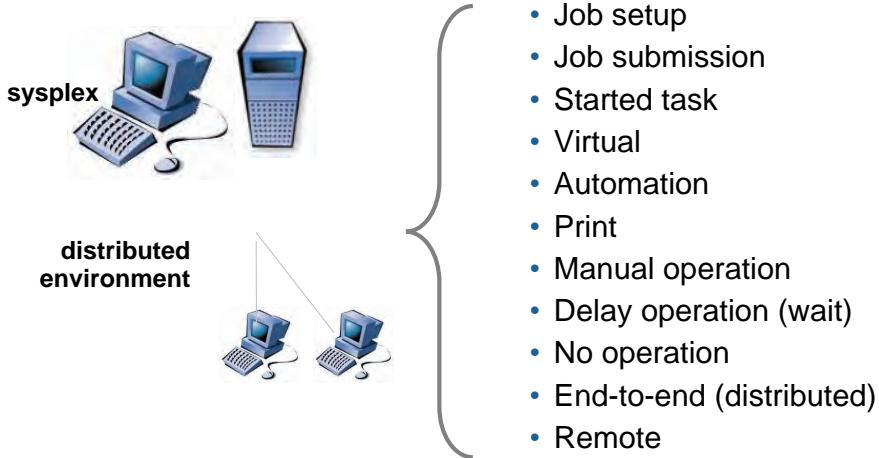
- Workstations and virtual workstations
- Long term, current, and trial plans
- Applications
- Dependencies
- Special resources
- Calendars
- Run cycles

© Copyright IBM Corporation 2015

In this lesson, you learn some of the more common concepts and terminology that is used with Tivoli Workload Scheduler for z/OS. After completing this lesson, you should be able to describe many of the key terms and objects that are used in Tivoli Workload Scheduler for z/OS.

References: SC32-1263 IBM Tivoli Workload Scheduler for z/OS Managing the Workload

Workstations



Workstations

When scheduling and processing work, Tivoli Workload Scheduler for z/OS considers the processing requirements of each job. Some typical processing considerations are as follows:

- Human or computer resources that are required for processing the work. Examples include operators, processors, and printers.
- Availability of resources
- Job tracking requirements
- Alternative locations for processing if the resources become unavailable

Tivoli Workload Scheduler for z/OS workstations objects map the processing requirements of each job to its type of activity. When you schedule, monitor, and control the activity, the workstation assignment defines how you and Tivoli Workload Scheduler interact with that activity. The types of workstations that are provided are as follows:

- Job setup, both manual and automatic.
- Job submission.
- Started-task actions.
- Automation (communication with the Tivoli NetView for z/OS or System Automation programs).
- Printing.
- Manual preprocessing or post-processing activity.
- Automatic delay (wait).

- Distributed (UNIX or Windows) computers.
- Remote engine workstations to associate your dependencies with jobs managed by other Tivoli Workload Scheduler environments (this kind of dependency is called a **cross-dependency**).

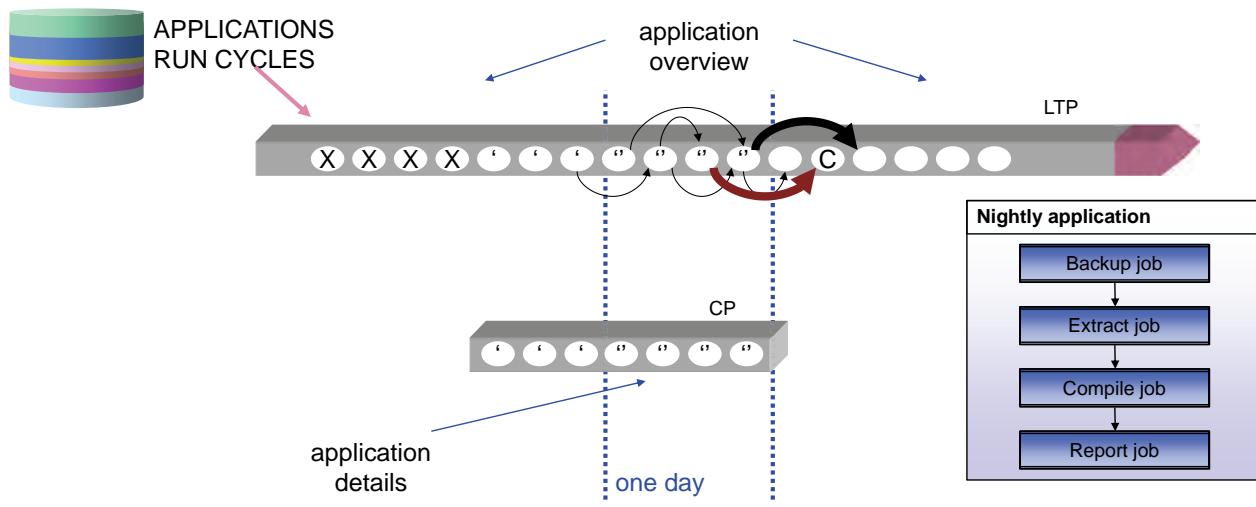
You can plan for maintenance windows in your hardware and software environments. By using the scheduler, you can perform a controlled and incident-free shutdown of the environment, preventing last-minute cancellation of active tasks. You can choose to reroute the workload automatically during any outage, whether planned or unplanned.

The scheduler tracks jobs as they process at workstations, and dynamically updates the plan with real-time information about the status of jobs. You can view or modify this status information online by using the workstation ready lists in the current plan.

Using **virtual workstations** improves workload balancing and the monitoring of system availability. This feature automatically directs the submission of workload to different destinations. There is no requirement to associate a workstation to a specific destination. The scheduling administrator can define a list of destinations for the submission of workload. The scheduler distributes the workload to the active destinations, assigning equal numbers of operations to each destination in a circular order.

By using virtual workstations, the scheduler distributes the workload across your trackers evenly, avoiding bottlenecks when submitting or running jobs. The scheduler splits the workload among the available destinations. The JES and Workload Manager (WLM) do not encounter overloaded input queues when selecting jobs for their action.

Long-term plan and current plan



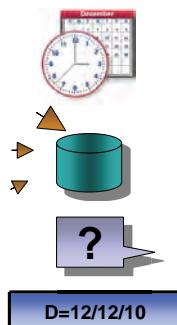
© Copyright IBM Corporation 2015

Long-term plan and current plan

The scheduler constructs operating plans that are based on the descriptions of the production workload that is created in the Tivoli Workload Scheduler for z/OS database. These plans provide the basis for automated production operations and give you a picture of the status of the production workload any time. There are three types of plans:

- **Long-term plan (LTP):** The plan that contains a high-level schedule of the anticipated workload. It lists, by day, instances of workloads to run during the period of the plan. Each instance is an **occurrence**. The long-term plan shows the occurrences that are to run, and the dependencies that exist between them. The plan can help you in forecasting and planning for heavy processing days.
- **Current plan (CP):** The plan that is the rolling detailed production database of operations to be run, along with their real-time status. The current plan is the center of Tivoli Workload Scheduler for z/OS processing. It drives the production workload automatically and provides a way to check its progress. A batch job extracts occurrences that fall within a specified period to produce the current plan. The current plan selects a window from the long-term plan and makes the jobs ready to run. Their start times depend on the decided restrictions. Examples of such restrictions are dependencies, resources availability, and time-dependent jobs.
- **Trial Plan:** A strategy that you generate to simulate the effects of changes to your production workload, calendar, and installation.

Scheduling objects



Objects

- Calendars
- Special resources
- Operator instructions
- Variable tables

Scheduling objects

Other scheduling objects in the Tivoli Workload Scheduler for z/OS database provide for customization of the workload automation planning. **Special resources** represent physical or logical objects that jobs use. The resource often represents a physical object in your configuration, although such representation is not required. You can use a special resource to manage running jobs concurrently. Examples include the following kinds of jobs:

- Serializing access to a data set.
- Limiting concurrent access to a database.
- Limiting the number of file transfers on a particular network link.

The scheduler keeps a record of the state of each resource and its current allocation status. The scheduling definition can hold resources allocated to a job if that job ends abnormally.

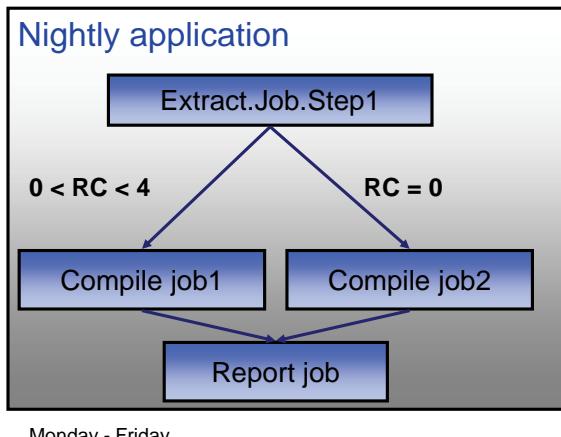
With **data set triggering**, you define data sets that TWS for z/OS should track. The data set triggering function of the scheduler automatically updates special resource availability when a data set closes. You can use this notification to coordinate planned activities or to add unplanned work to the schedule.

Tivoli Workload Scheduler for z/OS keeps information about work schedules or holidays. This information is useful to avoid scheduling workloads to run on days when processing resources are not available. Examples would be Sundays and holidays. **Calendars** store information about work days and free days. The scheduler supports multiple calendars for departments of enterprises that have different work days and nonworking days. Different groups within a business operate according to different calendars.

The scheduler supports automatic substitution of **variables** during the job setup and execution process. It also supplies several standard variables, which can be used in jobs. The workload designers can create their own variables by using **variable tables** that are stored in the scheduler database. You can use the same variable name in different variable tables. By associating the tables with different workloads, the value the variable takes depends on the application that uses it.

Some jobs might require specific instructions on handling. The scheduler database stores these **operator instructions**. The operator instructions are in a special field in the Tivoli Workload Scheduler for z/OS ISPF panels.

Applications



- Application name
- Validity dates
- Run cycles
 - Calendar references
 - Daily, weekly, monthly, for example
 - Periodically
- Jobs
- Time restrictions
 - Start times
 - Deadline times
 - Feedback
- Dependencies
 - Jobs in other applications (external)
 - Jobs in the same application (internal)
 - Resources
- Cleanup options

© Copyright IBM Corporation 2015

Applications

An **application** is a description of a unit of production work that contains related operations and runs during a specific period. Applications include items as follows:

- A list of the jobs and related tasks that are associated with that unit of work, such as the following examples:
 - Data entry.
 - Job Control Language (JCL) preparation for the job.
 - Job submission.
 - Started-task initiation.
 - Communication with the Tivoli NetView for z/OS or System Automation programs.
 - File transfer to other operating environments.
 - Printing of reports.
 - Post-processing activities, such as quality control or dispatch.
 - Any other tasks pertaining to the unit of work that you want to schedule, control, and track.
- A description of dependencies between jobs within an application and between jobs in other applications
- Information about resource requirements, such as exclusive use of a data set

- Special operator instructions that are associated with each job
- Methodology and location that each job is processed
- Run policies for that unit of work, such as its schedule or the name of a group definition that records the run policy

The items scheduled as part of an application are collectively called **operations**. When working with ISPF panels in Tivoli Workload Scheduler for z/OS, you use the term operation to describe a job or other task to be scheduled.

Tivoli Workload Scheduler for z/OS schedules workloads based on the information you provide in your application descriptions.



Hint: When you work with job streams in the Tivoli Workload Scheduler for z/OS ISPF panels, they are exclusively **applications**. When working with job streams in the Tivoli Dynamic Workload Console, they share the term *job streams* with Tivoli Workload Scheduler on distributed systems. In the units of this workshop that use ISPF, job streams are called *applications*.

Scheduled activities must occur in a specific order. Activities that are performed out of order might create invalid results and possibly even corrupt your data. Corrupted data might result in costly reruns, missed deadlines, and unsatisfied customers. You define dependencies for jobs that require specific processing order. When the scheduler manages the dependent relationships for you, the jobs always start in the correct order of scheduling.

- A dependency is *internal* when it is between two jobs in the same application.
- A dependency is *external* when it is between two jobs in different applications.

When specifying dependencies, the scheduling administrator can use both **return code** and **status** of an operation to determine the starting of another operation. Standard logical operators like *greater than* or *less than* define the check on status or return code values. These types of operators implement dependencies definitions with conditional logic. If the predecessor operation is associated to a job with different steps, the scheduling administrator can specify a conditional step-level dependency on individual step return codes.

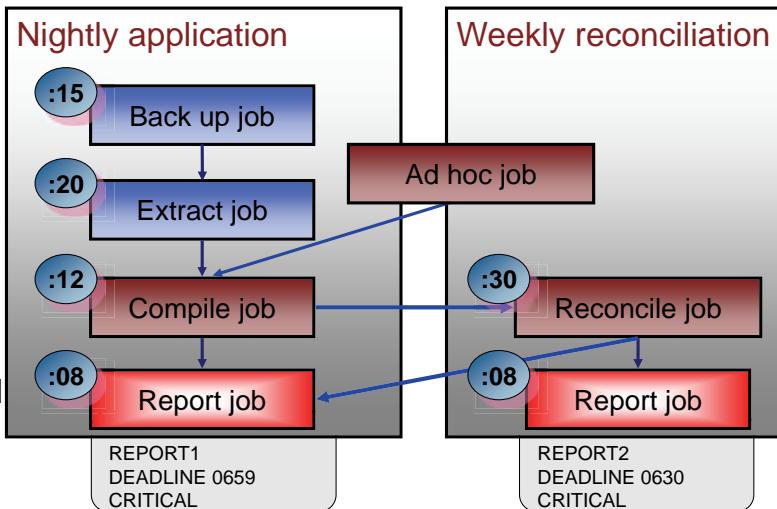
When an application is created, a **run cycle** determines which current plans contain that application to be run in production. The long-term planning process determines the days that an application is scheduled as part of that current plan by using calendar information, period definitions, and run-cycle groups with the applications' run cycle. Several factors, such as predecessor completion, resource availability, deadline time, and priority, determine when jobs (operations) within the current plan run.



Hint: Depending on how the run cycles and planning durations overlap, applications can be part of any given *current plan* zero or more times. Each instance of an application that is in the running current plan is an **occurrence**. Jobs and related tasks within the occurrences are **operations** in the current plan.

Workload Service Assurance

- Flags jobs as *critical*
- Associates them with a deadline time
- Tivoli Workload Scheduler calculates the latest start times for predecessors
- The chain with the least slack time becomes the *critical path*
- The *critical path* is recalculated in real time as the batch runs, considering overruns, late starts, and changes to the batch flow



© Copyright IBM Corporation 2015

Workload Service Assurance (critical path management)

IBM Tivoli Workload Scheduler for z/OS uses the capability of the Workload Manager (WLM) component of z/OS to ensure that critical jobs complete on time. If a critical job is late, Tivoli Workload Scheduler for z/OS uses the Workload Manager interface to give the job higher priority.

In addition to handling critical jobs based on Workload Manager, the scheduler dynamically handles critical workloads and their predecessors. This feature is called **Workload Service Assurance** or Critical Path Management

The **critical path** is the path, within a network of jobs, with the least slack time. **Slack time** is the amount of time the predecessor jobs can be delayed without impacting the deadline of a critical job. It is spare time that is calculated by using the deadline, input arrival, and duration settings of predecessor jobs.

The scheduler provides the following features:

- Monitoring of critical job predecessors that are late, long running, or ended with an error. This process uses the same internal logic that the scheduler applies to monitor alert conditions.
- Monitoring of the consumption of slack time in non-critical paths. These paths have the potential to become more critical than the paths calculated at plan generation.
- Monitoring of enhanced critical jobs by using ISPF panel flows.
- Server support for new views available by using the Tivoli Dynamic Workload Console.

If a job is critical and must complete by a deadline, you can specify the job as the target of a critical path. During daily planning processing, the critical paths are calculated, and a specific dependency network is created for each path. The dependency networks, including the internal and external

predecessors of the target jobs, are calculated. While the plan is running, Tivoli Workload Scheduler for z/OS monitors the paths that become more critical than paths calculated at plan generation.

For each job defined as critical, a specific critical path is calculated in the dependency network during daily planning processing. Starting from the target job, for each level of predecessors, the process chooses the most critical one and includes it in the critical path. Among all of the internal and external predecessors, the most critical one is the predecessor with the **latest end time**. Calculating the critical path starts from the target job and moves backward among its predecessors. The process ends when it reaches a job without a predecessor.

The planned end time is calculated as a result of the Tivoli Workload Scheduler for z/OS planning processing. The end time is based on the information you set for the job or application occurrence (for example the starting time, duration, and deadline). The workstation parallel servers, open intervals, special resources, and their availability also affect the end time. For this reason, the more accurate the application definition, the more accurate the critical path calculation. To have the scheduler update the estimates in the application description database with actual run values, duration feedback options are available.

While the plan runs, the scheduler checks if any predecessor of a critical job starts to delay. It also monitors how the non-critical path are using their slack time. The scheduler maintains a **hot list** for critical jobs, monitoring critical job predecessors that start to cause a delay in one of the following conditions:

- It is late, meaning that it did not start by its latest start time.
- It is long running, meaning that it is running longer than its estimated duration.
- It ends in error.
- A condition suppresses the job.

To monitor the job, Workload Scheduler applies the same internal logic that is used for monitoring alert conditions. Before recalculating a critical path, the scheduler estimates new start and end times for all the successors of the job that started to cause a delay:

- When any job in a critical path completes, the scheduler estimates new start and end times for its successors and recalculates the critical path.
- When a dynamic update changes the path, the scheduler estimates new start and end times for its successors and recalculates the critical path.
- When the estimated start and end times determine a critical path change, the scheduler updates the current plan.

The deadlines of the jobs in a critical job network affect the accuracy of the dynamic critical path handling. To optimize the accuracy, do not set deadlines on noncritical jobs or their applications.

You can monitor critical paths from the ISPF panels, through the Tivoli Enterprise Portal monitor in Tivoli Monitoring, or in the Tivoli Dynamic Workload Console.

Lesson 4 Version 9.1.0 differences overview

Lesson 4 Version 9.1.0 differences overview

- New run cycle group database object
- Several new matching criteria have been added for the Application Description
Defines how a dependency between predecessor and successor operations is resolved
- You can configure Tivoli Workload Scheduler for z/OS to comply with the Federal Information Processing Standards (FIPS) over SSL secured connections
- You can adopt the Workload Service Assurance (WSA) with prioritization on z-centric jobs
- IBM Tivoli Monitoring now uses a new customizable IBM Tivoli Monitoring agent instead of the Universal Agent

© Copyright IBM Corporation 2015

In this lesson, you learn about several of the major feature differences that are provided in Tivoli Workload Scheduler for z/OS version 9.1.0.

After completing this lesson, you should be able to describe the feature differences that are related to Tivoli Workload Scheduler for z/OS version 9.1.0.

There is a new database object that is called a ***run cycle group***. This new object is discussed in more detail as you proceed through the course. There are also new definitions for the criteria that are used to determine the dependencies between successor and predecessor operations. These new criteria are also discussed in more detail later in the course.

You can configure Tivoli Workload Scheduler for z/OS to comply with Federal Information Processing Standards (FIPS) over SSL secured connections. A new parameter that is named ENABLEFIPS was added to the initialization statements on the Tivoli Workload Scheduler for z/OS controller.

You can adopt the Workload Service Assurance (WSA) with the priority on zCentric jobs. This permits automatic promotion of the zCentric jobs that are on a critical path and are late.

The integration with IBM Tivoli Monitoring now uses a new customizable IBM Tivoli Monitoring agent instead of the Universal Agent for monitoring. Starting from version 9.1, the integration between Tivoli Workload Scheduler and IBM Tivoli Monitoring requires the installation of a new

customizable agent, the IBM Tivoli Monitoring agent for Tivoli Workload Scheduler. This agent replaces the Universal Agent that is used in previous versions. The new agent must be installed on the systems that you want to monitor, and it collects and distributes data to the Tivoli Enterprise Monitoring Server.

Run cycle groups

- A run cycle group is a distinct database object
- The same run cycle group can be used on different applications
- Run cycle groups enhance the use of negative run cycles
- Run cycle groups allow the use of a logical AND between individual run cycles in the group
- Full compatibility with traditional run cycles
- GENDAYS command at run cycle group level

© Copyright IBM Corporation 2015

Run cycle groups

A run cycle group is a list of run cycles that, combined together, produce a set of run dates. It can be reused by different application descriptions (job streams). A run cycle group can contain multiple subsets, each of which can have its own negative days, which do not affect dates in the other subsets. New types A (and) and D (negative and) can be used to dates such as, the last workday of a month that is also a Monday. You can use the Dynamic Workload Console or the ISPF panels to define and manage run cycle groups.

A new field, shift, is introduced on application run cycles.

Application Description dependency criteria

New criteria for selecting a dependent predecessor:

- The nearest preceding input arrival time
This option was already available in previous versions
- The nearest input arrival time within the same day of the successor
- The closest input arrival time within a specified interval
 - Interval boundaries are calculated using an offset or
 - Interval boundaries are calculated using a time and number days before or after the input arrival time of the successor

© Copyright IBM Corporation 2015

Application Description dependency criteria

Application Description matching criteria define how the dependency between predecessor and successor operations is resolved. The criterion that is used to select a predecessor that best resolves the dependency can be:

- The nearest preceding application input arrival (IA) time. In this case preceding includes the same input arrival time. This option was already available in previous versions.
- The nearest input arrival time within the same day of the successor.
- The closest input arrival time within a specified interval. The interval boundaries are calculated by using an offset that is expressed in hours and minutes before or after the IA time of the successor. The interval can be timed entirely before, entirely after, or across the IA time of the successor.
- The closest input arrival time within a specified interval. The interval boundaries are defined as a time and a number of days before or after the IA time of the successor. The interval can be timed entirely before, entirely after, or across the IA time of the successor.

In addition, the resolution of a normal dependency can be defined as mandatory. If no predecessor matches the specified criteria, the successor operation can not start until manually unblocked by the operator. The mandatory dependency resolution mechanism does not apply to conditional or cross dependencies.

Lesson 5 Version 9.2.0 differences overview

Lesson 5 Version 9.2.0 differences overview

- New z/OS and ISPF based features and new Dynamic Workload Console and Mobile Device features
- The following new features have been included in Tivoli Workload Scheduler for z/OS 9.2.0:
 - New filter for operations waiting for pending predecessors
 - Dependency resolution time interval for mandatory pending predecessors is now displayed in ISPF and the DWC
ISPF options 5.3 and 6.3
 - Status of system automation workstations can be changed in the plan
 - Maximum number of parallel servers per workstation increased to 65535
 - Job run history reports now have statuses of jobs ended in error and manually set to completed
 - Incident reports can be created through OSLC integration with Smart Cloud Control Desk
 - New heartbeat checking mechanism for z-centric agents and dynamic domain managers

© Copyright IBM Corporation 2015

In this lesson, you learn about the new features and enhancements that are provided with Tivoli Workload Scheduler for z/OS 9.2.0. After completing this lesson, you should be able to describe the new features and enhancements that are provided with Tivoli Workload Scheduler for z/OS 9.2.0.

Tivoli Workload Scheduler for z/OS 9.2.0 offers a wide range of new features and enhancements. These are mentioned briefly in this lesson. Several of the new features that are implemented in the ISPF dialogs are reviewed further as you progress through more units in the course. However, many are only mentioned here in this lesson. The new z/OS and ISPF based features are presented followed by the Dynamic Workload Console and mobile device features. The features that are reviewed on this slide are:

- New filter to show the operations that are waiting for pending predecessors. On ISPF, and on the Dynamic Workload Console, you can browse and modify operations in the current plan that are in waiting status for pending and mandatory pending predecessors.
- The dependency resolution time interval is now shown for mandatory pending predecessors. After you display a list of operations in waiting status for mandatory pending predecessors, you

view the information that is related to the predecessors and successors of a specific operation and find the following columns:

- Resl Crit: displays the criteria that are specified for the resolution of the dependency in the dependency definition.
- Mandatory Pending Interval: Shows the interval where the Input Arrival Time of the predecessor must be so that the dependency is solved. The interval boundaries are listed in terms of From and To dates. A blank From date signifies that the left side of the interval is open.

This information can be viewed also in the Dynamic Workload Console.

- The status of system automation workstations can now be changed in the plan. System automation workstations in the plan can change status in the same way as computer automatic workstations. You can see and change the status of a system automation workstation from the ISPF panels or from the Dynamic Workload Console. You can change the status of a system automation workstation from offline to active or vice versa in one of the following ways:
 - With the VARY command from the ISPF MCP panels.
 - Using the WSSTAT TSO command.
- Maximum number of parallel servers per workstation increased to 65535. The highest number of parallel servers that can be defined for a workstation increased from 99 to 65535. The parallel servers represents the number of operations that can be started simultaneously on a workstation.
- Job run history reports add statuses of jobs that ended in error and manually set to completed. Previous versions included only the history of jobs that were rerun after ending in error. Now jobs that are manually set to complete following an error status are included.

© Copyright IBM Corp. 1991, 2014 3

- Create incident reports through OSLC integration with SmartCloud Control Desk: Tivoli Workload Scheduler for z/OS integrates with SmartCloud Control Desk, through an Open Services for Life cycle Collaboration (OSLC) interface, so that when a job that matches a defined policy ends in error, an incident report is automatically opened. It is possible to configure this integration through the new OSLCOPTS statement (see Tivoli Workload Scheduler for z/OS Customization and Tuning). By configuring the OSLCOPTS parameters, users can:
 - Select the policies that determine which jobs that ended in error has an incident opened
 - Provide the ticket description text with the possibility to use predefined variables.
- Heartbeat checking mechanism for zCentric agents and dynamic domain managers: the Tivoli Workload Scheduler for z/OS controller uses the HTTP protocol to communicate with zCentric agents and dynamic domain managers. If these become unavailable, the controller does not register it until the next job submission takes place. To prevent the inherent loss of time, an internal heartbeat checking mechanism is added with this version. The controller checks at specified intervals the status of all attached zCentric agents and dynamic domain managers. This feature is optional and the pulse interval (in minutes) can be configured both globally for all

1 Introduction

Lesson 5 Version 9.2.0 differences overview

attached devices and individually for each agent. Destination pulse intervals can be updated by using the HTTP refresh destination command without having to stop the controller.

Version 9.2.0 z/OS based differences (continued)

- New JOBREC keyword for the script file extension
New EXTENSION keyword can now be added in JOBREC statements defining native jobs launched by z-centric agents
- The Tivoli Workload Scheduler for z/OS product version is now displayed in the EQQMLOG message log
Example: EQQZ403I RUNNING TWS for z/OS 9.2.0
- New message EQQIC66I issued in the MLOG of the migration JCL when VSAM files of the Application Description data base are migrated with the EQQICTOP conversion program
- Tivoli Workload Scheduler for z/OS can now be configured to use two message log data sets, EQQMLOG and EQQMLOG2
You alternate between the two MLOGs. While one logs messages, the other remains inactive
- View z/OS job logs with Tivoli Output Manager
View the z/OS job logs of operations run with Tivoli Workload Scheduler for z/OS

© Copyright IBM Corporation 2015

Version 9.2.0 differences overview (continued 1)

The features that are reviewed on this slide are:

- New JOBREC keyword for the script file extension: the new EXTENSION keyword can now be added in JOBREC statements that define native jobs that are launched by zCentric agents. The keyword is used with the JOBCMD(SCRIPT) keyword and its value is the file extension of the script started by the job. The script file extension can be specified also through the Tivoli Dynamic Workload Console in the executable plug-in.
- The product version is displayed in the Message Log (EQQMLOG): the message log now displays message EQQZ403I with the version of the Tivoli Workload Scheduler for z/OS installation. For example, MLOG listings for this product version display the following message: EQQZ403I RUNNING TWS for z/OS 9.2.0 The message follows the lines that show the parsing of the started task parameters in all the currently available message logs; that is:
 - Controller
 - Tracker
 - Server
 - Output collector
 - Data store
 - Batch MLOG
- New message EQQIC66I in the MLOG of the migration JCL when the Application Description database is upgraded with the EQQICTOP conversion program. The EQQICTOP conversion program now issues message: "EQQIC66I PROCESSING APPLICATION AD_data_set_name

VALID FROM From_date STATUS status" each time it migrates an AD record from the Application Description VSAM database to the new product release. It is possible to inhibit this feature by setting to N the TRACE parameter of the CONVERT command that is included in the EQQICTOP conversion program.

- Configuring Tivoli Workload Scheduler for z/OS to use two message log data sets, EQQMLOG and EQQMLOG2: using two MLOG data sets removes the necessity of stopping the controller to be able to save, or clear, the contents of a single data set. It also prevents the messages from being recorded in the system log when the controller is not stopped for a long time and EQQMLOG runs out of space. The two MLOGs are used alternatively: whilst one logs messages, the other is inactive. When the active data set reaches a defined level of completion, its contents are copied into a GDG data set and it becomes idle and the other data set starts logging. When the same level of completion is reached in the now active data set, the switch is repeated.
- Viewing z/OS job logs with Tivoli Output Manager: If you use Tivoli Output Manager in your enterprise, you can use it to view the job logs of operations run by Tivoli Workload Scheduler for z/OS. After Tivoli Workload Scheduler for z/OS and Tivoli Output Manager have been properly configured, enter the Browse joblog via ITOM (LJ) command, which is available in the same ISPF panels that the Browse joblog (L) command is available. The panels where the LJ command is available are related to browsing or modifying operations in the current plan:
 - Modifying operations in the current plan (EQQMMOPL, EQQMOPRL, and EQQMOPRR)
 - Operations history list (EQQHISTL)
 - Handling operations ended in error (EQQMEE1L, EQQMEE2L, EQQMEEP1L, and EQQMEEP2L)
 - List dependency status change (EQQMOSTL)
 - Rerunning an occurrence in the current plan (EQQMROCL)
 - Operations in the current plan (new panels - EQQMOPRV or EQQSRLCP)
 - Operation in the current plan (new panel - EQQSOPSD - Operation menu)

Version 9.2.0 Dynamic Workload Console and mobile device differences

The following features that were implemented in Tivoli Workload Scheduler distributed and Dynamic Workload Console are available now for Tivoli Workload Scheduler for z/OS:

- New Self-Service Dashboard mobile application
Use your mobile device to define one or more dashboards filtering on subsets of jobs and workstations
- New look-and-feel for Self-service Catalog mobile app
- Single sign-on configuration for Self-Service Catalog and Self-Service Dashboards applications
- New monitoring task query for jobs, job streams, workstations or resources
You can now define a task to monitor jobs, job streams, workstations, and resources by specifying a query in a query line

© Copyright IBM Corporation 2015

Version 9.2.0 DWC and mobile device differences

The following features that were implemented on Tivoli Workload Scheduler and Dynamic Workload Console are available also for Tivoli Workload Scheduler for z/OS:

- Self-Service Dashboards mobile application: you can use your mobile device to define one or more dashboards that filter on subsets of jobs and workstations. The dashboards give an overall picture of your jobs and workstations. They enable drill down to view more detailed information about jobs, such as job details and the job log, and about workstations and their availability. You can also perform some recovery actions on jobs and workstations.
- New look-and-feel for Self-service Catalog mobile app: Self-Service Catalog is a mobile device application to submit service requests to Tivoli Workload Scheduler and Tivoli Workload Scheduler for z/OS. Service requests correspond to Tivoli Workload Scheduler job streams and to Tivoli Workload Scheduler for z/OS applications. Self-Service Catalog underwent a user interface design update, improving its general look-and-feel.
- Single sign-on configuration for Self-Service Catalog and Self-Service Dashboards applications: if the Dynamic Workload Console instance to which the applications connect is configured for single sign-on. A user can log in once to the Dynamic Workload Console and then gain access to the applications without being prompted to log in again. Global settings file is also available with local installation. `TdwGlobalSettings.xml`, is a file that users can optionally edit to configure some advanced Dynamic Workload Console settings. This file is now also installed locally and you can find it in the following path after you install the Dynamic

Workload Console on your workstation: JazzSM_Installation_Path/profile/registry/ The file is installed as TdwcGlobalSettings.xml.template. If you edit it, save it as TdwcGlobalSettings.xml.

- Monitoring task query for jobs, job streams, workstations, or resources: you can now define a task to monitor jobs, job streams, workstations, and resources by specifying a query in a query line. You can specify the query by selecting from a series of user interface options. You can save your queries and then reuse or edit them as necessary.

Version 9.2.0 DWC and mobile device differences (continued)

- New performance improvement for batch operations with a large number of job streams on the Dynamic Workload Console
- Self-Service Catalog mobile application
Administrators can tailor services by defining variable validation criteria to be applied to the job streams associated to a service
- New predefined role to use and manage the Self-Service Catalog and Self-Service Management mobile applications
New role of TWSWEBUIBusinessDeveloper
- New Dynamic Workload Console global setting for displaying the rows in a table view corresponding to results of a monitoring task
Sets the maximum number of rows to display in a table view

© Copyright IBM Corporation 2015

Version 9.2.0 DWC and mobile device differences (continued)

The following features that were implemented on Tivoli Workload Scheduler and Dynamic Workload Console are available also for Tivoli Workload Scheduler for z/OS:

- Performance improvement for batch operations with large number of job streams on the Dynamic Workload Console: previously a timeout error was returned or the performance was very slow when batch operations were performed. Instead of sending separate requests for each job stream now only a single request is sent for all selected job streams.
- Self-Service Catalog mobile application: administrators can tailor services by defining variable validation criteria to be applied to the job streams associated to a service. By defining variable validation criteria, Administrators can ensure that input that is entered by the mobile user is checked against the criteria and only input that satisfies the criteria is accepted. When defining variable validation criteria, Administrators can specify, for example, whether the input value should be numeric or alphabetic, whether the entry must be within a range of values, or a certain length, or if the value is obligatory.
- A new predefined role to use and manage the Self-Service Catalog and Self-Service Management mobile applications: users with the new role assigned, TWSWEBUIBusinessDeveloper, can access and use the Self-Service Catalog and the Self-Service Dashboards mobile applications. From the Self-Service Catalog mobile application, these users can create and edit catalogs, create and edit services, add services to catalogs, and submit the services associated to job streams. From the Self-Service Dashboards mobile application, these users can create and edit services to filter for jobs and workstations, run services to view a dashboard of results, perform recovery actions on a single result. To share catalogs and services with other user, the TWSWEBUIBusinessDeveloper can assign the

catalog or service to custom roles they possess, and not to predefined roles. Users with these same custom roles can work with the catalogs and services.

- New global setting for displaying the rows in a table view corresponding to results of a monitoring task: the Dynamic Workload Console administrator can set the maximum number of rows to display in a table view after running a monitoring task. A new property, `maxRowsToDisplay`, is added to Section 8 of the Dynamic Workload Console global settings file, `TdwcGlobalSettings.xml`.

Version 9.2.0 DWC and mobile device differences (continued)

- Auditing mobile device application activities
Administrators can configure logging information in the Dynamic Workload Console global settings file for Self-Service Catalog and Self-Service Dashboards mobile applications
- You can now monitor critical jobs from your mobile device
You can define one or more dashboard filters for subsets of jobs and workstations
- New browser support for the Dynamic Workload Console, the Self-Service Catalog and Self-Service Dashboards mobile applications
Firefox 24 ESR

© Copyright IBM Corporation 2015

Version 9.2.0 DWC and mobile device differences (continued)

The following features that were implemented on Tivoli Workload Scheduler and Dynamic Workload Console are available also for Tivoli Workload Scheduler for z/OS:

- Auditing mobile device application activities: administrators can configure logging information in the Dynamic Workload Console global settings file, `TdwcGlobalSettings.xml`, related to operations performed from the Self-Service Catalog and Self-Service Dashboards mobile applications. A new section, Section 11, is added to the global settings file and the log file is enabled by default
- Monitor critical jobs from your mobile device: you can use your mobile device to define one or more dashboards that filter on subsets of jobs and workstations. The dashboards give an overall picture of your jobs and workstations and enables you to drill down and view more detailed information about jobs and perform recovery actions. Jobs can be further filtered to produce a set of results corresponding to critical jobs in the network that is categorized by jobs with a high risk and jobs with a potential risk.
- New browser support: the Dynamic Workload Console, the Self-Service Catalog, and Self-Service Dashboards mobile applications are all supported on Firefox 24 ESR.

Review questions

1. Name three products in the IBM Tivoli Workload Automation suite.
2. Name the three basic architectures for deploying the IBM Tivoli Workload Scheduler for z/OS.
3. Which Tivoli Workload Scheduler for z/OS entity maps processing requirements (such as job submission, printing or UNIX execution) to its type of activity?
4. Tivoli Workload Scheduler for z/OS generates operating plans based on the descriptions of the production workload in the database. What are the three types of operating plans?
5. Name three things that an application (also called job streams) can refer to.

Review answers

1. Name three products in the IBM Tivoli Workload Automation suite.

IBM Tivoli Workload Scheduler, IBM Tivoli Workload Scheduler for z/OS, and IBM Tivoli Workload Scheduler for Applications. Tivoli Dynamic Workload Console is also part of the automation suite.

2. Name the three basic architectures for deploying the IBM Tivoli Workload Scheduler for z/OS. *z/OS-only, classic fault-tolerant capable end-to-end, and zCentric end-to-end.*

3. Which Tivoli Workload Scheduler for z/OS entity maps processing requirements (such as job submission, printing or UNIX execution) to its type of activity?

Workstations.

4. Tivoli Workload Scheduler for z/OS generates operating plans based on the descriptions of the production workload in the database. What are the three types of operating plans?

Long-term plan, current plan, and trial plan.

5. Name three things that an application (also called job streams) can refer to.

Any three of the following items: jobs, Tivoli NetView for z/OS, or System Automation tasks, dependencies, operator instructions, workstations, run policies, cleanup options, validity dates, run cycles.

Summary

- Describe the IBM Tivoli Workload Automation suite
- Describe these properties of IBM Tivoli Workload Scheduler for z/OS
 - Structure
 - Major functions and components
 - Relationship to single-host and multiple-host systems
- Describe the feature differences provided in IBM Tivoli Workload Scheduler for z/OS 9.1.0 and 9.2.0

© Copyright IBM Corporation 2015

Summary

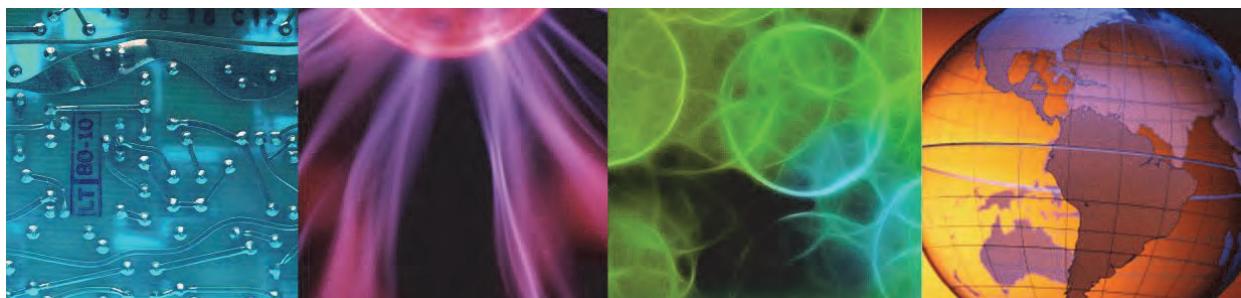


2 Workstations

IBM Tivoli Workload Scheduler 9.2.0



2 Workstations



© Copyright IBM Corporation 2015

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

In this unit, you learn how Tivoli Workload Scheduler for z/OS workstations function, and how to create such workstations.

References: SC32-1262 *IBM Tivoli Workload Scheduler for z/OS Managing the Workload*



Note: Workstation exercises follow this session.



Objectives

- List the types of available workstations
- Describe the functions of workstations in Tivoli Workload Scheduler for z/OS
- Create workstation definitions in Tivoli Workload Scheduler for z/OS

© Copyright IBM Corporation 2015

Objectives

A workstation in Tivoli Workload Scheduler for z/OS defines where work is scheduled to execute. This unit provides you with the knowledge to create workstations that reflect your working environment. The hands-on exercises re-enforce the lesson as you create workstations that will be used by the work you define throughout the course.

Lesson 1 Workstation overview

Lesson 1 Workstation overview

Physical workstations

- Use computers with operating systems: z/OS or distributed agents
- On z/OS, run jobs and started tasks
- WTO: Use the system console
- Automation: Use System Automation for z/OS and NetView
- Remote workstations for remote engine cross-dependencies

Logical workstations

- Can use manual process such as JCL preparation
- Just wait for a period of time
- Do not perform any action (dummy, or nonreporting)

© Copyright IBM Corporation 2015

This lesson provides an overview of the types of workstations that can be defined to Tivoli Workload Scheduler for z/OS. After completing this lesson, you should be able to list the various types of Tivoli Workload Scheduler Workstations.

References:SC32-1262 *IBM Tivoli Workload Scheduler for z/OS Managing the Workload*

Tivoli Workload Scheduler for z/OS supports various activities that you can schedule. Each activity is defined as an operation that is assigned to a **workstation**. Workstations map the processing needs of any operation in the production workload.

Each workstation supports a specific activity. You have the flexibility to schedule, monitor, and control various activities by assigning those activities as operations on different workstations. The activities can include the following examples:

- Setting up jobs, both manual and automatic
- Submitting jobs, including distributed Tivoli Workload Scheduler jobs
- Taking actions on started tasks
- Issue write-to-operator (WTO) messages to a z/OS system console
- Printing
- Connecting to Tivoli NetView for z/OS and requesting System Automation for z/OS services

- Submitting jobs to a workstation that is selected from a pool of workstations (z/OS only)
- Inserting a delay into an application job stream
- Manually preprocessing or post-processing jobs
- No activity at all
A workstation that supports null operations gives increased flexibility when establishing dependencies between operations
- Setting up cross-dependencies with remote jobs on remote engines
The engines can be either z/OS or distributed.

For scheduler planning and control purposes, the work to be performed is defined as individual operations. These operations take place at workstations. Each operation that the scheduler tracks, whether a job, started task, or other activity, must be associated with a workstation. The workstation is the point of control. The scheduler tracks jobs that are processed at workstations and dynamically updates the plan with real-time information about the status of jobs.

Users can browse or modify the status of work at workstations by using one of two means:

- The Ready List panel
- The Monitor Workstations task of the Tivoli Dynamic Workload Console

To define workstations in the scheduler, you must provide certain information that describes the basic characteristics of the workstation. These characteristics define the intended use of the workstation and the workstation capabilities.

Workstation types



Computer

- Jobs (z/OS and distributed)
 - z/OS started tasks
 - Virtual (pools of z/OS)



Printer

- Splittable
- Can be interrupted



General

- Job setup
- Write to operator
 - Automation
 - Wait
 - Splittable
 - Other tasks



Remote

- z/OS
- Distributed

© Copyright IBM Corporation 2015

Workstation types

The four major types of workstations are as follows:

- **Computer:** Workstations for running batch jobs and z/OS started tasks. Computer workstations for the distributed environment are defined with special attributes.
- **Printer:** Workstations for tracking, but not controlling, the production of printed output. Only the z/OS operating systems use printer workstations.
- **General:** Workstations that support a myriad of tasks, including manual activities, issuing WTO messages, and dummy operations.
- **Remote:** A remote engine workstation is used to represent a remote Tivoli Workload Scheduler engine or a remote Tivoli Workload Scheduler for z/OS controller

Workstation reporting attributes

Attribute	Description
Automatic	Tivoli Workload Scheduler for z/OS reports status changes automatically. Use for computer and print workstations.
Completion only	Completed work is normally reported by the workstation operator. Use for general workstations that are not used for job preparation.
Manual start and completion	All reporting is normally performed by a workstation operator. Use for general workstations used for job preparation or for other general workstations when the duration of a task needs to be tracked.
Non reporting	Jobs on this workstation are set to complete as soon as they become eligible to be started. Use for dummy jobs that do not require any processing.

© Copyright IBM Corporation 2015

Workstation reporting attributes

Every operation in the current plan is assigned a status. The status of an operation describes its current condition. When all processing for an operation finishes, the operation is assigned status C (complete).

Before an operation reaches completion, it moves through many statuses. The sequence of statuses that an operation is assigned, and the mechanism that is used for reporting status updates, depend on the reporting attribute of the workstation. The slide shows the four workstation reporting attributes.

Automatic: The status change of operations is normally reported automatically in response to event records created by the JES and SMF exit routines. You can also change the status of these operations manually by using the panels.

Completion Only: Used for workstations on which users want to control assignment of the Complete (C) status for operations that are scheduled on these workstations. Typically, operations that are defined on workstations with this attribute require manual interaction.

Manual Start and Completion: Used for workstations on which users want to manually edit job statements before job submission.

Non Reporting: You can use workstations with this attribute for operations that do not require any processing. For example, the operations can hold the dependencies for successors, or the operation might represent milestones in the processing.

Workstation options

Option	Description	Validity
Splittable	Jobs at the workstation can be interrupted	Cannot be specified for jobs running on computer workstations
Job setup	Used to prepare JCL	Computer workstations only
Started task	Jobs are treated as started tasks (STC)	Computer workstations only
WTO	Jobs are treated as WTO messages	General workstations only
Automation	Supports System Automation for z/OS	General, automatic reporting only
Wait	Performs no task, just waits	General, non-reporting workstations
Virtual	Load balancing of work across systems	Computer workstations only
Fault Tolerant	Used for jobs that run on a server via a fault-tolerant agent	Computer workstations only
z-centric Agent	Used for jobs that run on a server via a lightweight z-centric agent	Computer workstations only
Dynamic	Workstation represents a Dynamic Domain Manager – may provide workload balancing across servers	Computer workstations only
Remote Engine Type	z/OS or Distributed	Remote Engine Only

© Copyright IBM Corporation 2015

Workstation options

In addition to a type and reporting attribute, a workstation definition can have one of seven options. Only one option is valid for a single workstation definition. A workstation does not require the workstation option, but it might require the selection of an option to clarify the intended use of the workstation. You can think of the workstation option as a subtype. The options are as follows:

- **Splittable:** For general and printer workstations only. Work on this workstation is interruptible.
- **Started task (STC):** For running a started task, but only on z/OS2. This option is valid for computer workstations type with automatic reporting.
- **Automation:** Support for Tivoli System Automation for z/OS requests. This option is valid for general workstation type with automatic reporting.
- **Wait:** For creating a delay. This option is valid for general workstations with the nonreporting attribute.
- **Virtual:** To create multiple destinations which enables work load submission across multiple LPARs. This option is only valid for Computer workstations.
- **Job Setup:** Allows JCL editing. This option is valid for a general workstation with manual start and completion.
- **WTO:** For creating WTO messages on the console. This option is valid for general workstations with any reporting attribute.
- **Fault Tolerant Agent:** For specifying that the workstation definition is for a fault-tolerant agent active on a distributed system.

- **Z-Centric agent:** For specifying that the workstation is for a zCentric agent that is installed on a distributed system.
- **Dynamic:** For specifying that the workstation is for a Dynamic Domain Manager that is installed on a distributed system
- **Remote Engine Type:** Only valid for workstation type Remote Engine. It specifies the type of remote engine to connect to, a z/OS Controller or a distributed Master Domain Manager.r

Computer, remote, and general workstations

Computer workstations

- Job submission
 - z/OS
 - z-centric
 - Fault tolerant
- Started tasks
 - Remote workstations
 - Shadow jobs

General workstations

- Nonreporting
- Manual interaction
- Issue messages
- Delayed execution
- Tivoli System Automation for z/OS integration

© Copyright IBM Corporation 2015

Computer, remote, and general workstations

Computer workstations

Most operations in a batch-production schedule are batch jobs. These operations run on computer workstations where they automatically start when all requirements are met.

You can also use computer workstations to start started tasks. A started-task computer workstation has the STC option on the workstation general information panel set to **Y**. Operations that you specify to run on this workstation are started tasks, not jobs.

Only jobs can be defined as operations on a computer workstation if the STC option is set to **N**. Set the STC option to **Y** if you want to run started tasks.

The scheduler passes control to the z/OS operating system and JES when it submits a job or starts a started task. The scheduler tracks the job or started task but does not affect how the job or started task runs. It automatically tracks batch jobs and started tasks to completion.

You must keep the job statements for z/OS jobs in partitioned data sets that are allocated to the Tivoli Workload Scheduler for z/OS controller. The JCL for started tasks is stored in the same way as JCL for jobs. However, instead of submitting the job to the internal reader, the scheduler puts the STC JCL into a procedure library called EQQSTC. The scheduler then issues the z/OS START command.

A **virtual workstation** is a computer workstation that is associated with a pool of destinations.

A fault-tolerant agent is a computer workstation that is configured to schedule jobs on systems that are running a fault-tolerant agent. You use a fault-tolerant workstation to schedule jobs on

computers in the IBM Tivoli Workload Scheduler network. You define the workstation by setting the FAULT-TOLERANT AGENT attribute on the workstation general information panel to **Y**. You must also define the workstation in the CPUREC initialization statement in the Tivoli Workload Scheduler for z/OS parameter library.

Like z/OS and tracker agent jobs, you can store job statements for the distributed environment in the scheduler job library data sets. This option has its advantages and disadvantages. You must carefully weigh each case when deciding. This option is defined at the job level in the unit or work. By default, the job statements for FTA workstations are stored on the server they represent.

A zCentric agent is a computer workstation that is configured to schedule jobs on a distributed computer that has a zCentric agent installed. You use a zCentric workstation to schedule jobs on agents that are directly connected to the Tivoli Workload Scheduler for z/OS controller. The workstation is defined with the Z-CENTRIC AGENT attribute set to **Y**. You use the ROUTOPTS statement in the EQQPARM library to define the mapping between the DESTINATION field and the network address of the zCentric agent.

General workstations

With a general workstation, you control operations that are normally not automatically controlled. A Non-reporting, general workstation is often used for **dummy** operations, which are operations that do not require any processing. Operations at a Non-reporting workstation are set to completed status when they become eligible to be started. No job, started task, or WTO is submitted.

Tivoli Workload Scheduler for z/OS also uses special general workstations to schedule tasks as follows:

- Setting up JCL for jobs and started tasks.
 - A job setup workstation has the **JOB SETUP** option on the workstation general information panel set to **Y**. With a job setup workstation, users can prepare job or started-task JCL manually before execution. The **SPLITTABLE** option is often used with job setup so the activity can be interrupted.
 - Job setup operations must immediately precede the computer workstation operation for the job.
- To issue WTO messages, possibly to trigger action by an automation product.
 - WTO general workstations have the **WTO** option on the general information panel set to **Y**. With a WTO workstation, users schedule a WTO message at the designated destination.
 - The message is a multi-line WTO (MLWTO) message with the message ID of EQQW775I. It contains information about the operation, the application, and the workstation that issued it. The variant information in the message is the text that is associated with the operation.
 - Where an IBM System Automation product is installed an Automation Workstation can be used to interface more effectively.

- Implementing a delay in the job stream with a wait workstation.
- The Automation option causes a request to a System Automation for z/OS agent and receives a response.

Remote workstations

Use shadow jobs on remote workstations to track operation dependencies that are scheduled in another Controller or Master Domain Manager.

Lesson 2 Creating workstation definitions

Lesson 2 Creating workstation definitions

```
EQQWCGEP ----- CREATING GENERAL INFORMATION ABOUT A WORK STATION -----
Command ==> -
Enter the command R for resources A for availability O for end-to-end options
or D for Destinations above, or enter data below:

WORK STATION NAME ==> _____
DESCRIPTION ==> _____
WORK STATION TYPE ==> G      G General, C Computer, P Printer
                           R Remote Engine
                           A Automatic, S Manual start and completion
                           C Completion only, N Non reporting
REPORTING ATTR ==> S      The dname of daily plan printout data set
PRINTOUT ROUTING ==> SYSPRINT
SERVER USAGE ==> N      Parallel server usage C , P , B or N
DESTINATION ==> _____      Name of destination
Options: allowed Y or N
SPLITTABLE ==> N      JOB SETUP ==> N
STARTED TASK, STC ==> N      WTO ==> N
AUTOMATION ==> N      FAULT-TOLERANT AGENT ==> N
WAIT ==> N      Z-CENTRIC AGENT ==> N
VIRTUAL ==> N      DYNAMIC ==> N
REMOTE ENGINE TYPE ==> _____      Z z/OS or D Distributed
Defaults:
TRANSPORT TIME ==> 00.00      Time from previous work station HH.MM
DURATION ==> _____      Duration for a normal operation HH.MM.SS
```

- Required fields are red
- Workstation name consists of four characters, starting with a letter
- Destination name is required for a z/OS Tracker workstation

© Copyright IBM Corporation 2015

This lesson provides an overview of how to define workstations to Tivoli Workload Scheduler for z/OS. After completing this lesson, you should be able to define Tivoli Workload Scheduler Workstations.

References: SC32-1262 *IBM Tivoli Workload Scheduler for z/OS Managing the Workload*

The primary workstation definition panel is called Creating General Information about a Work Station. You have the following options on this panel:

- WORK STATION NAME: Type a four-character alphanumeric name for the object in the workstation database. The first character must be an alphabetic character.
- DESCRIPTION: Type a 24-character description.
- WORK STATION TYPE: **G** for general, **C** for computer, or **P** for printer to set the workstation type.
- REPORTING ATTR: **A** for Automatic, **S** for Manual start and completion, **C** for Completion only, or **N** for Non-reporting.
- SERVER USAGE: Specify whether Tivoli Workload Scheduler for z/OS must detect the availability of parallel servers when generating a current plan or submitting jobs. The valid options are **B**, **N**, **P**, and **C**. Take care when using **B** (Both) or **P** (Planning) as these values

might affect the Critical Path calculations if the workstations are offline or unavailable and are rerouted to an alternate workstation.

- DESTINATION: Use this option for computer and general WTO workstations to specify where the controller must send the job, started task, or message. However, do not use the DESTINATION field for fault-tolerant workstations. The destination can be places such as these examples:
 - A VTAM (Virtual Telecommunications Access Method) logical unit (LU) name in VTAM configurations.
 - An XCF (Cross Systems Coupling Facility) member name in a monplex or sysplex configuration.
 - A user-defined Data Definition name of a submit or release data set in a shared DASD (Direct Access Storage Device) configuration.
 - An IP (Internet Protocol) name when defining a workstation for a tracker agent.
 - Blank, when the job, started task, or message is destined for the system on which the controller is running.
- In the OPTIONS fields, you must type a **Y** or an **N**.
 - SPLITTABLE: Operations at the workstation can be interrupted.
 - JOB SETUP: The workstation is used for JCL preparation.
 - STARTED TASK, STC: Operations at the workstation are treated as started tasks.
 - WTO: Operations are treated as WTO messages.
 - AUTOMATION: Operations are treated as System Automation for z/OS commands.
 - FAULT-TOLERANT AGENT: The workstation is a fault-tolerant agent on a distributed system.
 - WAIT: Operations at the workstation perform no task, but they wait for the specified period.
 - Z-CENTRIC AGENT: The workstation is an agent on a distributed system.
 - VIRTUAL: The workstation contains a list of destinations.
 - DYNAMIC: The workstation points to a Dynamic Domain Manager.
- Defaults DURATION: (Optional) Provide an estimated processing time for operations that are run on the workstation, this default value is inserted in operation definitions if no other is provided.

To run scheduled work, scheduler workstations must be available (A) and online in the current plan.

Workstation availability

To be available for work, a workstation must be open, active, and connected. By controlling workstation availability, you control when the workstation runs operations that are defined on the

workstation. The scheduler establishes the availability of a workstation by using the open intervals in the workstation description database.

Open intervals are times of day when the workstation processes work. Define open intervals for each day of the week, or simplify the process by defining typical operating intervals as a STANDARD day. The ideal method for defining open intervals is defining open intervals for STANDARD and have the days of the week use those definitions.

On the Workstation Availability panel, you can specify deviations from the standard availability for certain dates. You can use this feature to identify a partial-day workstation closure for a planned maintenance window, or to close the workstation for an entire day. Fault-tolerant workstations do not use the workstation availability feature because they are always available, if they need to communicate with the Controller they need to be linked.

Open intervals

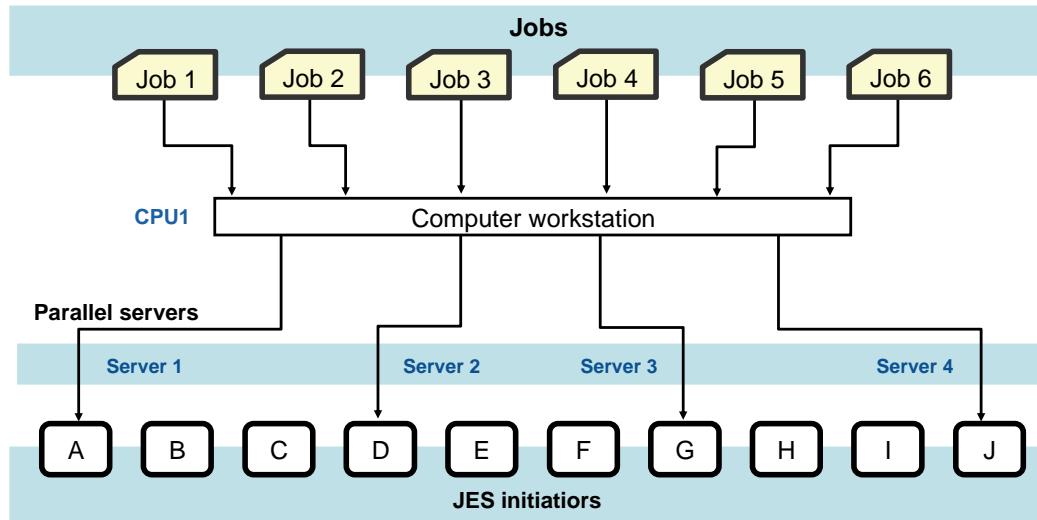
On this panel, you define the times when the workstation parallel servers are available to process work. If no parallel servers are available, no work runs at the workstation.

The defaults for STANDARD are as follows:

- **Open time:** 00.00
- **Interval:** 24.00
- **Parallel servers:** 00099
- **Resources R1 and R2:** 99

You can specify different Open time intervals with varying numbers or Parallel servers within a 24-hour period. Use this panel to specify an alternative workstation for the workstation you are creating. Operations that are ready to run at a workstation that is inactive or failed can be rerouted to an alternative workstation. The values that are specified on this panel reflect the resources of your system that are available to Tivoli Workload Scheduler for z/OS.

Parallel servers



© Copyright IBM Corporation 2015

Parallel servers

The number of parallel servers a workstation owns limits the number of operations that can be in Started or Error status at any one time. Parallel servers are not used for fault-tolerant workstations.

At a computer workstation, a parallel server can be thought to represent a JES initiator. Operations that are defined to a computer workstation must allocate at least one parallel server.

This example shows that jobs 1 - 6 were defined on a computer workstation, CPU1, which has four parallel servers. Server usage C (control) was specified for this workstation. Therefore, CPU1 can run only four of the six ready jobs concurrently. It does not matter that there are unused JES initiators available.

The default value for parallel servers is 99 the maximum value can be 65535.

You should determine with your system administrators what a good value might be here. If WLM initiators are being used, then the system spawns a new initiator for each job submitted. It might be that the capacity managers would like to restrict the number of batch jobs that run at certain times of the day.

In shared spool and SYSPLEX z/OS environments, consider having a virtual workstation represent the environment with a destination for each LPAR. This causes jobs to be submitted to all the systems in turn, rather than just one, rotating around the destinations. When batch should run on a specific LPAR use job class, system affinity or WLM Scheduling environments to direct the work.

You can change workstation definitions in the current plan to reflect real time situations and change the numbers of jobs that can be submitted/.

Workstation activity

BROWSING WORK STATION ACTIVITY ---- Row 33 to 56 of 65										
Command ==> _ Scroll ==> PAGE										
For more information, enter any of the row commands below: O Open intervals, S Summary information, I System Information										
Row cmd	Work station name text	F	V	L	S	T	R	Completed oper dur.	Active oper	Remaining oper dur.
		A								
INW3	GENERAL : WTO's on C1	N	U	G	C	0	0.00	0	0	0.00
OMVS	COMPUTER: JOBS ON OS/	N	U	C	A	0	0.00	0	0	0.00
PDMA	PRI DOM MGR-TOPDOMA (N	A	C	A	0	0.00	0	0	0.00
PDM1	PRI DOM MGR-TOPDOM1 (N	A	C	A	0	0.00	0	0	0.00
PDM5	PRI DOM MGR-TOPDOM5 (N	A	C	A	0	0.00	0	0	0.00
P000	COMPUTER: For ADHC on	N	U	C	A	0	0.00	0	0	0.00
SA03	Standard Agent in dom	N	U	O	C	A	0	0.00	0	0.00
U#CP	Computer work station	N	A	C	A	0	0.00	0	12	0.23
U#JS	Job Setup WS for Team	N		G	S	0	0.00	0	2	0.07
U#NR	Non reporting for Tea	N		G	N	0	0.00	0	0	0.00
U#WT	Wait WS for Team #	N		G	N	0	0.00	0	3	0.03
U#ZC	z-centric workstation	N	A	C	A	0	0.00	0	0	0.00

- **[F] All Virtual Workstations destinations active:** Y=All active N=Not all active
- **[V] Virtual:** Y=Yes, a virtual workstation, N=not a virtual workstation
- **[L] Linked FTA:** L=Linked, U=Unlinked, F=Fully Linked
- **[S] Status:** A=Active, F=Failed, O=Offline, U=Unknown, I=Inactive
- **[T] Type:** C=Computer, G=General, P=Printer, R=Remote
- **[R] Reporting attribute:** A=Automatic, C=Completion only, N=Non-reporting, S=Manual start and completion

© Copyright IBM Corporation 2015

Workstation activity

ISPF option 4 provides several options for communicating with workstations:

- View the list of operations that are ready to run.
- View a list of waiting operations.
- Perform job setup for operations, such as editing JCL.
- View the detailed status of a workstation. This ISPF option is 4.4, which is shown on the slide.

Workstation ready list

```

EQQRLRLM ----- READY LIST ----- Row 1 to 13 of 13
Command ==> -
Enter the HIST primary command or
enter any of the following row commands:
N - Set NEXT logical status, N-x - Set specific status( x ),
R - Reset status, O - Operator Instructions, I - Information about operation,
MH manual hold operation, MR manual release operation, NP nop operation,
UN un-nop operation, EX execute operation, BND Reset bind information.

WORK STATION      ==> OPAC      Change to switch work station
LAYOUT ID         ==> C1       Change to switch layout id

Cmd St no. Jobname Operation text Job id Application Oi U
E 5 OPACD107 Rerun of OPACD107 JOB05917 JOBRERUN N Y
E 10 OPACJ10A
E 20 OPACJ10B
E 60 OPACNCT2
E 5 OPACPM0A Rerun of OPACPM0A JOB15594 KBTESTMSGCLASS N Y
SS 50 OPACD100
SU 20 OPACPM6B
SS 30 OPACPM6C
SS 110 OPACPM7A
**X 10 OPACD106
AT 20 OPACD107
**X 80 OPACD108
AT 40 OPACD109

```

The Layout ID determines what values are displayed (column headings). Several different layouts are provided, however you can build your own customized layouts to show the information you like to see.

You can swap between layouts and workstations from here by overtyping the current value.

Operations in Waiting or Complete cannot be seen on this display.

© Copyright IBM Corporation 2015

Workstation ready list

ISPF option 4.1 contains the following fields:

- **WORK STATION:** Specify a workstation name. If you leave this field blank or type an asterisk (*), you get a list of all workstation names.
- **LAYOUT ID:** Specify the name of a ready list layout. If you leave the name blank, the layout that loads is the last one that you used. You receive a selection list the first time you use it. Type an asterisk (*) to see a list of all existing layouts.

You can specify one of these row commands:

- **I:** Display operation details.
- **N:** Set next logical operation status. What that is depends on the reporting attribute of the workstation
- **N-x:** Set a specific status, where x can be one of the following:
 - **C:** Completed
 - **E:** Ended-in-error
 - **I:** Interrupted
 - **R:** Ready
 - **S:** Started
- **O:** Display operator instruction.
- **R:** Reset operation status to previous status

- **MH:** Manual Hold operation.
- **MR:** Release a manually held operation
- **NP - NOP** (No operation): Tivoli Workload Scheduler for z/OS sets the status of the operation to status C (complete) immediately the operation becomes ready.
- **UN:** NOP the operation. Reverses the effect of NOP.
- **EX:** Start the operation, ignoring all usual start criteria. An active workstation must be available for this operation.
- **BND:** Reset the bind information for the shadow job. A new bind process can start.

You can enter the **SORT** command to change the sort order of the list. You can select items to sort by that are not in your layout.

To modify a workstation status, you can use ISPF option **5.5.0** and enter the **S** row command to select a specific workstation.

Student exercises



See the Student Exercise Guide Unit 2

© Copyright IBM Corporation 2015

Student exercises

Perform the exercises for this unit.

Summary

- List the types of available workstations
- Describe the functions of workstations in Tivoli Workload Scheduler for z/OS
- Create workstation definitions in Tivoli Workload Scheduler for z/OS

© Copyright IBM Corporation 2015

Summary

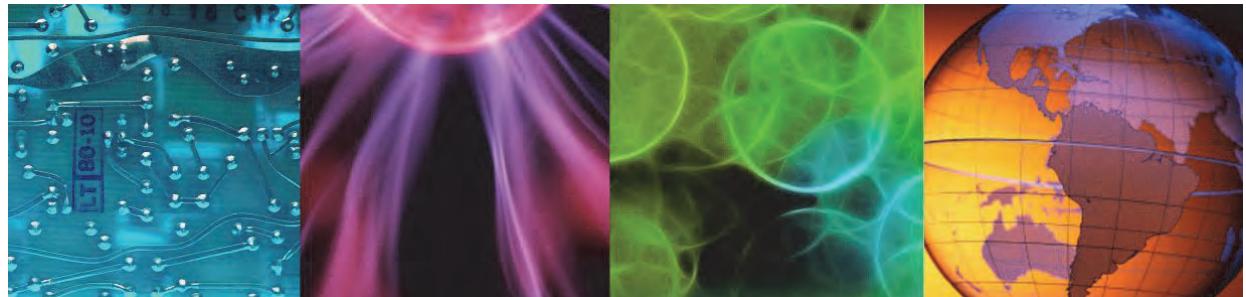


3 Calendars, periods, and run cycle groups

IBM Tivoli Workload Scheduler for z/OS 9.2.0



3 Scheduling objects: Calendars, Periods, and RunCycle groups



© Copyright IBM Corporation 2015

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

In this unit, you learn about Tivoli Workload Scheduler for z/OS calendars and calendar periods. You learn to describe calendar and period functions, and how to create them.

Reference: SC32-1262 *IBM Tivoli Workload Scheduler for z/OS Managing the Workload*

Objectives

- Describe the purpose of calendars in Tivoli Workload Scheduler for z/OS
- Create calendars
- Describe the function of periods
- Create periods
- Describe the function of run cycle groups
- Create run cycle groups

© Copyright IBM Corporation 2015

Objectives

In order to schedule batch jobs to run Tivoli Workload Scheduler for z/OS needs to know about the days of the week and the run frequency of the work. This unit shows how periods and run cycle groups are combined with calendar definitions, to meet the both the simple and complicated scheduling requirements of todays batch. Examples of all three elements are built during the hands-on exercises, continuing the construction of the scheduling environment that will be used throughout this course.

Lesson 1 Introduction to calendars

Lesson 1 Introduction to calendars

- Calendars define two type of days
 - Work days
 - Free days
- Calendars are a basic component of the scheduling data
 - One of several data base objects used in conjunction to determine when work will run

© Copyright IBM Corporation 2015

This lesson reviews the use of calendars in Tivoli Workload Scheduler for z/OS.

After completing this lesson, you should be able to perform the following tasks:

- Install Tivoli Storage Manager Administration Center.
- Install Tivoli Storage Manager server.

References: SC32-1262 *IBM Tivoli Workload Scheduler for z/OS Managing the Workload*

The **calendar** specifies the usual processing days, weekends, and public holidays that are recognized by a business. The processing days are **workdays** and the others are **free** days:

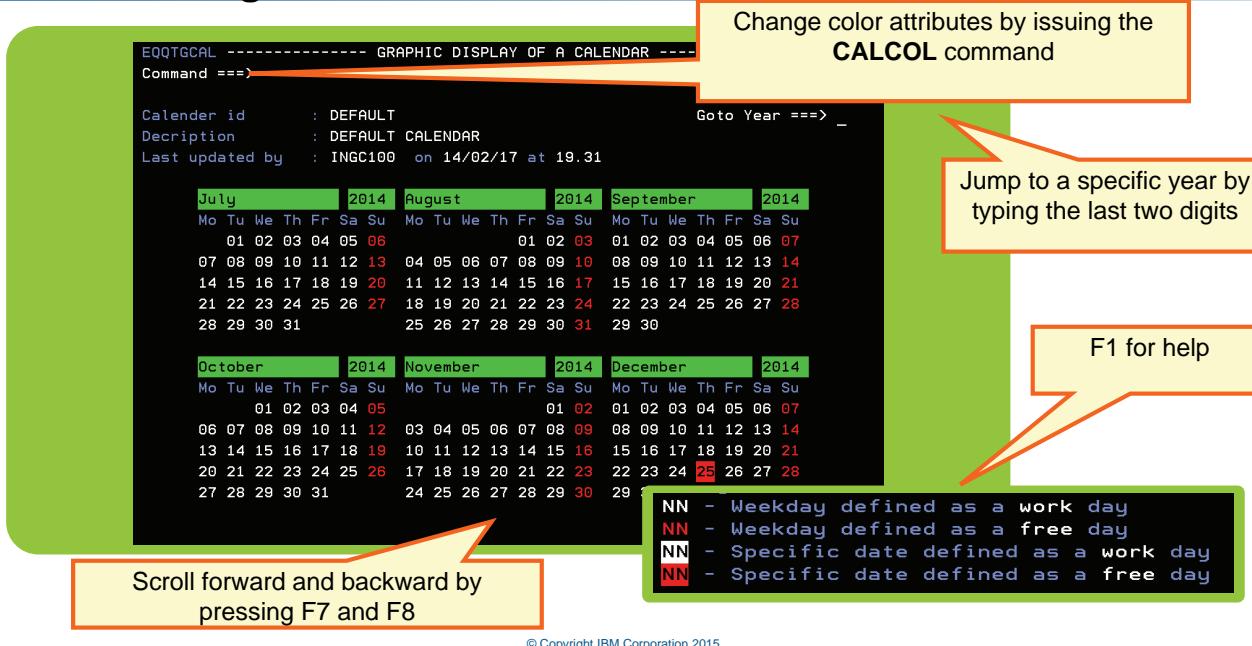
- Workdays are days when the normal batch for the business is run. These days are typically Monday through Friday, which historically are the usual days for doing business.
- Free (nonworking) days are days when you do not process some or all of the work in the usual processing schedule.

You can define as many calendars as are needed to describe the work patterns your organizations has, for example where systems span geographical areas with different public holidays. It is not only public holidays that require different calendars, business units might also have different working and non-working days from each other, which and be described by calendars.

Although you can have several calendars, be sure that you know which is the default calendar that is used when none are explicitly specified. It is sensible to call this calendar DEFAULT. Running Extends or Modifies of the Long-Term Plan in batch, the default calendar is specified in the CALENDAR parameter of the BATCHOPTS initialization statement.

You should also define a default calendar to be used by your dialog session when you request scheduling calculations. Without this it assumes that all days are working days. Use option 0 from the main Tivoli Workload Scheduler menu followed by option 2, date, to set your default calendar.

Calendar usage



Calendar usage

The scheduler uses the calendar to detect when applications are scheduled and to calculate dates for automatic JCL tailoring. The primary purpose of the scheduler calendar is to specify the workdays and the non-workdays. Non-workdays are also called free days. The scheduler typically schedules jobs to run on workdays.

This slide is an example of a calendar that is shown graphically. Use Tivoli Workload Scheduler for z/OS ISPF option 1.2.1 and the **G** row command.

The dates for Monday through Friday are shown in white and represent workdays. The dates for Saturday and Sunday are free days and are shown in a different color.

The long-term planning process uses the calendar with user-specified scheduling definitions called **run cycles**. The scheduler processes this information to determine the days that an application is scheduled.

You specify the calendar in the application or run cycle group definition. If no calendar is specified, Tivoli Workload Scheduler for z/OS locates a calendar in this sequence:

1. The defined default calendar.
2. The calendar that is used for online services, such as testing a rule with GENDAYS. You specify this calendar name on the option **0.2** panel.
3. A calendar with the name **DEFAULT**.
4. If the **DEFAULT** calendar does not exist, all days are considered workdays.

Creating calendars

Weekday or date (yy/mm/dd)	Comments	Status
Sunday	Sunday is usually a free day	F
Monday	Monday is usually a work day	W
11/12/25	December 25 th is a holiday	F
11/03/05	March 5 th is a working day for inventory control	W

- ❖ The FREE DAY RULE determines how Applications and Run-cycle groups are scheduled in relation to free days

© Copyright IBM Corporation 2015

Creating Calendars

This slide shows the types of detailed calendar data you provide to create a calendar. Use the Tivoli Workload Scheduler for z/OS ISPF option **1.2.2** and issue the CREATE command. Enter detailed calendar data in the EQQTCCAL ISPF panel.

- Specify the workday end time as **00.00**. A work day can end at a different time. For example, entering 02.00 as the end time means that the production day for Monday starts at 2:00 A.M. and ends at 1:59 A.M. on Tuesday.
- Specify each weekday workday with a status of **W** (workday).
- Specify each nonbusiness day with a status of **F** (free day).
- Override the status of a day by using a specific date. Dates take precedence over generally defined days of the week.

Lesson 2 Periods

Lesson 2 Periods

Cyclic period

- Starts on a specific date and restarts the interval after the specified number of days, continuously repeating
- 2 Types
 - Only workdays (result depends on the calendar used in the Application or Run-cycle Group)
 - All days

Noncyclic period

- Has a varying interval length
- Specifies the origin date of each interval
- Each interval has an end date
 - Implied by the start of the next interval
 - Explicitly defined

© Copyright IBM Corporation 2015

This lesson reviews the use of periods. After completing this lesson, you should be able to describe the use of calendar periods.

References: SC32-1262 *IBM Tivoli Workload Scheduler for z/OS Managing the Workload*

A period is a user-defined template that represents a business processing cycle for one or more of its scheduler applications. The period database holds definitions of business cycles for scheduling work

Periods were the original scheduling mechanism in Tivoli Workload Scheduler for z/OS. The need to define periods for many of the normal scheduling scenarios such as monthly, 1st Friday, etc., was greatly reduced with the introduction of rules. These provided a simple way to select common scheduling requirements for each application. With the introduction of Run cycle groups in release 9.1, even those patterns that were not possible with rules are simple to create. Now the main use of Periods is likely to be related to scheduling run dates that have no regular pattern.

Periods can either be **cyclic**, such as a weekly or 28-day period, or **noncyclic**, such as a month or an academic semester. Cyclic periods are created using an origin date and a cycle length.

Noncyclic periods, by the origin date of each interval and, optionally, an end date for each interval.

When scheduling an application to run, each of its run cycles can point to a period (with or without an offset), a rule, or a run cycle group to determine when it should run.

The long-term planning procedure uses the calendar information, period definitions, run cycle groups and free day rules that are defined in the applications run cycle to determine the days on which the applications are scheduled.

When work is scheduled to run

Different applications run on different cycles

- Ordering applications occur daily
- Sales applications occur weekly
- Payroll applications occur semimonthly
- Inquiries applications occur on each fiscal end of the month
- Accounting applications occur monthly and quarterly
- Taxation applications occur annually
- These cycles are called ***business processing cycles***

© Copyright IBM Corporation 2015

When work is scheduled to run

Many applications are run on a daily, weekly, or monthly basis. Others are run at less frequent intervals. Some run on an irregular basis. They might run once a week for the first quarter and twice a week in the next quarter. They can run three times a week in the third quarter and four times a week for the last quarter.

Most production work runs on regular schedules. The scheduling rules provide a way to define run schedules for applications.

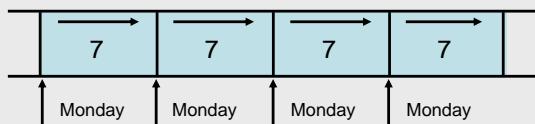
You can define most common application schedules by using rules in an application run-cycle or in a run-cycle group. For unusual requirements, a user-defined period might be needed.

Rules or a combination of user-defined periods and specified run days can define the run frequency of an application. Run frequencies are known as ***run cycles***. An application can have more than one run-cycle.

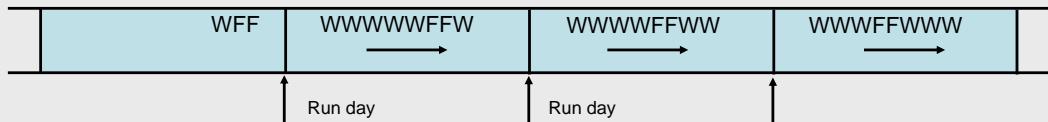
Examples of periods

Cyclic periods

Week

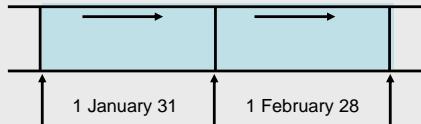


Every 6th work day



Noncyclic periods

Month



© Copyright IBM Corporation 2015

Examples of periods

The two types of periods are cyclic and noncyclic:

- **Cyclic:** Periods of a constant length, such as a week or 4 weeks (28 days) or even just a single day.
- **Noncyclic:** Periods of variable length, such as a calendar month, year, or user defined run dates with no discernible pattern.

The interval start dates for cyclic periods are calculated when the period is used. This can be when running the GENDAYS command or by a Long-Term Plan batch job (Extend, Modify, or Create). At this point the free days in the calendar are associated with it. The intervals starts counting from the period origin date for the number of days that are specified as the cyclic length then resets the counter. Each time the number is reached, an interval start date is found. The application run date is calculated in the run cycle to be this date, or an offset from it.

With a non-cyclic period, you can see the list of dates that start each interval. Noncyclic periods are ideal templates for unique, specialized business processing cycles. A good example is a month. Because months have varying lengths, you must specify the origin date of each interval. However, you do not need to specify the end date as that is implied by the next interval start date. A university might like to process work in relation to the end of each academic term. In this case, both the interval start and end dates are required as they are specific dates.



Note: The interval start dates do not specify the dates jobs run. A Period is used by an application run cycle or a run cycle group with an offset to calculate a run day. To run an application on the 21st of the month you can use a period with interval start dates of the first of each month, and apply a positive offset of 21.

Using period offsets

Offsets can be positive or negative

OFFSET = 4

Period	WEEK	M T W T F S S
Period	MONTH	1 2 3 4 5 6 7

OFFSET = -1

Period	MONTH	25 26 27 28 29 30 31 1 2
--------	-------	---------------------------------

© Copyright IBM Corporation 2015

Using period offsets

Offsets specify the actual run days for the application. Offsets are relative to the first day of the period interval for positive offsets, and relative to the last day of the period interval for negative offsets.

An offset of **1** indicates that the application runs on the first day of the period. An offset of **4** indicates that it runs on the fourth day of the period.

Negative offsets count back from the last day of the period, so **-1** specifies the last day of the period. Negative offsets are useful for scheduling work to run on the last day of the month without knowing the length of the month.

Use the Tivoli Workload Scheduler for z/OS ISPF option **1.3.2** and issue the CREATE command. To define a **cyclic period**, specify the following fields:

- The period name
- The type:
 - **A** includes all days
 - **W** includes workdays only
- The period length: **Interval**
- The start date for the period: **Interval origin date**

To define a **noncyclic period**, specify the following fields:

- The period name
- The type: **N**
- An interval of **0 (zero)**
- The interval start date: **Interval Origin Date**
- The interval end date: **Interval End Date** (Optional)

Lesson 3 Run cycle groups

Lesson 3 Run cycle groups

```
EQQODBSP ----- MAINTAINING TWSZ DATA BASES -----
Option ===> _

Select one of the following:

1 WS           - Work station descriptions
2 CALENDAR     - Calendar descriptions
3 PERIOD       - Period descriptions
4 AD           - Application descriptions
5 OI           - Operator instructions
6 SPECRES      - Special resource descriptions
7 EDWA         - Event driven workload automation
8 JD           - Job descriptions
9 JCLVAR        - JCL variable tables
10 RUN CYCLE   - Run cycle groups
```

- Defined using ISPF option 1.10
- A run cycle group is a distinct database object
- Multiple applications can use it
- It generates a list of dates that behave like the interval start dates of a non-cyclic period

© Copyright IBM Corporation 2015

In this lesson, you learn how to define a run cycle group. After completing this lesson, you should be able to define a run cycle group.

To schedule an application, you define its run cycles. Until TWS for z/OS 9.1 the application specified the calendar to use and then each of its run cycles determined which *period* or individually defined *rules* to use. An application run cycle could be either positive (when to run) or negative (when not to run) with the negative always winning if the date and time collided.

The introduction of run cycle groups means that commonly used rule definitions can be created once as a run cycle group and then referenced by any applications run cycle. A run cycle group is a list of rules that are combined together to produce a set of run dates.

Specifying run cycle group list criteria

The screenshot shows two panels from the Tivoli Workload Scheduler command-line interface.

The top panel is titled "EQQRSEL P ----- SPECIFYING RUN CYCLE GROUP LIST CRITERIA -----". It displays the following command:

```
Command ===>
```

Specify selection criteria below and press ENTER to create a RG list.

Run Cycle Group:

RUN CYCLE GROUP ID	==> _____
OWNER ID	==> _____
CALENDAR ID	==> _____
VARIABLE TABLE	==> _____

Specific Run Cycle in Group:

RULE NAME	==> _____
CALENDAR ID	==> _____
VARIABLE TABLE	==> _____
SUBSET ID	==> _____

A red arrow points from the text "Press Enter" down to the bottom panel.

The bottom panel is titled "EQQNRLSL LIST OF RUN CYCLE GROUPS". It displays the following command:

```
Command ==> create_
```

View: Compact (EQQNRLST) Row 1 of 0 >>
Row Run cycle group description Tot run
cmd ID cycles

***** end of data *****

Run the CREATE command to define a new run cycle group

© Copyright IBM Corporation 2015

Specifying run cycle group list criteria

To list run cycle groups, do the following:

1. In the MAINTAINING TWSZ DATA BASES panel, select option 10 RUN CYCLE. The SPECIFYING RUN CYCLE GROUP LIST CRITERIA panel is displayed.
2. To get a complete list of defined run cycle groups, leave all fields blank and press ENTER. The LIST OF RUN CYCLE GROUPS panel is displayed. It contains a complete list of all the run cycle groups that are currently defined in the Tivoli Workload Scheduler for z/OS database.
3. To get a filtered list, enter information that leads to the selection of the run cycle group (or groups) you are seeking and press ENTER. Entering more specific information produces narrower results. To get wider lists, you can also use the asterisk and percent signs as masking characters.

From the LIST OF RUN CYCLE GROUPS panel, you can run the CREATE command to create a new run cycle group.

Creating a run cycle group

```
EQQRCRGL ----- CREATING A RUN CYCLE GROUP ----- Row 1 of 2
Command ==> _
Scroll ==> CSR

Enter/change data below and in the rows,
and/or enter any of the following row commands:
I(nn) - Insert, R(nn),RR(nn) - Repeat, D(nn),DD - Delete
S - Specify run days/Modify rule
Enter the GENDAYS command to show the dates generated by this Run cycle group

RUN CYCLE GROUP ID ===> PRDGRO1 Run cycle group name
DESCRIPTION ===> Prod run group 1
OWNER ID ===> TEAM 1 Owner name
INPUT HH.MM ===> 06.00 Input arrival time
DEADLINE DAY/TIME ===> 00 / 08.00 Deadline
CALENDAR ID ===> DEFAULT Calendar name
VARIABLE TABLE ===> GVARTABL JCL variable table id

Row Name of Input Deadline F day In Out of
cmd rule HH.MM day HH.MM Type rule YY/MM/DD YY/MM/DD Variable table
... RULE1 06.00 00 07.00 R 4 14/02/25 71/12/31
Description: _____
Subset ID: _____ Calendar: _____
... RULE2 06.00 00 08.00 R 4 14/02/25 71/12/31
Description: _____
Subset ID: _____ Calendar: _____
```

A run cycle group is a combination of rules collected into one or more subsets

© Copyright IBM Corporation 2015

Creating a run cycle group

Defining run cycle groups (RCG)

Fields

- Run cycle group ID: A name for the run cycle group (required)
- Description: Optional field to describe the total run cycle group
- Owner id: Optional, but often used to secure the run cycle group
- Input arrival (optional if the rules will define their own, else it provides a default value for the rules)
- Deadline day and time (optional)
- Calendar (optional) if blank will use the default calendar in the dialog for running the GENDAYS command and the default calendar specified in the BATCHOPTS statement.
- Variable table (optional) a table to associate with this RCG
- Define rules for the RCG by selecting the rule with the **S** row command
- Delete a rule with the **D** row command
- Copy (repeat) a rule with the **R** row command

© Copyright IBM Corporation 2015

Run cycle group fields

Defining rules in run cycle groups

Fields

- Rule name (required and must be unique in the run cycle group, RCG)
- Input arrival (required, but may default to the run cycle group value)
- Deadline day and time (optional)
- Rule type (R, E, A, D)
- Free day rule (1, 2, 3, 4, E)
- In effect (the first date calculated will not be before this date)
- Out of Effect (the last date calculated will be before this date)
- Variable table (optional) a table to associate with this rule in the RCG
- Description of the rule
- Subset ID collects together some of the rules within the RCG
- Calendar (optional) used if this rule should consider a different calendar

© Copyright IBM Corporation 2015

Defining rules in run cycle groups

Run cycle group rule types and sub-sets

Name of rule	Input HH:MM	Deadline day HH:MM	Type rule	In effect YY/MM/DD	Out of Effect YY/MM/DD	Variable table
RULEA	R	4	14/09/20	71/12/31		
Description: Every Friday of the Year						
Subset ID:						
RULEB	A	4	14/09/20	71/12/31		
Description: Only Last Day of Month						
Subset ID:	SB					
RULEB2	A	4	14/09/20	71/12/31		
Description: Only the last workday of the Month						
Subset ID:	SB					

January 2015	February 2015	March 2015
Mo Tu We Th Fr Sa Su	Mo Tu We Th Fr Sa Su	Mo Tu We Th Fr Sa Su
01 02 03 04	02 03 04 05 06 07 08	01 02 03 04 05 06 07 08
05 06 07 08 09 10 11	09 10 11 12 13 14 15	09 10 11 12 13 14 15
12 13 14 15 16 17 18	16 17 18 19 20 21 22	16 17 18 19 20 21 22
19 20 21 22 23 24 25	23 24 25 26 27 28	23 24 25 26 27 28 29
26 27 28 29 30 31		30 31
April 2015	May 2015	June 2015
Mo Tu We Th Fr Sa Su	Mo Tu We Th Fr Sa Su	Mo Tu We Th Fr Sa Su
01 02 03 04 05	01 02 03	01 02 03 04 05 06 07
06 07 08 09 10 11 12	04 05 06 07 08 09 10	08 09 10 11 12 13 14
13 14 15 16 17 18 19	11 12 13 14 15 16 17	15 16 17 18 19 20 21
20 21 22 23 24 25 26	18 19 20 21 22 23 24	22 23 24 25 26 27 28
27 28 29 30	25 26 27 28 29 30 31	29 30

© Copyright IBM Corporation 2015

Run cycle group rule types and sub-sets

Select, or insert, individual rules to create the run cycle group.

Values in the fields will be carried through to the applications that use the run cycle groups.

- **Deadline:** (optional since TWS 9.1)
 - **Day** can be up to 99 days from the date when the application is added to the current plan.
 - **Time** is the time when all operations in the application occurrence must complete.
- **Effective dates of run cycle:** Lifespan for inclusion in the long-term plan.
 - **In effect:** Defaults to the creation date.
 - **Out of Effect:** Default is the TWS high date - currently 31st December 2071
- **Type of rule:**
 - R - When to run and logical “OR” to the other rules.
 - E - When not to run applied to the other rules.
 - A - When to run and a logical “AND” to the other rules.
 - D - When not to run applied as a logical “AND” to other rules.
- **Subset ID:** a free form name that is used to collect some rules into a subset. Allows negative rules to be applied only to the subset of positive rules not to **all** the rules.

You can define multiple rules for each run cycle group. Each rule is calculated alone or as part of a subset and the combined result is a set of dates that can then be used to schedule the workload.

Free day rules 1, 2, 3, and 4

Rule 1	Su	Mo	Tu	We	Th	Fr	Sa
	1	2	3	4	5	6	7
	8	9	10	11	12	13	14

Rule 2	Su	Mo	Tu	We	Th	Fr	Sa
	1	2	3	4	5	6	7
	8	9	10	11	12	13	14

Rule 3	Su	Mo	Tu	We	Th	Fr	Sa
	1	2	3	4	5	6	7
	8	9	10	11	12	13	14

Rule 4	Su	Mo	Tu	We	Th	Fr	Sa
	1	2	3	4	5	6	7
	8	9	10	11	12	13	14

- Consider a run cycle that selects every Monday
- Saturdays, Sundays, and the 9th are free days in the selected calendar
- Each free day rule moves the occurrence scheduled for the 9th to a different day

© Copyright IBM Corporation 2015

Free day rules 1,2,3, and 4

Tivoli Workload Scheduler for z/OS uses free-day rules to handle scheduling on days that are defined as free days in the calendar. Each run cycle group and each rule can have an associated calendar. You define one of five free-day rules for each rule as follows:

- 1: Move the date to the closest workday *before* the free day.
- 2: Move the date to the closest workday *after* the free day.
- 3: Keep on the free day.
- 4: Drop the free day (default value if not defined).

Free day rule E

No Rule	Su	Mo	Tu	We	Th	Fr	Sa
	1	2	3	4	5	6	7
	8	9	10	11	12	13	14

Rule E	Su	Mo	Tu	We	Th	Fr	Sa
	1	2	3	4	5	6	7
	8	9	10	11	12	13	14

- Consider a run cycle that selects a MONTH period with an offset +9
- Saturdays, Sundays, and the 9th are free days in the selected calendar
- For run cycles that use periods and offsets, free day rule E means “use only work days to compute the offset”

© Copyright IBM Corporation 2015

Free day rule E

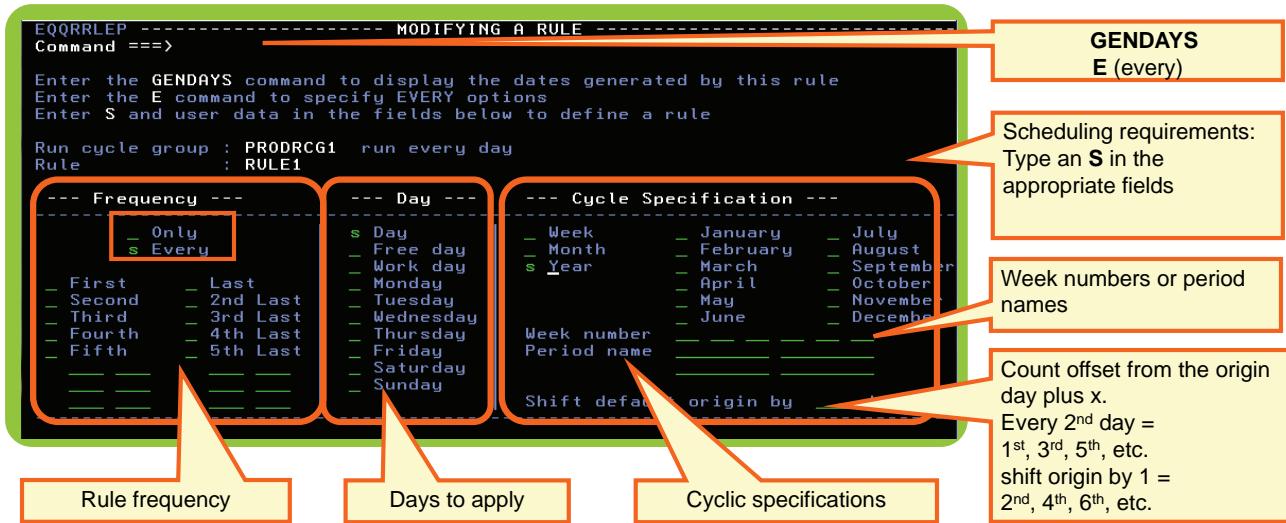
Free day rule E is a special case for run cycles that use periods and offsets.

When free day rule 1,2,3,or 4 is in effect, the scheduler counts **all** days when offsetting from the interval start date in the period. The date is then checked against the calendar to see if it has landed on a free day. When it is a free day the free day rule used comes into effect.

When free day rule E is used, Tivoli Workload Scheduler for z/OS only counts work days. It skips the free days in the calendar assigned to the application. Using a MONTH period with an offset and free day rule E is a way to schedule work on the *n*th **work day** of every month.

Specifying run cycle rules

Select a rule with the **S row** command on the RUN CYCLE panel



© Copyright IBM Corporation 2015

Specifying rules

Select a run frequency for **Every** or **Only** when you specify your rule. You can also select an ordinal number from **First** to **Fifth** and 6 - 999 or its **Last** equivalent. Enter values higher than Fifth or 5th Last in the blank fields below Fifth or 5th Last. You can make multiple selections. If you select **Only** or **Every** and make no other selections, dates are generated as if you had selected **First**.

In the Modifying a Rule panel, you define scheduling requirements by selecting at least one item from each of the following three columns:

- **Frequency**: Select **Only** or **Every**. **Every** implies running at a repeated interval that is defined by the numeric selection in **Frequency**.
- You can also select from or specify numbers in the **First** and **Last** lists.
- **Day**: Make one or more selections to specify the run days for the application.
- **Cycle Specification**: Select from lists of Tivoli Workload Scheduler for z/OS-supplied cycle templates or patterns. Use them to define the processing cycle for business applications.
- You must make one or more unambiguous selections in this column. Selecting **Month** and one of the listed months is an ambiguous selection. The **Week** cycle in this column starts on **Monday**. You can specify a user-defined period if the listed cycles do not meet your needs. You can shift the start day of a cycle by using the **Shift default origin** date field.

Using the Every option

RC1 09:00	Su	Mo	Tu	We	Th	Fr	Sa
	1	2	3	4	5	6	7
RC1 10:00	8	9	10	11	12	13	14
RC1 11:00	Su	Mo	Tu	We	Th	Fr	Sa
	1	2	3	4	5	6	7
RC1 12:00	8	9	10	11	12	13	14

- Create multiple occurrences in one panel
- Repeat the run cycle on equal intervals
- Repeat time
 - From (start time)
 - Until (end time)

Example:

Repeat every 01:00

- From 09:00
- Until 12:00

© Copyright IBM Corporation 2015

Using the Every option

By using the **Every** option, you define a repeating rule from one panel. You can schedule an application to run several times a day without manually creating individual run cycles for each occurrence. The slide shows how run cycle RC1 repeats four times, by using the **every** option.

The **Every** option is processed by the long-term plan to create multiple occurrences. Manually adding an application to the current plan does not create multiple occurrences based on run cycle **Every** options that are defined for the application.

Use the **E command** in the Modifying a Rule panel to specify the **Every** options.

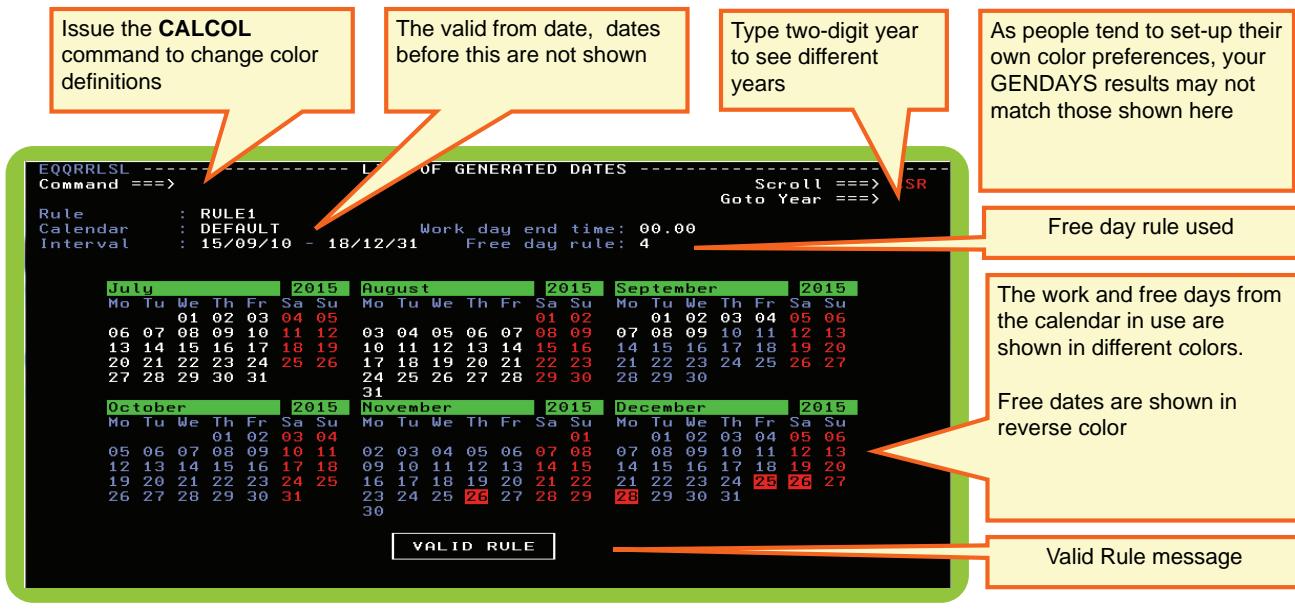
```
EQQRMREP ----- EVERY OPTIONS -----
Command ==> _

REPEAT EVERY ==> _____           Repeat every HH.MM
FROM          ==> 00.00           Input Arrival time
UNTIL         ==> _____           Repeat end time in the HH.MM format
```

The field descriptions for the Every Options panel are as follows:

- FROM: Shows the input arrival time of the run cycle that identifies the repeat start time.
- REPEAT EVERY: An input field that specifies the repeat time interval in the *HH.MM* format.
- UNTIL: An input field that specifies the repeat end time in the *HH.MM* format. It must be a value between the input arrival time and the calendar workday end time.

Using the GENDAYS command



Using the GENDAYAS command

After selecting frequency, day, and cycle specifications, you can enter the GENDAYS command to show a calendar with the run days generated for the rule. This slide shows an example of the GENDAYS results.

If no calendar is specified in the rule or the run cycle group, the GENDAYS function uses the calendar that is specified on the option **0.2** panel. The GENDAYS function presents the run days for each rule at a time. When on the Run cycle group panel GENDAYS shows the combined effect of all the rules.

The GENDAYS command also notifies you if the **Every** option was specified. It also shows the number of repeat occurrences for each day.



There are multiple advantages in using run cycle groups.

- A run cycle group is a distinct database object. It is defined by itself and referenced by one or more applications.
- The same run cycle group can be used by different applications. This improves the overall usability of the run cycles, since it is possible to specify the same run cycle group in multiple

applications which avoids the need to define the same scheduling requirement on multiple applications.

- They provide more functions, like being able to ‘and’ two or more rules together. It is possible to find the dates where Monday is also the 15th of the month, or Wednesday is also a free day. This is achieved with the addition of ‘subsets’. Previously this required building a manual period. They also enhance the use of negative run cycles by limiting the negative effect to just rules in the same sub set. The addition of two new run cycle types has added this flexibility:
 - A: Calculate positive dates within a subset and apply a “*logical and*” to other rules in the same subset.
 - D: Calculate negative dates within a subset and apply a “*logical and*” to other rules in the same subset.
 - For example, you can add multiple conditions together, such as in Run on Monday “AND” a free day “AND” not the last day of the month. In this way, the only scheduled dates would be holiday Mondays but not the last day of the month.
 - In the previous point you got a set of dates from a subset that would match a free day that was a Monday but not the last day of the month. A second subset might be created that discovered days that were, the last day of the month, and a free day, and a Monday, but points to a different Variable table, enabling different JCL or parameter values to be passed into the jobs on those days.
- They enable the equivalent of automating the maintenance of periods, where the periods have some pattern that can be associated with them.
- Run cycle groups can reference a manually created period.
- Because a run cycle group can have the same name as a period there is a hierarchy between periods and run cycles. A matching period name is used before a run cycle group making it easy for installations that use periods to replace them with run cycle groups in a controlled way.
- Run cycle groups and periods are referenced by application run cycles in the same way. The application can run on the dates that are calculated by the run cycle group (or interval start days from a period) or the dates provide a reference point for the use of shift or offset.

With the previous example, you calculated Mondays that were free days and were, or were not, the last day of the month. By referencing this run cycle group you can shift an applications run days, forward or backward to the day you like it to run on. Shift 1 WD (workday) would run the application on the workday following holiday Mondays.

- Even more flexibility is introduced by allowing the run cycle group to use multiple calendars to calculate its dates and then have the application use a different calendar.

Looking back to your free day and Monday example, you can imagine the calendar that is used to determine free days is for one country (UK), but the application that references it is for another country, with different free days in its calendar. The shift that is used by the application

is not dependent on the run cycle group calendar, but on its own calendar that might alter when the jobs associated with that application run.

- The GENDAYS command can be run at run cycle group level. In this way, you can check the result of the combination of all the run cycles in the group plus each individual rule.
- In the long-term plan (LTP), run cycle groups are interpreted as non-cyclic periods with open intervals (no end dates). The generated days are used as start dates for the intervals and every interval that starts from a generated day, finishes at the next generated day. To ensure the proper operation of the run cycle group, the last interval should extend beyond the LTP end-date, so its interval is closed by this date.
- You can use the Dynamic Workload Console or the ISPF panels to define and manage run cycle groups. These notes document the ISPF interface. The documentation about the equivalent actions on the Dynamic Workload Console is available in the online help of the console.

Student exercises



See the Student Exercise Guide Unit 3

© Copyright IBM Corporation 2015

Student exercises

Perform the exercises for this unit.

Summary

- Describe the purpose of calendars in Tivoli Workload Scheduler for z/OS
- Create calendars
- Describe the function of periods
- Create periods
- Describe the function of run cycle groups
- Create run cycle groups

© Copyright IBM Corporation 2015

Summary

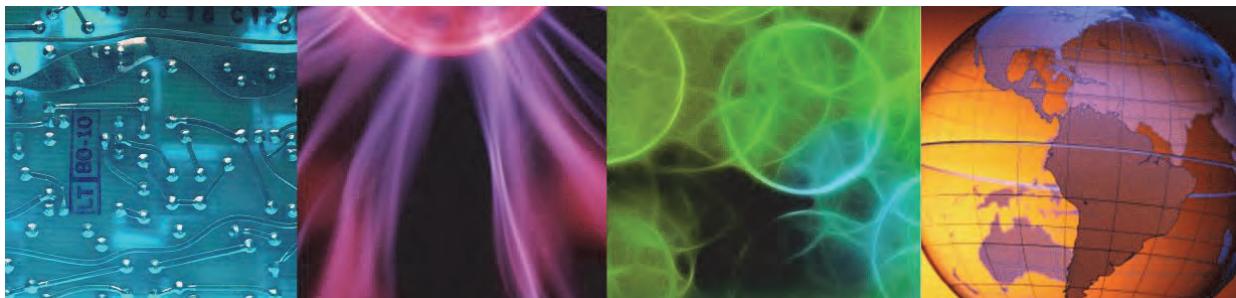


4 Applications

IBM Tivoli Workload Scheduler 9.2.0



4 Applications



© Copyright IBM Corporation 2015

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

In this unit, you learn how to create and schedule Tivoli Workload Scheduler for z/OS applications, application groups, and job descriptions.

References: SC32-1262 *IBM Tivoli Workload Scheduler for z/OS Managing the Workload*

Objectives

- Describe the parts of an application
- Describe the types of applications
- Create applications with or without run cycles
- Create operations within an application
- Create a job description

© Copyright IBM Corporation 2015

Objectives

We now know when to run work and where to run work. This unit explains how to define the jobs and their inter-relationships. In the hands-on exercise a small network of jobs is defined and scheduled to run.

Lesson 1 Applications and operations

Lesson 1 Applications and operations

Application descriptions

- Collection of operation definitions
- Scheduled based on run cycles
- Stored in a VSAM database called the *AD*
- Main parts
 - General information
 - Up to 255 operations
 - Run cycles
- Can be scheduled together in application groups

Operations

- Defined in an Application or Job Description.
- Describes a task to be scheduled
- Associated with a workstation which determines the type of task
 - CPU operation
 - Dummy Task
 - WTO to the system
 - Remote Shadow job
 - zCentric Operation
 - etc...
- Numbered uniquely within each application
- Workstation ID + operation number is the Operation ID

© Copyright IBM Corporation 2015

This lesson provides an overview of applications and operations. After completing this lesson, you should be able to describe applications and operations at a high level.

A Tivoli Workload Scheduler for z/OS **Application Description (AD)** is a set of related jobs or tasks called **Operations**. Application Descriptions range in size and complexity, from a single computer operation with no dependencies, to a set of tasks at various types of workstations, with complex dependencies both internally, within the application description, and externally with other applications.

An application is the basic unit of work that you can schedule in Tivoli Workload Scheduler for z/OS. An application can hold a maximum of 255 operations. Most of tasks that you want to control are operations that represent real jobs, that is: z/OS-based JCL jobs or tasks that run in the end-to-end environment on distributed servers (Linux, UNIX, Windows, and so on).



Note: As the AD is the scheduled element, all the operations in an application must want to run with the same frequency.

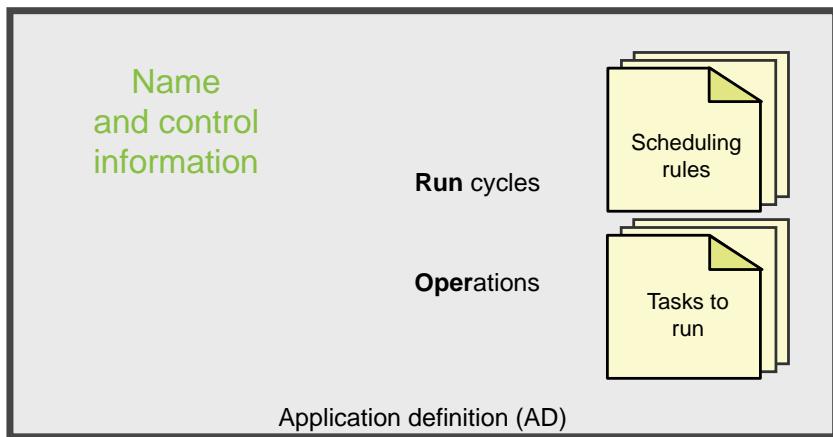
Every task in the application has an associated **operation number** (a unique identification number between 001 and 255) and a specific workstation. The workstation tells the operation what type of task it is and where it is processed. The operation numbers are used to define the dependency links

between the operations in an application. Tivoli Workload Scheduler for z/OS runs the operations in the sequence that is dictated by the defined dependency relationships, that is, predecessors followed by successors. Together the operation number and workstation form the operation id.



Note: The operation number is not a sequence number, for instance, operation number 010 can depend on operations 025 and 005.

Application structure



© Copyright IBM Corporation 2015

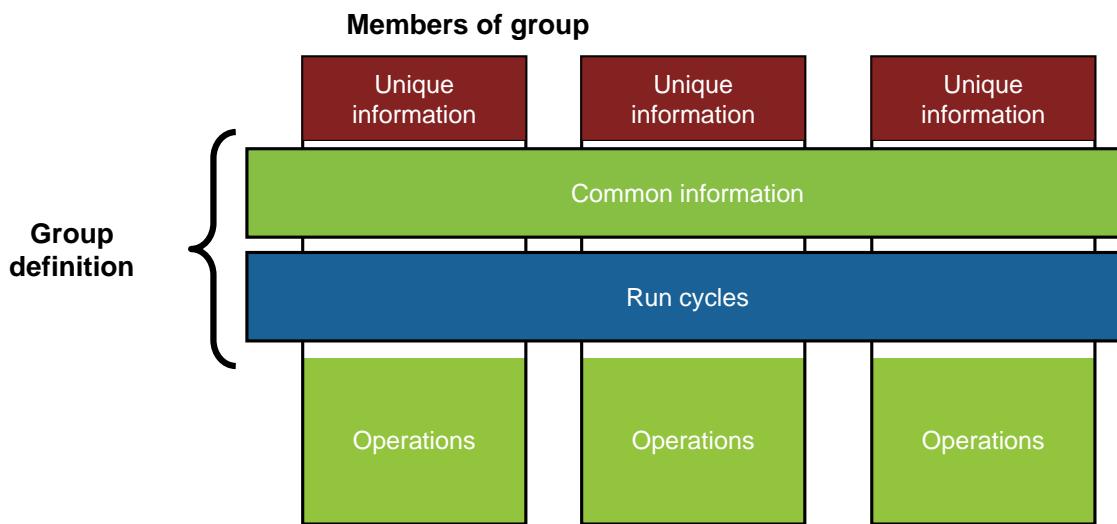
Application structure

The three basic segments of an Application Description (AD) are as follows:

- **General (Common) Information:** Contains the application name and other application control information, such as the calendar, owner ID, and submission priority.
- **Run Cycles:** Contains the scheduling rules that define when the application runs.
- **Operations:** Contains the jobs, tasks, and other activities that make up the batch work.

You use the Tivoli Workload Scheduler for z/OS ISPF panels to define applications. Every AD has a general information panel that names the AD and provides data relevant to all the operations contained within it, such as Owner and Authority Group ID. For scheduling, the AD is either part of an Application Group or has its own run cycles. Issue the **run** command to build the run cycles for the AD or provide it with the Application Group name. Use The **oper** command to define the operations. Each operation can have details such as dependencies, special resources and, time dependencies, defined.

Application groups



© Copyright IBM Corporation 2015

Application groups

You can group applications and job descriptions that are always scheduled together by creating them as members of an **application group**. You can create and maintain a group definition with run cycle information that supports multiple member applications. Or you can create and maintain run cycles for multiple individual applications. You can also use groups in the Modify Current Plan panel to add, delete, and complete all or part of an application group.

Applications in a group are protected against accidental deletion or modification of occurrences in the current plan. You can delete or manually complete an application occurrence that is part of a group in the current plan. However, you must first remove it from the group.

The application types are application descriptions, job descriptions, and application groups. Standard application descriptions each contain between 1 and 255 tasks that you want the scheduler to manage. Applications:

- Contain the operations to be run.
- Can contain general, operations, and run cycle data segments.
- Can contain up to 255 operations.
- Can be scheduled individually.
- Can be scheduled as part of a group.

Application Groups are containers of standard application descriptions. Application Groups have characteristics as follows:

- Only have data in the general information and run cycle segments.
- Cannot contain operations.
- Are used for holding the scheduling rules for multiple applications

Job Descriptions are special Application Descriptions containing only one Computer Operation. They are described more fully later in this chapter.

The group definition in this example contains data in the general information and run cycle segments. Data in the general information segment includes the following types of data:

- Application ID
- Application type
- Owner ID
- Calendar ID

Data in the run cycle segment includes the scheduling rules for when applications in the group are selected to run. By using application groups, you avoid needing the same calendar and run policy information for each individual application.

The **application descriptions** in this example are members of the example group definition. They also contain data in the general information and operation segments that supersedes the data at the group level, such as the owner ID. Data in the general information segment of the members includes the following information:

- Application ID
- Application type
- Owner ID
- Priority
- Group definition

No data is defined in the run cycle segment of the group members. Data in the operations segment includes the operations to be scheduled.

Lesson 2 Creating applications

Lesson 2 Creating applications

- Purpose
- Launch target
- Project duration
- Dependency relationships
- Priority and criticality
- Naming conventions
- Application ownership
- Definition creations
 - ISPF panels
 - Batch loader
 - Graphical interface

Analyze and document your production job requirements in Tivoli Workload Scheduler for z/OS terms

© Copyright IBM Corporation 2015

The lesson covers how to define applications by using the ISPF dialogs. After completing this lesson, you should be able to create applications.

Before you can create application descriptions in the scheduler, you must understand your batch workload requirements. After you know your requirements, analyze them and specify them in terms that Tivoli Workload Scheduler for z/OS can use. Make the connection between your workload requirements and the way the scheduler is designed to function. Completing this training is a first step in understanding how you can use the scheduler to support your batch workload requirements.

Naming conventions are always important. If you have a large application base, it is even more important to have a well-designed naming convention.

General information for application types

	Type A	Type G
Application ID: Name of the application	Required	Required
Type: A (application) or G (group)	Required	Required
Owner ID: Name of the owner	Required	Required
Priority: 1 = low, 9 = high	Required	Optional
Valid from: Date the application becomes available	Required	Required
Status: A (active) or P (pending, draft)	Required	Required
Authority Group: Name of authority group	Optional	Optional
Calendar: Name of the free days calendar	Non group members	Optional
Group definition: Name of the application group	Group members	N/A

© Copyright IBM Corporation 2015

General information for applications

Use the Tivoli Workload Scheduler for z/OS ISPF option **1.4.2** to create applications. On this panel, EQQACGPP, you enter the general information for the application. The panel also supports the **run** and **oper** commands for defining run cycles and operations. Fields on the general information panel are as follows:

- **Application ID:** Up to 16 alphanumeric characters. The first character must be alphabetic (required).
- **Type:** **A** (application) or **G** (application group) (required).
- **Owner ID:** Up to 16 alphanumeric characters. The first character must be a letter. The owner ID might be a department, an individual, a support team. How it is used will be installation specific. The Owner field data can be used to secure access to the whole application. (required).
- **Priority:** Used by the scheduler to help determine submission priority for operations in this application relative to ones in other applications. This field is not used in group definitions. Values are 1 = low, 2–7 = medium, 8 = high, 9 = urgent. It is normal to assign a medium priority of **5** to all application descriptions unless the application has special significance (required).
- **Status:** **Active** or **Pending**. *Active* means that the application can be included in the long-term planning process or manually added to the current plan. *Pending* prevents inclusion in the LTP and adding to the CP (required).
- **Valid From:** Enter the date that the application becomes available to be scheduled. By default, the creation date of the application applies (required).

Creating an application in ISPF

```
EQQACGPP ----- CREATING AN APPLICATION -----
Command ==> -

Enter/Change data below:
Enter the RUN command above to select run cycles or enter the OPER command
to select operations.

Application:
  ID          ==> THAPPL1_____
  TEXT        ==> Team#_Application one____ Descriptive text
  TYPE        ==> A          A - Application, G - Group definition

Owner:
  ID          ==> TEAM#_____
  TEXT        ==> _____ Descriptive text of application owner

PRIORITY     ==> 5          A digit 1 to 9 , 1=low, 8=high, 9=urgent
VALID FROM   ==> 11/09/10  Date in the format YY/MM/DD
STATUS        ==> A          A - Active, P - Pending
AUTHORITY GROUP ID ==> _____ Authorization group ID
CALENDAR ID   ==> DEFAULT____ For calculation of work and free days
GROUP DEFINITION ==> _____ Group definition id
SMOOTHING FACTOR  ==> LIMIT    ==> _____ Deadline Feedback options
```

ISPF option 1.4.2

© Copyright IBM Corporation 2015

Creating an Application in ISPF

- **Calendar:** Use for calculating the run days for the application when you are scheduling this application. If you have a default calendar in your profile, it shows in the calendar field.
- **Group Definition:** Use for identifying the name of the group that you want to add the application to.
- **Text:** Use to document associated fields.
- **Smoothing Factor:** Specify a value from 0 to 999. The DSF controls how actual deadline data is used. It also determines the level of feedback of the new estimated deadline.

$$NDL = ODL + ((ADL - ODL) * DSF/100)$$

- **Feedback Limit:** Specify a value from 100 to 999. The DFL establishes the limits for the feedback data (actual deadline) to be considered normal and acceptable. Numbers outside the limits are ignored. Calculate the limits as follows:

Lower limit: $ODL * 100/DFL$ Upper limit: $ODL * DFL/100$



Note: You can create up to four versions of an application by giving a new application the same name as an existing one, but with a different **Valid From** date. Creating a fifth version causes the oldest version to be deleted. Tivoli Workload Scheduler for z/OS schedules the version of the application that has a **Valid From** date that is today or earlier.

Lesson 3 Timing workloads

Lesson 3 Timing workloads

RC1	Su	Mo	Tu	We	Th	Fr	Sa
	1	2	3	4	5	6	7
	8	9	10	11	12	13	14

RC2	Su	Mo	Tu	We	Th	Fr	Sa
	1	2	3	4	5	6	7
	8	9	10	11	12	13	14

RC3	Su	Mo	Tu	We	Th	Fr	Sa
	1	2	3	4	5	6	7
	8	9	10	11	12	13	14



Result	Su	Mo	Tu	We	Th	Fr	Sa
	1	2	3	4	5	6	7
	8	9	10	11	12	13	14

Run cycles determine when applications are to run

- Description
- Rule-based or offset-based
- Inclusive: when to run
- Exclusive: when not to run
- Input arrival time (occurrence)
- Deadline time
- Schedule around free days
- Effective dates
- Variable table usage
- Run cycles are cumulative

In this lesson, you learn how to specify when applications run and how to control the timing that is associated with the run schedule. You learn how to define a run cycle for an application. After completing this lesson, you should be able to control when applications run by using run cycles.

RUN command panel

```
EQQAMRPL ----- RUN CYCLES ----- Row 1 of 1
Command ==> Scroll ==> CSR

Enter/Change data in the rows, and/or enter any of the following
row commands:
I(nn) - Insert, R(nn),RR(nn) - Repeat, D(nn),DD - Delete
S - Specify run days/Modify rule

Application : PRDAPPL1
Name of rg/ Input Deadline F day In effect Out of Effect Variable table
Row period/rule HH.MM day HH.MM Type rule YY/MM/DD YY/MM/DD
cmd **** RULE1 07.00 00 08.00 R 4 14/02/20 71/12/31 GVARTABL
Text : Run by 8:00 am
Shift: 0 Shift Day Type: _
```

Issue the RUN command from the Creating An Application panel

Run command panel

The following information about defining run cycles is applicable to both application groups and applications. During this session, only the term *application* is used.

After you complete the general information of the application, issue the **run** command to create one or more run cycles. You can also add or amend run cycles later by modifying the application.

The application run cycles are similar in structure to the run cycle group. However, types A and D are not applicable and two more fields; *Shift* and *Shift Day Type* are available to use.

The Run Cycles panel, EQQAMRPL, contains the list of run cycles for that application. Each run cycle has a name (might match the name of a period or run cycle group), an input arrival time, a free day rule and an in and out of effect date. The type field on each run cycle determines whether it is positive or negative, whether the run days are calculated by using a rule, or offsets from dates that are defined in a period or calculated by a run cycle group. The **S** row command selects a panel to specify the scheduling details for that run cycle.

Specifying application run cycles

Use the Tivoli Workload Scheduler for z/OS ISPF option **1.4.3.0** and select an application to modify (M). Issue the **run** command to modify or define a new run cycle.

Defining run cycles

Run cycles rules

- Each run cycle has rules
- Define rules for each run cycle by selecting the run cycle with the **S** row command
- Delete a run cycle with the **D** row command
- Copy (repeat) a run cycle with the **R** row command

Fields

- Run cycle name
- Input arrival
- Deadline day and time
- Rule type (R, E, N, X)
- Free day rule (1, 2, 3, 4, E)
- First day in effect
- Day after the last day in effect

© Copyright IBM Corporation 2015

Defining run cycles

On the Run Cycles definition panel, you specify information as follows:

- **Period/RG/Rule name:** Predefined Period or Run cycle group name or a unique name for this rule in this application. The **Type** field indicates whether you are creating a rule-based or offset-based run cycle.
- **Input Arrival Time:** Time that is associated with the occurrence. Input Arrival Time is one of the fields that make up the occurrence key and is also used for dependency resolution.
- **Deadline:** (optional)
 - **Day** can be up to 99 days from the date when the occurrence is added to the current plan.
 - **Time** is the time by which all operations in the application occurrence should complete.
- **Type:** Define whether the run cycle name is a rule or a time period, and whether the run cycle is scheduling or suppressing a run.
 - **R:** A rule that is used to schedule a run (rule-based)
 - **E:** A rule that is used to suppress a run (rule-based)
 - **N:** A period or run cycle group is used to schedule a run (offset-based)
 - **X:** A period or run cycle group is used to suppress a run (offset-based)



Note: When using rules (type R or E) to define the run cycle of an application, you use the Modifying a Rule panel. Use this panel to specify the run days and processing cycle. When using a period or Run Cycle Group (Type N or X) you use the Run Days panel to specify the offsets to be used. Remember you can use both periods and Run Cycle Groups in the cycle specification of a rule definition.

- **Free day rule:** Handling of the free-day scheduling policy. Specifies the action that the Tivoli Workload Scheduler for z/OS must take when the application occurrence is scheduled on a free day
 - **1:** Move back to the closest preceding work day.
 - **2:** Move forward to the closest working day after.
 - **3:** Run on a free day.
 - **4:** Do not run.
 - **E:** Exclude free days from the calculation. Count and specify work days only when the run cycle is defined by using a numeric offset. For example, suppose you define offset 10 in a monthly noncyclic period, or the 10th day of the month in a rule, and you specify free day rule **E**. In this case, occurrences are generated on the 10th workday of the month, not the 10th day.
- **Effective dates of run cycle:** Lifespan for inclusion in the long-term plan.
 - **In effect:** Default is the application creation date.
 - **Out of Effect:** Default is the Tivoli Workload Scheduler for z/OS high date (currently 31st December 2071).
- **Variable table name** (optional). Associate a variable table for the resolution of JCL variables that are used in jobs in the application
- **Descriptive text** (optional)
- **Shift** (optional)
- **Shift Day Type** (optional)

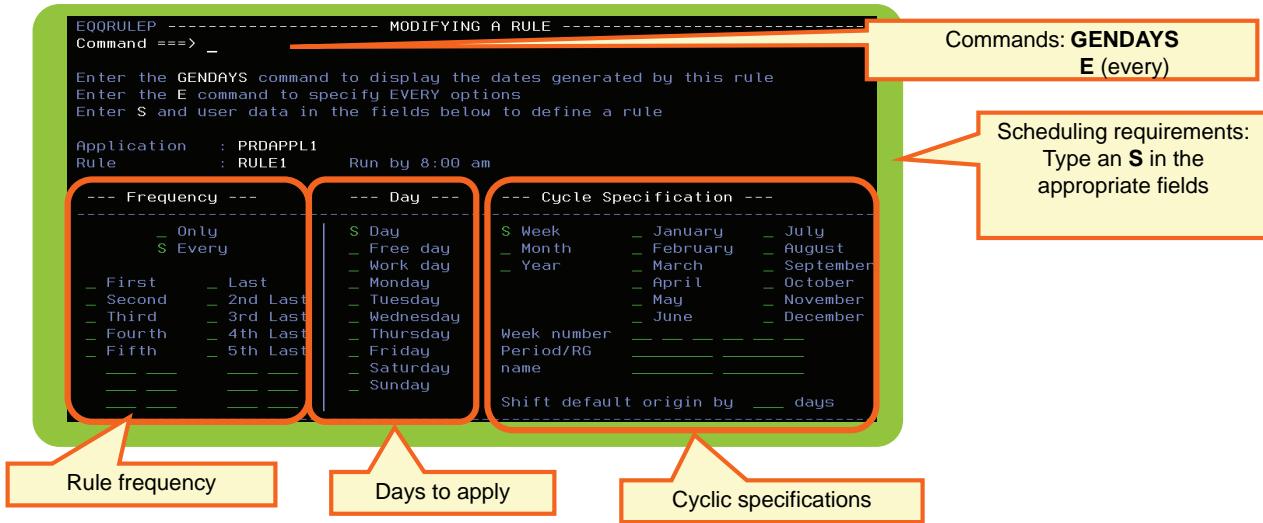


Note: The Browsing Run Cycles panel is shown when you enter the **RUN** command on the Browsing an Application general information panel. Press the F1 (Help) key from the Tivoli Workload Scheduler for z/OS Browsing Run Cycles panel to view a description of each run cycle field.

You can define multiple run cycles for an application. Each run cycle represents one or more occurrences of the application. The long-term batch planning job evaluates run cycles in a top-to-bottom sequence. The long-term planning job generates the application occurrences in the long-term plan.

Specifying run cycle rules

Select a rule with the **S** row command on the RUN CYCLE panel



© Copyright IBM Corporation 2015

Specifying run cycle rules

In the Modifying a Rule panel, you define scheduling requirements by selecting at least one item from each of the following three columns:

- **Frequency:** Select a run frequency for **Every** or **Only** when you specify rules for your run cycle. You can also select an ordinal number from **First** to **Fifth** and 6 - 999 or its **Last** equivalent. Values higher than Fifth or 5th Last can be entered in the blank spaces below Fifth or 5th Last. You can make multiple selections. If you select **Only** and make no other selections, dates are generated as if you selected **Only** and **First**.
- **Day:** Make one or more selections to specify the run days for the application.
- **Cycle Specification:** Select from the list of available Tivoli Workload Scheduler for z/OS defined scheduling cycles. These cycles are like supplied periods which satisfy most requirements. For unusual schedules, user-defined periods and run cycle groups can also be specified in this column.



Note: You must make one or more unambiguous selections in this column. Selecting **Month** and one of the listed months is an ambiguous selection. The **Week** cycle in this column starts on **Monday**. You can specify a user-defined period or run cycle group if the listed cycles do not meet your needs. You can shift the start day of a cycle by using the **Shift default origin** date field.

Using the Every option

RC1	Su	Mo	Tu	We	Th	Fr	Sa
09:00	1	2	3	4	5	6	7
	8	9	10	11	12	13	14

RC1	Su	Mo	Tu	We	Th	Fr	Sa
10:00	1	2	3	4	5	6	7
	8	9	10	11	12	13	14

RC1	Su	Mo	Tu	We	Th	Fr	Sa
11:00	1	2	3	4	5	6	7
	8	9	10	11	12	13	14

RC1	Su	Mo	Tu	We	Th	Fr	Sa
12:00	1	2	3	4	5	6	7
	8	9	10	11	12	13	14

- Create multiple occurrences in one panel
- Repeat the run cycle on equal intervals
- Repeat time
 - a. From (start time)
 - b. Until (end time)

Example:
Repeat every 01:00
a. From 09:00
b. Until 12:00

© Copyright IBM Corporation 2015

Using the Every option

By using the **Every** option, you define repeating run cycles from one panel. You can schedule an application to run several times a day without manually creating individual run cycles for each occurrence. The slide shows how run cycle RC1 repeats four times, by using the every option.

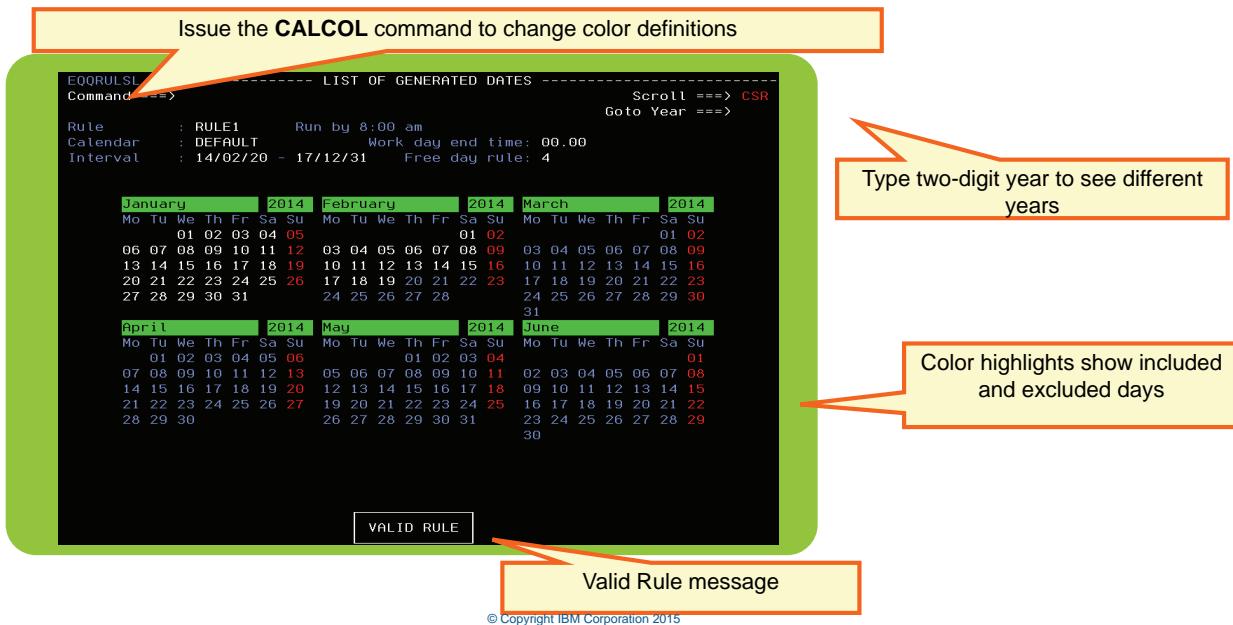


Note: The **Every** options is processed by the long-term plan to create multiple occurrences. Manually adding an application to the current plan does not create multiple occurrences based on run cycle **Every** options that are defined for the application.

Use the **E command** in the Modifying a Rule panel to specify the **Every** options. The field descriptions for the Every Options panel are as follows:

- **FROM:** Shows the input arrival time of the run cycle that identifies the repeat start time.
- **REPEAT EVERY:** An input field that specifies the repeat time interval in the *HH.MM* format.
- **UNTIL:** An input field that specifies the repeat end time in the *HH.MM* format. It must be a value between the input arrival time and the application calendar workday end time.

Using the GENDAYS command



Using the GENDAYAS command

After selecting frequency, day, and cycle specifications, you can enter the GENDAYS command to show a calendar with the run days generated for the rule. This slide shows an example of the GENDAYS results.

If no calendar is specified in the application, the GENDAYS function uses the calendar that is specified on the option **0.2** panel. The GENDAYS function presents the run days for one run cycle at a time. GENDAYS does not show you combined run cycles, only the current run cycle.

The GENDAYS command notifies you that the **Every** option was specified. It also shows the number of repeat occurrences for each day.

Creating unique application occurrences

- Each instance of an application in the long-term plan and the current plan is called an occurrence
- Tivoli Workload Scheduler for z/OS creates unique occurrences using the Application ID, Input Arrival Date, and Input Arrival Time



© Copyright IBM Corporation 2015

Creating unique application occurrences

Each instance of an application in the LTP and CP is called an **occurrence**. Each occurrence in the plans is a unique record in the plan files. The fields that are used to form the unique key for occurrences are as follows:

- The Application ID
- The Input Arrival date
- The Input Arrival time

Lesson 4 Defining operations

Lesson 4 Defining operations

Tasks that the Tivoli Workload Scheduler for z/OS runs

- Job operations on computer workstations
- Started-task operations
- Operations on distributed agents
- Job setup operations
- Print operations
- WTO operations
- Operations that have events reported
- Dummy operations

When defining operations, you also define items as follows

- Dependencies (internal, external)
- Resource requirements
- Automatic options
- Times and dates
- Operator instructions
- Restart and cleanup options

© Copyright IBM Corporation 2015

In this lesson, you learn how to define operations within an application. After completing this lesson, you should be able to define operations within an application.

Entering operations information

Use row commands to insert (I), repeat (R) or delete (D) operations

- Use the **TEXT** or **PRED** command to switch modes
- Use the **GRAPH** command to display the operation list graphically
- Use the **S** row command to edit details of an operation

Field	Type	Required
Workstation name	1-4 alphanumeric	Y
Operation number	1-255	Y
Estimated duration	HH.MM.SS	Y
Job name	1-8 alphanumeric	*
Internal predecessors	Operation number of each predecessor in the same application	*
Operation text	Text	

© Copyright IBM Corporation 2015

Entering operations information

The operations segment holds the operations in type A applications. It is not used in group applications. A application requires at least one operation. Issue the OPER command to see the Operations panel.

```
EQQACGPP ----- CREATING AN APPLICATION -----
Command ==> OPER_
Enter/Change data below:
Enter the RUN command above to select run cycles or enter the OPER command
```

Tivoli Workload Scheduler for z/OS has two panels for defining operations in an application description:

- A text operations panel
- A predecessor operations panel

On the text panel, you specify up to 24 characters of descriptive text for each operation. This text is included in WTO messages that are issued for the operation. To use this panel, type the TEXT command on the predecessor operations panel.

Use the predecessor panel to define the internal dependencies between operations in the application description. To use this panel, enter the PRED command on the text operations panel.

- Each operation is a task or job at a workstation; therefore, you must specify a workstation to indicate the type of task to be run.
- Every operation must have a unique number within the application. This operation number defines the dependency links within the application.

- For planning purposes, you must give every operation an estimated duration, or run time.
- Other than write to operator (WTO) and non-reporting workstations all operations must have an assigned job name.

The execution of an operation in the plan might depend on the completion of another operation.

The operation that depends on another is called the successor. A successor depends on a predecessor.

In applications with more than one operation, you must specify dependencies between the operations. Tivoli Workload Scheduler for z/OS uses operation numbers to define the internal dependency links. Define each operation with a workstation, an operation number, the estimated duration, and the job name.

Row cmd	Oper ws no.	Duration HH.MM.SS	Job name	Internal predecessors	Morepreds -IntExt-	No.of Conds
OPJC 001	00.02.00	JOBJ1		-----	0 0	0
OPJC 002	00.05.00	JOBJ1		-----	0 0	0
OPJC 003	00.03.00	JOBJ1		-----	0 0	0

Add the internal predecessor specifications. Use operation numbers to identify the predecessors.

Row cmd	Oper ws no.	Duration HH.MM.SS	Job name	Internal predecessors	Morepreds -IntExt-	No.of Conds
OPJC 001	00.02.00	JOBJ1		-----	0 0	0
OPJC 002	00.05.00	JOBJ1	001	-----	0 0	0
OPJC 003	00.03.00	JOBJ1	002	-----	0 0	0

You can specify up to seven internal predecessors for each operation on this panel. To define more than seven internal predecessors, or to enter other operation-related details, enter the **S** (Select operation details) row command next to the operation.

Operations panel

OPERATIONS							Row 1 to 7 of 7
							Scroll ==> PAGE
Enter/Change data in the rows, and/or enter any of the following row commands:							
I(nn) - Insert, R(nn),RR(nn) - Repeat, D(nn),DD - Delete							
S - Select operation details, J - Edit JCL							
Enter the TEXT command above to include operation text, or, enter the GRAPH command to view the list graphically.							
Application : U#APPL1A TEAM # APPL 1 A							
Row cmd	Oper ws no.	Duration HH.MM.SS	Job name	Internal predecessors			Morepreds -IntExt- No. of Conds
005	U#CP 005	00.05.00	SM40#J1	005			0 2 0
010	U#CP 010	00.05.00	SM40#J2	005			0 0 0
014	U#JS 014	00.02.18	SM40#J3	005			0 0 0
015	U#CP 015	00.01.30	SM40#J3	005 014			0 0 0
020	U#CP 020	00.05.00	SM40#J4	010 015			0 0 0
024	U#JS 024	00.05.00	SM40#J5	020 024			0 0 0
025	U#CP 025	00.05.00	SM40#J5	020 024			0 1 0

Execute the **OPER** command from the CREATING AN APPLICATION panel

© Copyright IBM Corporation 2015

Operations panel

You define the general information for one or more operations on the Operations panel. You then use the **S** row command to select individual operations and define the details for that operation. Using the **S** row command takes you to a menu panel where you can enter additional dialogs to enter more detail about each operation.

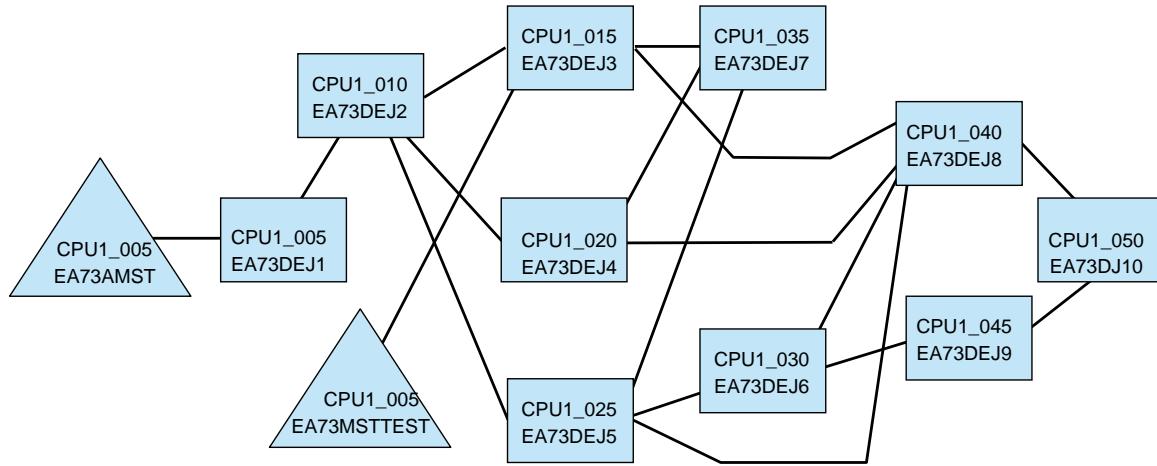
You can use these three commands on the command line of the Operations panel:

- **TEXT**: Text view
- **PRED**: Predecessors view
- **GRAPH**: GDDM graphical view

Use the TEXT and PRED commands to show two distinct panel views for the Operations panel: EQQAMOPL (TEXT command view) and EQQAMOSL (PRED command view).

The GRAPH command uses GDDM to create a graphical view of the operation predecessors. This slide shows normal internal application predecessors. You can also see in this slide that operations 005 and 025 both have normal external predecessors. You will see an example of the GRAPH command on the next slide.

GRAPH command example display



© Copyright IBM Corporation 2015

GRAPH command example display

Tivoli Workload Scheduler for z/OS uses the z/OS graphics software GDDM to show the layout of operations and their dependencies. GDDM supports three graphic shapes:

- Boxes (the default)
- Circles
- Triangles

On most operations list panels, you can enter the GRAPH command to show the operations graphically. Requirements for using this display are as follows:

- GDDM must be installed on the z/OS host.
- Host Graphics must be configured on the 3270 terminal.

After the graphic is shown, you can customize your view for shape, color, and line type by entering the ATTR command. You can then specify the appropriate attributes. You read the graphic from left to right. In the slide, **boxes** represent computer operations in the application and **triangles** represent external dependencies. The application has two external predecessor dependencies.

- The first line of text in each of the shapes is an operation ID.
- The second line of text in the boxes is the job name that is associated with the operation.
- The second line of text in the triangles is the name of the application for the operation.

In the application database, an application description graphic is a tool for viewing an application and all of its immediate predecessors.

More on predecessor dependencies

- Operation predecessors are dependencies
 - Several types
 - Predecessor to successor relationship between operations
 - Operations can have multiple predecessors and successors
- Internal
 - Predecessor operations within the same application
 - Defined on the OPERATIONS panel and in operation details
- External
 - Predecessor operations in other applications
 - Defined in the operation details
- Conditional
 - Job level (can be internal or external)
 - Step level (can be internal or external)
 - Defined in the operation details
- Multiple matching criteria for dependency resolution
 - Defined in the operation details on the PREDECESSORS panel

© Copyright IBM Corporation 2015

More on predecessor dependencies

To define more than seven internal predecessors and more details for an operation, issue row command S to select the operation, the OPERATION DETAILS panel opens. Option 1 on this menu panel is the Predecessors panel, from here you can define dependencies as follows:

- **Internal dependencies:** More than the seven that are allowed on the Operations panel. If an operation has more than seven internal predecessors, you specify them here without the application name.
- **External dependencies:** Dependencies on operations in other applications. External dependencies are linked at the operation level.

You need the workstation name, operation identification number and name of the external application. The job name is not actually used to identify the dependency in TWSz, but it can be used as a search criteria. If you do not know all the values, you can enter the information that you do have. You can use wildcards to list matching jobs, and then select the correct one.

- **Conditional dependencies:** Dependencies, which you specify, that an operations depends on, such as the return code or status of an entire job or on the return code or status of a specified step. You can base the execution of successor jobs on step-level dependencies. An operation can depend on the return code of a specified step within a predecessor multiple-step operation. This dependency adds further flexibility in modeling and controlling complex work flows.
 - **Job level conditional dependencies:** Use the highest return code or status of the predecessor operation.
 - **Step-level conditional dependencies:** A step-level dependency that is based on the condition that the step of a job returns a specified code. The step is identified by the step

name, the proc name, or both. The **step name** is the procedure invocation step name. It is blank if the step is not in a procedure. The proc name is the step name in the JCL procedure or the job step name if the step is not in a procedure.

There are several options for defining in the Application Description how a dependency between predecessor and successor operations is resolved. The criterion that is chosen to pick a matching predecessor that best resolves the dependency can be that the matching predecessor is the one with:

- The nearest preceding input arrival time. The successor operation input arrival time (OPIAT) must be the same or later than the application input arrival time (ADIAT) of its predecessor. An operation can have its own OPIA time specified, that differs from the input arrival time (IAT) of its application. Otherwise it inherits its ADIAT.
- The nearest input arrival time within the same day of the successor.
- The closest input arrival time within a specified interval. The interval boundaries are calculated by an offset that is expressed in hours and minutes before or after the IA time of the successor. The interval can be timed entirely before, entirely after, or across the IA time of the successor.
- The closest input arrival time within a specified interval. The interval boundaries are specified by a time and a number of days before or after the IA time of the successor. The interval can be timed entirely before, entirely after, or across the IA time of the successor.
- Multiple matching criteria

The process by which a dependency is resolved in the plan can be defined as finding the best match between the successor operation and the predecessor application. The process takes as its primary basis the input arrival times of the successor operation and of the predecessor application. Starting from the input arrival time of the successor, the dependency is resolved when a predecessor with the input arrival time that best matches the selection criteria that was defined in the database is found.

In addition, the resolution of a normal dependency can be defined mandatory at plan level where, if the predecessor is not in the plan, the plan (LTP and DP) fails directly, or at control level where, if the predecessor is missing, a pending mandatory predecessor entry is added and the occurrence remains in waiting status.

The successor operation dependency resolution mechanism does not apply to conditional or cross dependencies.

Dependencies are defined in the database and resolved in the plans (long-term, current, and daily). In the long-term and daily plans, they are resolved when the batches run. In the current plan, they are resolved when occurrences are added dynamically through PIF, ETT, ISPF, or the Dynamic Workload Console.

Operation details panel

```
EQQAMSDP ----- OPERATION DETAILS -----
Option ==> _

Select one of the following:

 1 PREDECESSORS      - List of predecessors
 2 WS RES AND SERVERS - Work station resources and servers
 3 SPECIAL RESOURCES - List of special resources
 4 AUTOMATIC OPTIONS - Job, WTO, and print options
 5 FEEDBACK          - Feedback options
 6 TIME               - Time specifications
 7 OP INSTRUCTIONS   - Operator instructions
 8 JCL EDIT           - Edit JCL
 9 CLEANUP OPTIONS   - Cleanup Options
10 EXTENDED INFO     - Operation extended info
11 AUTOMATION INFO   - System Automation operation info
12 USER FIELDS        - User Fields operation info
13 REMOTE JOB INFO    - Remote job information

Application       : U#APPL1A      TEAM # APPL 1 A
Operation         : U#CP 010      Job 2
Jobname           : SM40#J2      Number of int preds : 1
Duration          : 00.05.00    Number of ext preds : 0
                           Number of conditions : 0
```

© Copyright IBM Corporation 2015

Operations details panel

You select an operation from the Operations panel by using the **S** row command. You can add new predecessors by selecting option **1** on the Operation Details panel. These predecessors can be a mix of internal, external, and conditional. Other options are also available from this panel and are described later in the course.

Predecessors panel

EQQAMPDL ----- PREDECESSORS ----- Row 1 of 1
 Command ===> Scroll ===> CSR

Enter the COND command to view the list of conditions, enter/change data in the rows, and/or enter any of the following row commands:

I(nn) - Insert, R(nn),RR(nn) - Repeat, D(nn),DD - Delete
 S - Description of external dependency, T - Dependency resolution criteria

Application : PRDAPPL1	Production Appl 1			
Operation : INCP 005	JPROD1			
No. of conditions: 0				
Row cmd	Oper ws no.	Transport time	Application id (for ext pred only)	Jobname
T ''	INCP 010	HH.MM	EA73MST	EA73MST1

© Copyright IBM Corporation 2015

Predecessors panel

From the Predecessors panel, you can define normal external predecessors. To define conditional predecessors, you enter the COND command on the command line.

You can also define dependency resolution criteria by using the T row command. You will see the DEPENDENCY RESOLUTION CRITERIA panel on the next slide.

This panel also displays normal internal predecessors. You cannot select the internal predecessors from this panel, but you can delete them.

Dependency resolution panel

```
EQQAMMAT ----- DEPENDENCY RESOLUTION CRITERIA -----
Command ==>
Application : EA73MST          Operation : INCP   10   EA73MST1
Resolution is mandatory:        N  (P/C/N)
Resolution criteria chosen is: C  (C/S/R/A)
-(C) Closest preceding
-(S) Same scheduled date
-(R) Within a relative interval
  FROM:
    hours (HHH): ____ minutes (MM): ____ when (B/A before/after IA): __
  TO:
    hours (HHH): ____ minutes (MM): ____ when (B/A before/after IA): __
-(A) Within an absolute interval
  FROM:
    time (HH.MM): _____ for days (D): ____ when (B/A before/after IA): __
  TO:
    time (HH.MM): _____ for days (D): ____ when (B/A before/after IA): __
```

© Copyright IBM Corporation 2015

Dependency resolution panel

Predecessor Application

Shows the name of a predecessor application.

- Operation: Shows the workstation name, operation number, and job name.

Resolution is mandatory:

- P Plan Dependency is mandatory and must be immediately resolved.
- C Control Dependency is mandatory and can be resolved later.
- N No Dependency is not mandatory.

Resolution criteria that can be chosen is:

- C Closest Preceding Match occurs with the closest preceding Input Arrival Time.
- S Same scheduled date Match occurs with closest Input Arrival Time in the same day.
- R Within a relative interval Match occurs with closest Input

(R) Within a relative interval From and To options.

Match occurs with closest Input Arrival Time in that specified relative interval. If all the FROM fields are left blank, the left side of the interval is open.

FROM

- HHH Set the number of hours before or after the Input Arrival time (as specified in when field) from which the interval must start. It can be blank to indicate no limit on the left part of the interval.
- MM Set the number of minutes before or after the Input Arrival time (as specified in when field) from which the interval must start. It can be blank only if HHH is blank.
- WHEN Possible values:
 - - B meaning before IA.
 - - A meaning after IA.
 - Both values refer to the input arrival of the operation. It can be blank only if HHH is blank.

TO

- HHH Set the number of hours before or after the Input Arrival time (as specified in when field) to which the interval must end.
- MM Set the number of minutes before or after the Input Arrival time (as specified in when field) to which the interval must end.
- WHEN Possible values:
 - - B meaning before IA.
 - - A meaning after IA. Both values refer to the input arrival of the operation.

(A) Within an absolute Interval From and To options

Match occurs with closest Input Arrival Time in the specified absolute interval.

FROM

- HH.MM Set the time before or after the Input Arrival time (as specified in the when field) from which the interval must start.
- FOR DAYS Number of days to go back and Arrival time (as specified in the when field) from which the interval must start.
- FOR DAYS Number of days to go back and ahead.
- WHEN Possible values:
 - - B meaning before IA.
 - - A meaning after IA.
 - Both values refer to the input arrival of the operation.

TO

- HH.MM Set the time before or after the Input Arrival time (as specified in the *when* field) to which the interval must end.
- FOR DAYS Number of days to go back and ahead.
- WHEN Possible values:
 - - B meaning before IA.
 - - A meaning after IA.
 - Both values refer to the input arrival of the operation.

Conditional dependency components

Condition

- A group of one or more condition dependency definitions and a condition rule
- Each condition is assigned a number of 1 to 999 when it is defined
- A condition is either true or false, based on the outcome of the condition dependencies that comprise the condition and the condition rule

Condition dependency

- A predecessor and successor relationship between two operations that is defined as part of a condition
- A dependency that uses either a predecessor operation status or return code as part of a logical expression (Example: RC \geq 4)
- The expression is evaluated as either true or false by Tivoli Workload Scheduler for z/OS

Condition rule

- A three-digit decimal number that is defined as part of the group of one or more condition dependencies within a condition
- Determines how many condition dependencies must be true for the condition to be true (000 = ALL)
- Defined on the same ISPF panel as condition dependencies

© Copyright IBM Corporation 2015

Conditional dependency components

Conditional dependencies are defined as a sub component of a condition. This hierarchical relationship can be one-to-many, and you can define more than one condition for an operation.

The condition rule determines how many of the defined conditional dependencies must evaluate true for the condition to be true. The default is ALL, meaning that if more than one condition is defined for an operation, *all* conditions must evaluate TRUE before the operation can run.

Condition dependencies are not used as a substitute for the required internal dependencies.

Conditions list panel

```
EQQAMCCL ----- CONDITIONS LIST ----- Row 1 of 1
Command ==> _                                         Scroll ==> CSR

Enter/change data in the rows, and/or enter any of the following
row commands:
I(nn) - Insert, R(nn),RR(nn) - Repeat, D(nn),DD - Delete
S - Specify the condition details

Application      : PRDAPPL1                      Production Appl 1
Operation        : INCP 005          JPROD1

Row   Condition Text           Cond Deps   Rule
cmd  no.    Text
***  001    COND 1_____       1         all
```

- Run the **COND** command to see the CONDITIONS LIST panel
- Set the condition rule and dependencies by using the **S** row command

© Copyright IBM Corporation 2015

Conditions list panel

In this example, operation 005 has one condition that is defined. The condition has one conditional dependency that must evaluate true for the condition to be true. Use the **S** row command to select the condition and define conditional dependencies for the condition.

Condition dependencies definition panel

- Define one or more external or internal operation dependencies and a three-digit rule that sets the total number required to be true 000 means ALL
 - The T row command can also be used for conditional dependencies to access the DEPENDENCY RESOLUTION CRITERIA panel

© Copyright IBM Corporation 2015

Condition dependencies definition panel

You can base the evaluation of the rules on the operation status or the return code. Use **ST** in the CoTy column to specify status. Processor status does not apply to step dependencies. Use **RC** in the CoTy column to specify return code evaluation.

When you add rules to the condition, you can specify how many rules must be true for satisfying the condition.

- If you leave the value 0, *all* of the rules must be true. This option corresponds to Boolean AND logic across all the rules.
 - If you enter a 1 for the value, only *one* of the rules must be true. This option corresponds to Boolean OR logic.

You can also specify that a specific number of rules must be true. For example, if a job is waiting for five predecessors, but the job can start as soon as three of them are complete, you define the following information:

- Five rules, one rule for each of the five predecessors, specifying that the status is complete.
 - The number of conditions that must be verified is three.

To specify a dependency on a job step, use the **Proc Step** field if the step is not in a procedure. If the step is in a procedure, use both **Step Name** and **Proc Step** fields. The Proc Step that you specify must correspond to a step used in an EXEC PGM= statement.

For return code rules evaluation, you specify one or two return codes and a logical operator for the required check. In the Co OP column, you specify the logical operator. In the Ret.Code columns, enter the lower value for **Val1** and the higher value (if two values are needed) for **Val2**. The

operators that you can use are: GE, GT, LE, LT, EQ, NE, and RG. RG is for a range of values and uses both val1 and val2.

For status rules evaluation, the values you can specify are: C (complete), S (started), E (error), and X (suppressed by condition).

Condition-related status codes

For operations with defined conditions, the following codes are used:

- The operation status starts as W Waiting, until the scheduler has evaluated all the defined conditions
- The operation status changes to R Ready, when all the defined conditions are True. The scheduler processes next possible statuses as usual, up to a final status of C or E
- The operation status changes to X, Suppressed by condition, when any condition is False and the operation does not run

© Copyright IBM Corporation 2015

Condition related status codes

As part of this feature an operation status, **X**, is introduced. The X status means *suppressed by condition*. An operation receives this status if the operation is not selected to run because of a false condition.

Condition dependency example

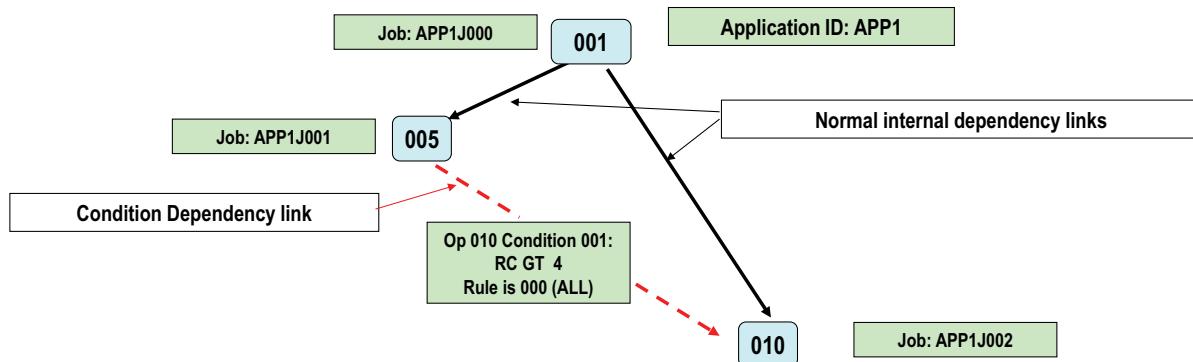
Internal-only condition example:

There are three operations: **001**, **005**, **010**

005 must run after **001** ends successfully

If job **005** fails with an RC greater than 4, job **010** must run

If job **005** ends with an RC of 4 or less, job **010** does not run and ends with a status of X



© Copyright IBM Corporation 2015

Condition dependency example

This slide shows an example of a single application, APP1, with three operations. Operations 005 and 010 have normal internal dependencies with operation 001. Each operation in an application is required to have a normal internal dependency. Operation 010 also has an internal Condition Dependency with operation 005. This Condition Dependency is optional. The thick black arrows represent the normal dependencies and the dashed red arrow represents the Condition Dependency. The Condition number is 001 and contains only one Condition Dependency of Return Code that is greater than 4. The Condition Rule is 000, which means All Condition Dependencies must be true.

The goal of this application is to always have operations 001 and 005 run, but not operation 010. Operation 010 runs only when operation 005 has a return code greater than 4. Operation 010 can provide some recovery and diagnostics for the failed operation 005. Even though operation 010 has a normal dependency with operation 001, operation 010 does not run when operation 001 completes successfully. It runs only if the Condition Dependency with operation 005 evaluates to true.

If operation 001 fails, no other operations run. You must handle the error manually or by some other means.

Specifying other operation details

- Defining workstation resources and servers
- Setting automatic options
- Configuring historical data feedback
- Setting time dependencies
- Applying operator instructions
- Configuring restart and cleanup options
- Adding extended operation information
- Specifying systems automation information

© Copyright IBM Corporation 2015

Specifying other operation details

You can specify other operation details, such as these examples:

- **Work Station Resources and Servers:** Enter the number of parallel servers and workstation resources that are consumed by the operation. The default is one parallel server and 0 workstation resources.
- **Special Resources:** Symbolic names for any physical or logical resources that are needed for an operation.

Special resources are often used for data sets, or for tapes and cartridges, typically for preventing allocation conflicts between jobs that share files or data sets. These resources can be shared between operations or held exclusively by a single operation. Job tracking prevents operations that require a special resource from starting if another operation holds it exclusively.

- **Automatic Options:** Options for specifying more finite details, such as the following examples:
 - Job class
 - Highest acceptable return code
 - Time dependency
 - Rerouting and restarting in case of workstation failure
 - Suppressing the job if it is late getting started
- **Feedback:** Instructions for overriding the initialization defaults.
- **Time:** Start time for time-dependent operations or a deadline time for the operation.

- **Operator Instructions:** Text field for writing online instructions for setting up, starting, restarting, or rerunning the operation.
- **Cleanup Options:** Options for job restarts and data set cleanup for the operation.
- **Extended Operation information:** Optional 54-character field for specifying extended job name information that is needed in the distributed environment.
- **Automation Info:** Field to specify System Automation for z/OS request and response information.
- **User-defined fields:** Fields that you can create and specify additional information about the operation.

The information that you enter is stored in the application definition and current plan. Tivoli Workload Scheduler for z/OS does not process user fields. They do not affect how operations run. You can add up to 120 user fields. Each user field name can be up to 16 characters. Each user field value can be up to 54 characters. User fields can be viewed in the current plan under operation details panels. The following user exits can also read user fields:

- EQQUX001
- EQQUX002
- EQQUX007
- EQQJVXIT

Operation automatic options

```

EQQAMJBP ----- JOB, WTO, AND PRINT OPTIONS -----
Command ==> _
Enter/Change data below:
Application   : U#CRITA          TEAM # Critical App
Operation     : U#CP 020 U#CRITJ4
JOB CLASS     ==> _             Job class of corresponding job
ERROR TRACKING ==> Y           Y means error automatically tracked
HIGHEST RETURNCODE ==> _       Highest return code not in error
EXTERNAL MONITOR ==> N         Job monitored by external product (Y/N)
CENTRALIZED SCRIPT ==> N       Centralized script Y/N (for FTW only)
COND RECOVERY JOB ==> N        Recovery job for a cond predecessor (Y/N)
CRITICAL      ==> P           Critical job: N P W
POLICY ==> _ CLASS ==> _       WLM Policy and Service Class
Job release options:
  SUBMIT      ==> Y           Answer Y or N for options below
  HOLD/RELEASE ==> Y           Automatically submitted
  TIME DEPENDENT ==> N       Automatically held and released
  SUPPRESS IF LATE ==> N      Run the job at specified time
  DEADLINE WTO ==> N         Suppress the job if not on time
  Deadline WTO ==> N         Deadline WTO, Y or N
WS fail options:
  RESTARTABLE ==> _          Operation is restartable
  REROUTEABLE ==> _          Operation is eligible for reroute
Print options:
  FORM NUMBER ==> _          SYSOUT CLASS ==> _

```

A job can be flagged as critical

© Copyright IBM Corporation 2015

Operation automatic options

You can set the following options:

- **JOB CLASS:** Specify the JES input class of the job that the operation represents. The job class that you specify here is for documentation purposes.



Note: This option does not apply to fault-tolerant workstations.

- **ERROR TRACKING:** Specify that Tivoli Workload Scheduler for z/OS marks the operation Ended in error (**E**) if the job or started task that is defined in the operation fails.
- **HIGHEST RETURNCODE:** Specify the highest acceptable return code from any step in the job or started task. If a return code for a step in the job or started task exceeds this value, the status is set to an **E** (Ended in error). For fault-tolerant workstations, the valid value is either zero or blank.
- **EXTERNAL MONITOR:** Specify whether the operation should be monitored by an external product (for example, Tivoli Business Systems Manager). If you specify **Y** (yes) for this option, monitoring is used. For fault-tolerant workstations, the valid value is either zero or blank.
- **CENTRALIZED SCRIPT:** Applies to fault-tolerant workstations only. By default, scripts for jobs that run on fault-tolerant workstations are not centralized. They are stored on the agent, and a job definition is added to the SCRPTLIB data set on the Tivoli Workload Scheduler for z/OS controller. By contrast, centralized scripts are stored in one or more data sets in the JOBLIB

allocated to the controller. It provides features such as automatic job tailoring, job setup, automatic recovery, and Tivoli Workload Scheduler for z/OS user exit **01** functions.

- **CRITICAL:** Specify whether the operation is considered critical, and eligible for Workload Manager assistance if late. Valid values are as follows:
 - **P:** Critical path target
 - **W:** Eligible for Workload Manager assistance
 - **N:** Not eligible for Workload Manager assistance
 - **POLICY:** Specify the Workload Manager service class assist policy that Tivoli Workload Scheduler for z/OS uses. The policy must be met before Tivoli Workload Scheduler for z/OS requests a service class promotion. The valid values for this field are as follows:
 - **L** (Long duration): If the job runs beyond its estimated duration
 - **D** (Deadline): If the job is not finished when its deadline time is reached
 - **S** (Latest start time): If the job is submitted after the latest start time
 - **C** (Conditional): Determines whether to use the Deadline or Latest start time policy

This option does not apply to fault-tolerant workstations.
 - **CLASS:** Specify a Workload Manager Service Class.
 - **SUBMIT:** Specify whether Tivoli Workload Scheduler for z/OS starts the operation when all predecessors are satisfied and all required resources are available.
 - **HOLD/RELEASE:** Control held jobs not originally submitted by the scheduler. This option affects jobs that enter JES in HOLD status. Jobs can be put in HOLD status by specifying **TYPRUN=HOLD** on the job card, for example. You can set HOLD/RELEASE to **Y** (yes) for these jobs, and the scheduler releases them according to its schedule. All dependencies must be satisfied and requested resources available first.

If you set HOLD/RELEASE to **N** (no), the scheduler releases a held job immediately without considering the schedule. You cannot set this option to **N** (no) for fault-tolerant workstations.
 - **TIME-DEPENDENT:** Specify **Y** (yes) to tell Tivoli Workload Scheduler for z/OS to start the operation only at the specified input arrival time.
 - **SUPPRESS IF LATE:** Specify **Y** (yes) to stop this time-dependent operation from being started if it is late. If you specify **N** (no, the default), Tivoli Workload Scheduler for z/OS ignores the fact that the operation is late and tries to start it as soon as possible.
 - **DEADLINE WTO:** If you specify **Y** (yes) for this option and the operation is started after its deadline time, the scheduler issues the operator message EQQW776I. The WTO is issued only for z/OS operations that have status **S** (started).
- The user-defined text that you specified on the operations panel by using the TEXT command is included in the WTO. This option is applicable to z/OS workstation destinations only.
- **RESTARTABLE:** Specify whether you can restart the operation after Tivoli Workload Scheduler for z/OS reroutes it to an alternative computer workstation. This option is only considered as a result of a failure or offline condition of the primary computer workstation. This option applies to

the operation only when its status is **S** (started). This option does not apply to fault-tolerant workstations.

- **REROUTABLE:** Specify the action for Tivoli Workload Scheduler for z/OS to take for this operation if the primary assigned workstation is inactive at the scheduled run time. An alternative workstation must be specified for the operation before this option is considered. This option applies to the operation only when it is in status **R** (ready) or **W** (waiting). After the operation is in status **S** (started), the RESTARTABLE option determines the action. This option does not apply to fault-tolerant workstations.

Considerations for operations

- Job setup operations
 - Must immediately precede the CPU workstation
 - Job name must match one at the CPU workstation
- Wait operation
 - Purposely creates a delay in the operations flow
 - Must run on a wait workstation
 - Duration of the operation is the delay time
- Automation operation
 - Must run on an automation workstation
 - Uses the Automation Info Details (option 11) to specify the Tivoli System Automation for z/OS request and response details

© Copyright IBM Corporation 2015

Considerations for operations

A job setup operation must have a computer operation with the same job name as its immediate successor. JCL preparation for the job setup operation is performed on the workstation Ready List panel.

Tivoli Workload Scheduler for z/OS copies the JCL from the production JCL library into its own JCL repository. It does so if the JCL is not available when the user makes the first edit request.

Subsequent edit requests use the copied JCL. The scheduler never changes JCL in the production library.

When the operator completes the preparation, Tivoli Workload Scheduler for z/OS changes the status of the operation to **C** (complete). The successor operation that is assigned to a computer workstation is then ready for processing.

By definition, any operation that is assigned to a wait workstation is a wait operation with the wait time specified in the duration of the operation.

By definition, any operation that is assigned to an automation workstation is an automation operation. Use option **11** (Automation Info) on the Operation Details panel to specify the details of the System Automation for z/OS request and response.

Operation-related data sets

Mainframe-related data

- EQQJBLIB: PDS where you define the members containing job JCL
- EQQJS1DS and EQQJS2DS: VSAM KSDS JCL repository data sets
- EQQJTARC: Sequential file containing the job archive
- EQQJTnn: Sequential file containing job tracking log
- EQQDLnn for dual job tracking
- EQQSTC: PDS for started tasks

End-to-end related data

- EQQJBLIB: PDS where you define the members containing job records for zCentric agent workstations
- EQQSCLIB: PDS used for the end-to-end script library for plan-based end-to-end workstations
- EQQTWSCS: PDS used for end-to-end centralized scripts for plan-based end-to-end workstations

© Copyright IBM Corporation 2015

Operation-related data sets

Allocate the job library data sets with only a primary space allocation. If a secondary allocation is defined and the library goes into an secondary extent when Tivoli Workload Scheduler for z/OS is active, restart the controller. Do not compress members in this PDS (partitioned data set). Do not use the ISPF PACK ON command because Tivoli Workload Scheduler for z/OS does not use ISPF services to read it. The limitation of allocating the job library data set with only a primary space allocation does not apply for PDSE (partitioned data set extended) data sets.

Each member in the EQQJBLIB must contain one job stream (only one job card). The job name on the job card must match the job name in the Tivoli Workload Scheduler for z/OS scheduled operation.

The scheduler maintains its own copy of JCL in the JCL repository data set for every job that it submits in the current plan. The scheduler uses a primary and alternate data set for the JCL repository: EQQJS1DS and EQQJS2DS. It reorganizes the JCL repository data set that is in use by copying it to the alternate data set. It then switches over to the newly copied data set. The value that you specify on the MAXJSFILE keyword defines whether the JCL repository is automatically copied. It also determines how often the automatic copy process occurs.

Job-tracking data sets are a log of updates to the current plan. They optionally contain audit trail records. Job-tracking data sets include the following items:

- Job-tracking logs (EQQJTnn)
- Dual job-tracking logs (EQQDLnn)
- Job-tracking archive (EQQJTARC)

You must allocate EQQJTARC and at least five job-tracking logs (EQQJT01, EQQJT02 - EQQJT05) for a controller. The actual number of JT logs that you allocate must match the value that you specify on the JTLOGS keyword. The JTLOGS keyword is part of the JTOPTS initialization statement. If you decide to allocate more than five job-tracking logs, specify the DD names EQQJT01 to EQQJTnn (where nn is equal to the number specified on the JTLOGS Keyword). If, you skip a file in the sequence, an error occurs and the scheduler terminates. The scheduler uses the job-tracking logs in turn. When a current plan backup task is performed, the active job-tracking log is closed, the next started, and appended to the EQQJTARC data set.

Tivoli Workload Scheduler for z/OS uses the started-task-submit data set to temporarily store JCL when a started task is to start. Use the EQQSTC attributes for this data set:

```
SPACE=(TRK,(5,0,1)),  
DCB=(RECFM=FB,LRECL=80,BLKSIZE=3120)
```

If RCLEANUP(YES) is used, the dataset that is allocated to DD EQQSTC must be in the master scheduler IEFJOBS or IEFPDSI concatenation. Both IEFJOBS and IEFPDSI require that the FIRST JCL card in the member to be used in the START command **must** be a valid JOB card.

If RCLEANUP(NO) is used, the EQQSTC dataset must be allocated as the first library in JES PROC00 concatenation on each system where Tivoli Workload Scheduler for z/OS schedules started-task operations.

The data set is a temporary staging area for the started-task JCL procedure.

To use centralized script support when scheduling end-to-end, you must use the EQQTWS CS DD statement in the controller and server started tasks. The data set must be a partitioned extended data set. Tivoli Workload Scheduler for z/OS uses the end-to-end centralized script data set to temporarily store a script. It is downloaded from the JOBLIB data set to the agent for its submission. Set the following attributes for EQQTWS CS:

```
DSNTYPE=LIBRARY,  
SPACE=(CYL,(1,1,10)),  
DCB=(RECFM=FB,LRECL=80,BLKSIZE=3120)
```

The EQQSCLIB script library data set includes members that contain the commands or the job definitions for fault-tolerant workstations. It is required in the controller if you want to use the end-to-end scheduling feature. See the *IBM Tivoli Workload Scheduler for z/OS Customization and Tuning Guide* for details on the JOBREC, RECOVERY, and VARSUB statements. Do not compress members in this PDS. Do not use the ISPF PACK ON command.

Lesson 5 Job descriptions

Lesson 6 Job descriptions

- The job contains no more than three operations:
 - One is computer operation or WTO operation
 - The other two operations can be either manual completion or JCL preparation operations
- The job name of the computer operation is the same as the job description ID
- If you create the job using the AD panel, its status is Active

© Copyright IBM Corporation 2015

In this lesson, you learn how to define a job description. After completing this lesson, you should be able to define a job description.

If you want only one main processor operation (job) in an application, you can create it with the Job Description panel. This panel compresses most of the function of the Application Description panel into one panel. The panel makes assumptions about the application that you are creating. An application that is created by using the Job Description panel, or that has the restrictions that are enforced by that panel, is called a **job description**.

It contains no more than three operations:

- One computer or WTO operation
- Two other operations, which can be either manual completion or JCL preparation operations

The job name of the computer operation is the same as the job description ID.

Job description example

© Copyright IBM Corporation 2015

Job description example

Use Tivoli Workload Scheduler for z/OS ISPF panel option **1.8.2** to create a job description with the following fields:

- **JOBNAME - TEXT:** Specify the name of the job. Use 1 to 8 alphanumeric characters; the first character must be alphabetic or national. You can also specify a description of up to 24 alphanumeric characters. This field is required.
 - **OWNER: ID - TEXT:** Specify the name of the owner, by using 1 to 16 characters. You can also specify a description of up to 24 alphanumeric characters. This field is required.
 - **CALENDAR ID:** Specify the name of a calendar. Use 1 to 16 alphanumeric characters; the first character must be alphabetic or national. Calendar ID cannot be specified for a job in an application group. If no name is specified, the name DEFAULT is used. This field is optional.
 - **AUTHORITY GROUP ID:** Specify the name of the authority group for the job. Use 1 to 8 alphanumeric characters; the first character must be alphabetic or national. This field is optional.
 - **VALID FROM - TO:** Specify the date when the job is available for scheduling and can be run. Use the YY/MM/DD format. This field is required.
 - **DURATION:** Specify the estimated duration of the job in the *HH.MM.SS* format. This field is required.
 - **RUN TIME FROM - TO:** Specify the time when the job is expected to start and to end in the *HH.MM* format. This field is required if the job is specified as time-dependent; otherwise, it is optional.

- **TIME DEPENDENT:** Specify whether the submit or release of a job depends on a specified time. If **Y** (yes), start the job according to the input-arrival time of the operation. If **N** (no), start the job as soon as all predecessors are completed and the resources are available. This field is optional.
- **WORK STATION:** Specify the name of an automatic computer workstation or a general workstation with the WTO indicator set to **Y** (yes). Use 1 to 4 alphanumeric characters; the first character must be alphabetic or national. This field is required.
- **PRIORITY:** Specify the priority, where **1** = low, **2-7** = medium, **8** = high, and **9** = urgent. This field is required.
- **JCL PREPARATION:** Specify **Y** (yes) and type a general workstation name with attribute **S** (start and completion only), or specify **N** (no) and leave the workstation field blank. Use 1 to 4 alphanumeric characters; the first character must be alphabetic or national. This field is optional.
- **HIGHEST RETURN CODE:** Specify a numeric value from 0 to 4095 that is the highest return code for a job *not* to be reported as ended-in-error. This field is optional.
- **MANUAL INTERACTION:** Specify manual work that you must complete before the job is released. The manual work description can have a maximum of 24 characters.
- **MANUAL WS:** Specify the name of a manual workstation. The field is optional when no manual interaction is required. Use 1 to 4 alphanumeric characters; the first character must be alphabetic or national. The workstation must be a manual workstation with attribute **C** (completion only).
- **RUN CYCLES:** Specify the name of a period and an offset number. For the period name, use 1 to 8 alphanumeric characters; the first character must be alphabetic or national. For the offset, use 1 to 3 numeric characters in the range -999 to -1, or 1 - 999. This field is optional. A plus sign (+) indicates there are run cycles that cannot be shown on the panel. You cannot specify this field cannot be specified for a job in an application group.
- **PREDECESSORS:** Specify up to five predecessors or use global search characters (*) and (%) to define groups of predecessors. Use 1 to 8 alphanumeric characters; the first character must be alphabetic or national. This field is optional. A plus sign indicates that you have predecessors that cannot be shown on the panel.
- **SPECIAL RESOURCES:** Specify resources that are used by the job. This field is optional. For each allocation, specify as follows:
 - A resource name that has 1 to 44 characters.
 - A quantity from 1 to 999999 or blank. Blank means that the entire quantity must be available when the job starts.
 - The allocation type, **S (shared)**, or **X (exclusive)**.
 - A Keep On Error value. **Y** (yes) indicates keeping the allocated quantity. **N** (no) indicates freeing the allocated quantity. **Blank** (default) indicates that it uses the ON ERROR value in the database definition of the resource. If this is also blank, the value of the **ONERROR** keyword of **RESOPTS** is used.

- An Avail On Complete value indicates how to set global availability at job completion: **Y** (yes), **N** (no), **R** (reset), or **blank**. Blank means that system defaults are used.
- A plus sign indicates more resources that cannot be shown on the panel.
- **GROUP DEFINITION:** Specify the group definition that is used for the calendar and generation of run cycles.
- **Smoothing Factor:** Specify a value from 0 to 999. The smoothing factor (DSF) controls how actual deadline data is used and determines the level of feedback of the new estimated deadline.

$$\text{NDL} = \text{ODL} + ((\text{ADL} - \text{ODL}) * \text{DSF}/100)$$

- **Feedback Limit:** Specify a value from 100 to 999. The feedback limit (DFL) establishes the limits that feedback data (actual deadline) is considered normal and acceptable. Outside this limit, the value is ignored. It is calculated as follows:
 - Lower limit: **ODL * 100/DFL**
 - Upper limit: **ODL * DFL/100**

Student exercises



See the Student Exercise Guide Unit 4

© Copyright IBM Corporation 2015

Student exercises

Perform the exercises for this unit.

Summary

- Describe the parts of an application
- Describe the types of applications
- Create applications with or without run cycles
- Create operations within an application
- Create a job description

© Copyright IBM Corporation 2015

Summary

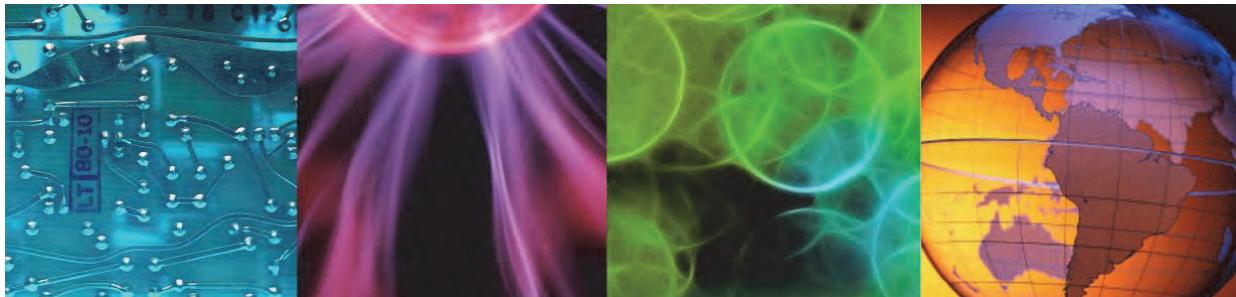


5 Operation submission, throughput, and monitoring

IBM Tivoli Workload Scheduler for z/OS 9.2.0



5 Operation submission, throughput, and monitoring



© Copyright IBM Corporation 2015

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

In this unit, you learn when and how applications and operations are selected for submission in Tivoli Workload Scheduler for z/OS. The unit also covers how operations are managed to help maximize throughput by using several Tivoli Workload Scheduler for z/OS options.

References:

- SC32-1262 *IBM Tivoli Workload Scheduler for z/OS Managing the Workload*
- SC32-1265 *IBM Tivoli Workload Scheduler for z/OS Customization and Tuning*

Objectives

- Describe the process that Tivoli Workload Scheduler for z/OS uses to select operations for submission
- Describe the various ways that Tivoli Workload Scheduler for z/OS helps to manage the throughput of operations
- Implement the feedback function and its purpose
- Implement critical path management in Tivoli Workload Scheduler for z/OS
- Describe how Tivoli Workload Scheduler for z/OS integrates with the z/OS Workload Manager

© Copyright IBM Corporation 2015

Objectives

Tivoli Workload scheduler for z/OS decides which job to submit next based on an internal priority. In this unit you discover how this process works and how its critical path management ensures jobs are processed in accordance with SLAs. In the hands-on exercises you learn how to further define job details that help Tivoli Workload Scheduler for z/OS priorities the work and maintain duration times.

Lesson 1 Operation submission and throughput

Lesson 1 Operation submission and throughput overview

- Tivoli Workload Scheduler for z/OS uses several options to help select operations for submission and manage their throughput
- Operation submission and throughput is user-implemented through ISPF panels and initialization parameters
- The goal is meeting the intended batch workload schedule

© Copyright IBM Corporation 2015

This lesson provides an overview of how Tivoli Workload Scheduler for z/OS decides why and when operations are selected to run. After completing this lesson, you should be able to describe in general how Tivoli Workload Scheduler for z/OS decides why and when operations will be selected to run.

Many factors, such as the associated run cycle, predecessor completion, resource availability, deadline time, and priority influence when an operation runs. Tivoli Workload Scheduler for z/OS provides several features and options that can help manage the efficient throughput of operations.

The scheduler calculates an operation's **latest out time**, which is the latest possible time that the operation can start for meeting the deadline time. Operations that run close to their deadlines are given priority when competing for resources. If possible, define accurate resource use and deadline times in the application description. Tivoli Workload Scheduler for z/OS can best schedule your work and optimize the use of resources when they are accurate.

Planning the operation run order

1. The daily planning function designates the operations that are eligible for submission
2. Daily planning builds a list of work that is ready to be started on each computer workstation and non-reporting workstation (This list is called the ready list)
3. Tivoli Workload Scheduler for z/OS selects the most urgent operation for the resources that are available from all the ready lists
4. The single most urgent operation is selected to start

© Copyright IBM Corporation 2015

Planning the operation run order

The scheduler builds the current plan from information in the long-term plan, calendar database, application description database, workstation database, and resource database. When the current plan is created, the scheduler assigns start times to each operation. These start times are the scheduler estimates of when the operations start. They are not always the actual times that the scheduler uses to start the operations. This time is used only when the operation is designated as a time-dependent operation. The scheduler maximizes throughput in your system by starting operations at the earliest possible times within the current plan.

Before assigning which operation to start, the daily planning function first builds queues of work that is ready to be processed at each workstation. These queues are lists of operations that are ready to be run, that is, operations with no outstanding predecessors. The operations are placed on the workstation ready list in the order in which the scheduler calculates they run. They are sorted in the following order:

1. Operations that have the urgent flag set.

When an operation is defined with priority 9, the scheduler automatically sets the urgent flag. The urgent flag is also set when an operation has missed its deadline.

2. Earliest latest start time.

The latest start time is the latest time that the operation can start so it meets the deadline. This calculation involves the operation deadline, estimated duration, resource requirements, and successor processing. When the scheduler creates the current plan, it calculates the latest start time for all the operations in a chain of dependencies. It starts at the last operation and works back when it does the calculation.

3. Operation priority (other than 9).
4. Shortest estimated duration.

Operations that are time-dependent or waiting for resources to become available are not included in the ready lists.

Operation throughput management options

- Provides application deadline and operation duration feedback (dynamic feedback)
- Assigns application priority
 - Specified at the application level
 - Might affect time for an operation to be submitted
- Critical Path Management affects the current scheduling and priority of operations on a critical path
- With Workload Manager Scheduling Environment, you can schedule operations based on the availability of z/OS Workload Manager scheduling environments

© Copyright IBM Corporation 2015

Operation throughput management options

You can select Tivoli Workload Scheduler for z/OS options that can help improve operation throughput on your system. You can use either Tivoli Workload Scheduler for z/OS initialization statements or options that you specify in the Tivoli Workload Scheduler for z/OS ISPF panels.

Activating job submission

The Tivoli Workload Scheduler for z/OS parameter JTOPTS JOBSUBMIT(NO) specifies that job submission is turned off. Therefore, automatic job submission is initially inactive after you start Tivoli Workload Scheduler for z/OS.

You can activate automatic job submission after startup by using the Tivoli Workload Scheduler for z/OS ISPF option **9.2**. The Tivoli Workload Scheduler for z/OS option **9** panel includes many service functions. Select option **2** from this panel to activate job submission.

Lesson 2 Dynamic feedback overview

Lesson 2 Dynamic feedback overview

Two kinds of feedback

- Deadline feedback for run cycles and operations
- Duration feedback for operations

User-invoked and optional

- Specify controller initialization statement parameters: Global Defaults
- Specify application specific deadline feedback options using the ISPF panels
- Specify operation specific duration feedback options using the ISPF panels

Attempts to fine-tune operation duration times and application deadlines

Accurate operation duration times and run-cycle deadlines are an important aspect of effective operation throughput management

© Copyright IBM Corporation 2015

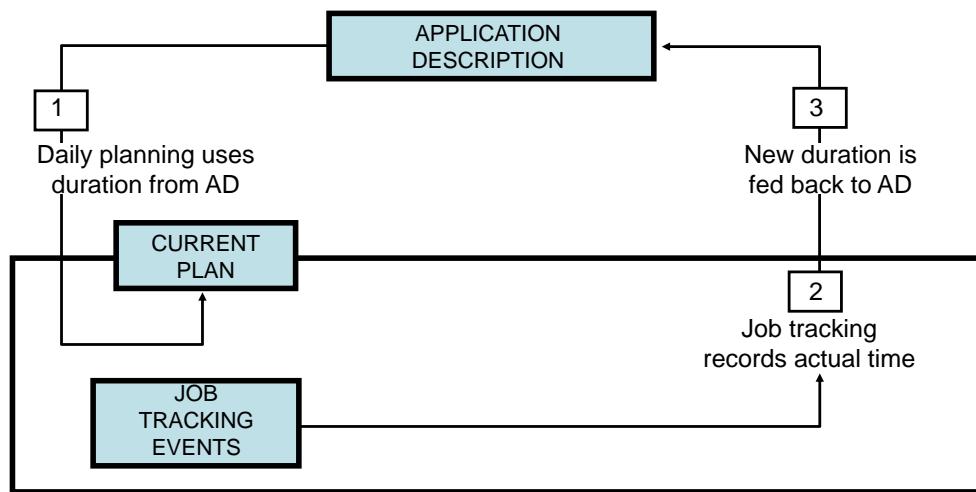
In this lesson, you learn how to define dynamic feedback options. After completing this lesson, you should be able to define dynamic feedback options.

Tivoli Workload Scheduler for z/OS uses the estimated durations of operations to calculate the start times for operations in the current plan. The planned start times that are generated are as accurate as the durations set for the operations. You can see these times in online views and the daily planning reports.

Tivoli Workload Scheduler for z/OS can automatically adjust the durations of operations in the application database by capturing the actual run times of the operations. This optional feature is known as **dynamic feedback**. The feature requires either initialization parameters or ISPF options.

Tivoli Workload Scheduler for z/OS also uses the estimated deadlines of run cycles, operations, and job descriptions to calculate a new deadline. The new deadline is then stored in the application database.

Dynamic feedback flow: Duration feedback



© Copyright IBM Corporation 2015

Dynamic feedback flow: Duration feedback

This flow for operation-duration feedback is similar to that of run-cycle deadline feedback. The application database dynamically updates for both run-cycle deadline feedback updates and for operation-duration feedback updates. The deadline in the application description run cycle or operation updates when an occurrence of the application reaches complete (C) status.

Dynamic feedback helps to ensure that Tivoli Workload Scheduler for z/OS scheduling remains accurate by continually updating the operation run times in the AD database.

Guided by user-specified criteria, the dynamic feedback function uses valid actual run times. The actual run times are recorded by job tracking. The actual run times are also used for recalculating the estimated durations for operations and for updating the application database with the new durations. When updates are necessary, they occur during the daily planning batch process.

If the actual run time duration for an operation falls outside the user-specified criteria window, Tivoli Workload Scheduler for z/OS does not update the duration. Instead, it generates a report that is called a **missed feedback report**. The report lists all operations with run times outside the limits and also the estimated and actual run times for each of those operations.

Dynamic feedback initialization parameters

Duration feedback defaults

- LIMFDBK sets range criteria for acceptable deadlines
- SMOOTHING uses a percentage of the difference between actual duration (AD) and the old duration (OD)

Deadline feedback defaults

- DLIMFDBK sets range criteria for acceptable deadlines
- DSMOOTHING uses a percentage of the difference between the actual deadline (ADL) and the old deadline (ODL)

JTOPTS ...
LIMFDBK(100)
SMOOTHING(50)

JTOPTS ...
DLIMFDBK(100)
DSMOOTHING(50)

© Copyright IBM Corporation 2015

Dynamic feedback initialization parameters

Two controller initialization parameters, the deadline smoothing factor (DSMOOTHING) and the deadline limit for feedback (DLIMFDBK) provide a global default to control the use of measured deadlines. By using the ISPF panel options, you override these two parameters.

Two other controller initialization parameters, the duration smoothing factor (SMOOTHING) and the duration limit for feedback (LIMFDBK) provide a global default to control the use of measured durations. By using the ISPF panel options, you can override these two parameters.

These parameters are keywords in the JTOPTS initialization statement of the controller. The default values are 100 for LIMFDBK and DLIMFDBK and 50 for SMOOTHING and DSMOOTHING. The default value of 100 for limit of feedback means that no feedback is done. Any value that you specify in the Creating an Application or Operation Details Feedback panels overrides the installation defaults specified in JTOPTS.

The deadline feedback limits for ADL are calculated as follows:

- Lower limit: **ODL * 100/DLF**
- Upper limit: **ODL * DLF/100**

For the smoothing calculation, the new deadline is calculated as follows:

$$\text{NDL} = \text{ODL} + ((\text{ADL} - \text{ODL}) * \text{DSF}/100)$$

The deadline smoothing factor determines how much a measured deadline changes existing values in the application description database. Both run-cycle and operation-level values can be affected.

The formulas for operation duration feedback limits and smoothing are the same as for deadline feedback. However, they require duration values instead of deadline values.

Operation details feedback options

In [Unit 4, “Applications”](#) on page 87, you learned that you can specify the deadline smoothing factor and deadline feedback limit on the Creating an Application panel. You can specify the operation duration feedback limit and operation duration smoothing factor on the operation detail Feedback Options panel.

In the remainder of this lesson, operation duration feedback is used to provide examples and teach the process of dynamic feedback. However, the concepts that are provided in the remaining slides apply to deadline feedback as well.

Operation feedback limit and smoothing factor

```
EQQAMFBP ----- FEEDBACK OPTIONS -----
Command ==>

Enter/Change data below:

Operation           : INCP 005
SMOOTHING FACTOR   ==> 50          A value 0 to 999 where 0=no smoothing
FEEDBACK LIMIT      ==> 200         A value 100 to 999 where 100=no feedback
```

Examples of Feedback values:

- **100** = No new estimated duration is stored in the application description database
- **110** = The new estimated duration is stored if the measured duration is approximately between 90% and 110% of the old estimated duration
- **200** = The new estimated duration is stored if the measured duration is approximately between half and double the old estimated duration

Examples of Smoothing Factor values:

- **0** = There is no feedback used
- **50** = The new estimated duration is the old estimated duration, plus 1/2 of the difference between the measured and old estimated duration
- **100** = The measured duration replaces the old estimated duration

© Copyright IBM Corporation 2015

Operation feedback limit and smoothing factor

Select ISPF option 5 FEEDBACK on the OPERATION DETAILS panel.

Feedback

The limit for feedback is a number from 100 through 999. The feedback limit value establishes the limits for measured run times to be considered valid. A value of 0 causes Tivoli Workload Scheduler to use the duration for feedback regardless of measured value.

A value of **100** tells Tivoli Workload Scheduler for z/OS to disregard all run times and never update the application database. A value of **999** defines the most latitude in run times. The scheduler accepts any run time from 1/10th to 10 times the estimated duration. If a measured run time falls outside the limits, it is ignored, and the application description database does not update.

Smoothing

Excluding exceptional run time collections by using Limit for Feedback does not prohibit differences in scheduled run times. Run times are likely to skew back and forth.

Tivoli Workload Scheduler for z/OS can smooth out wild variations in durations by using the actual run time. The actual run time is compared with previous estimated run times. A new time is calculated based on the difference and a parameter that is specified by the user.

Use the Smoothing Factor parameter to dampen extreme changes in run times. The smoothing factor is a number 0 - 999. The SMOOTHING value indicates how much a measured duration changes existing values in the application description database:

- A value of **0** tells the scheduler to disregard all run times and never update the application database.
- Values **1 - 99** tell the scheduler to replace the estimated duration. It uses the sum of the old duration and a percentage of the difference between the actual run time and the old duration.
- A value of **100** tells the scheduler to replace the estimated duration with the actual run time.
- A value of **999** tells the scheduler to replace the estimated duration with a longer duration. It uses the sum of the old duration and 10 times the difference between the actual run time and the old duration.

Feedback and smoothing formula terminology

Term	Description
ND	The new estimated duration to be stored in the application description database
OD	The old estimated duration stored in the database
AD	The actual measured duration
SF	The smoothing factor $ND = OD + ((AD - OD) * SF / 100)$
LF	The limit for duration feedback Lower limit = $OD * 100 / LF$ Upper limit = $OD * LF / 100$

© Copyright IBM Corporation 2015

Feedback and smoothing formula terminology

These terms are used in the smoothing and feedback limit formulas for operation durations. *Duration* can also be called **run time**.

The operation duration feedback limits for a valid actual duration (AD) are calculated as follows:

- Lower limit: **OD * 100/LF**
- Upper limit: **OD * LF/100**

For the smoothing calculation,

the new duration is calculated as follows:

$$ND = OD + ((AD - OD) * SF/100)$$

Run cycle deadlines have a different set of terms. Operation durations directly affect run cycle deadlines. If operation durations change, the deadlines for associated run cycles change.

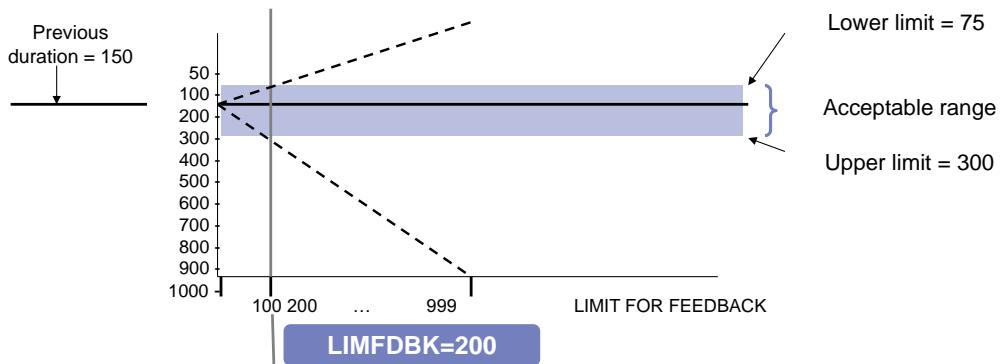
The deadline formula terminology is in the following list.

- **NDL** is the new estimated deadline for the occurrence or operation. It is considered to be an offset in minutes from the IA. It is stored as the new deadline in the AD database.
- **ODL** is the old estimated deadline for the occurrence or operation.
- **ADL** is the actual deadline time, the elapsed time in minutes between the IA and the occurrence or operation completion time.
- **DSF** is the deadline smoothing factor.
- **DLF** is the feedback limit for deadline feedback.

Operation duration feedback example

Define the limits for duration feedback to be kept

- Range: 100 – 199
- Default: 100 (same value as before)



Example: When **LIMFDBK = 200**, disregard the operation durations feedback that is greater than twice or less than half of previous duration (>300 and <75)

© Copyright IBM Corporation 2015

Operation duration feedback example



Note: The graphs in this slide and the next represent approximations; they are not exact.

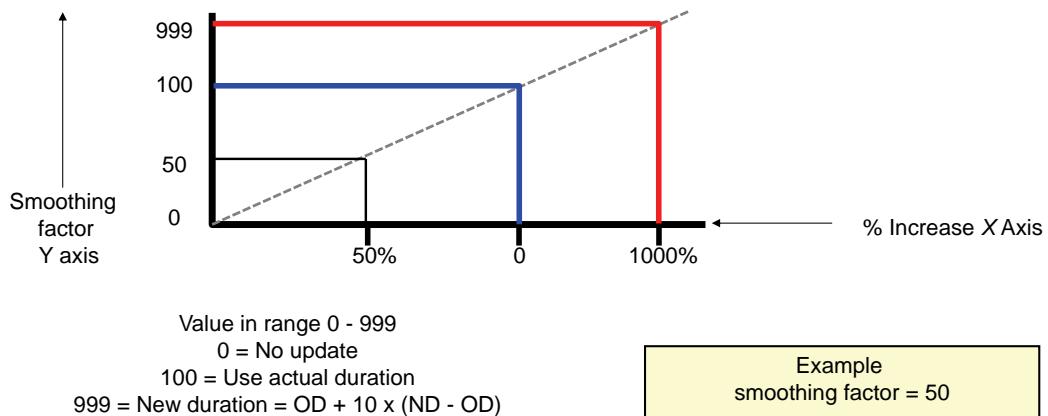
A starting value for your installation-wide feedback limit is **200**. A feedback limit of 200 tells Tivoli Workload Scheduler for z/OS to accept any duration between one-half and twice the estimated duration of operations. If the duration is accepted, it is eligible to have the smoothing factor applied. The database is updated with the new smoothed duration.

In this example, the center line is the previous run time of 150 and the feedback value is 200. These values create a lower limit of 75 and an upper limit of 300. The x-axis shows the feedback limit parameters. Run times that are above the lower limit or below the upper limit in the graph are disregarded.

The two dashed lines in the graph represent the possible lower and upper limits as the feedback limit changes. Based on the feedback limit formula, the slopes of the two dashed lines in the graph differ, indicating that the feedback limit affects the upper limit more than the lower limit. The slope of the upper limit is steeper. As you increase the feedback limit, the upper limit for acceptable operation durations increases at a greater rate than it does for the lower limit.

Smoothing factor example

New duration = Old duration + (Actual duration - Old duration) x SF/100



For example, ND is OD plus 50% of the difference between OD and AD

© Copyright IBM Corporation 2015

Smoothing factor example

The default value for an installation-wide smoothing factor is 50. Using this default to calculate a new duration, Tivoli Workload Scheduler for z/OS replaces the old duration. The new duration is the old duration plus 50% of the difference between the actual duration and the old duration.

As shown in the diagram, a smoothing factor of 999 would add almost 1000% of the actual runtime difference to the old duration. This sum is used to create the duration. If the old run time was 100 and the actual run time was 150, the new run time is close to 600:

$$\text{New Duration} = 100 + (150 - 100) * 999/100 = \sim 600$$

Getting started capturing operation durations

Steps for capturing actual durations:

1. Use workstation operation default times
2. Set DSMOOTHING(100) to feedback actual durations
3. Set DLIMFDBK(999) to accept all values between 1/10 x ODL and ODL x 10
4. Let all applications run at least once
5. Modify the smoothing factor and limit for feedback as required at the application and operation level to attain desired values
6. Monitor and adjust as required

© Copyright IBM Corporation 2015

Getting started capturing operation durations

Use dynamic feedback from the beginning. Use the best estimates possible in the initial setup, and then try these values:

- A feedback limit of 999
- A smoothing factor of 100

After all your applications process, you can change the feedback limits to your own preferred settings. Tivoli Workload Scheduler for z/OS take account of the changes in run times as applications grow and hardware becomes more efficient.

Lesson 3 Operation priority

Lesson 3 Operation priority

- Inherited from the application general information definition
- Required for nongroup application definitions
- Priority values
 - 1: low
 - 2-7: medium
 - 8: high
 - 9: urgent

© Copyright IBM Corporation 2015

This lesson discusses operation priority values. After completing this lesson, you should be able to describe the use of operation priorities.

An operations priority value is inherited from the application general information. A sample scenario of Tivoli Workload Scheduler for z/OS that uses operation priority is as follows:

1. If Tivoli Workload Scheduler for z/OS detects two operations in Ready status, it checks whether either operation is marked as urgent.
2. If neither is urgent, the scheduler compares their latest-start times.
3. If the latest-start times are equal, the scheduler examines the priority.
4. If the priority is also equal, the scheduler first starts the operation that has the shortest estimated duration.

Tivoli Workload Scheduler for z/OS can dynamically increase the priority of an operation on a critical path.

Lesson 4 Critical operations

Lesson 4 Critical operations

- Is user-defined in ISPF panels, is highly important, and must run on schedule
- Receives special attention, including run-time scheduling adjustments
- Creates a critical path
 - Other operations that can affect the completion of the critical operation are also considered critical
 - End-to-end distributed operations are included
 - The critical path is recalculated when it changes
- Can have more than one critical operation: Each critical path target creates a separate critical path

© Copyright IBM Corporation 2015

In this lesson, you learn how to define critical operations and understand how critical operations are used. After completing this lesson, you should be able to define and understand critical operations.

The Tivoli Workload Scheduler for z/OS ISPF panels, batch loader, and programming interface can be used to create and modify operation definitions in the application database. The scheduling user can use any of these interfaces to define an operation as critical.

Most batch has within it a number of jobs who need to complete by a specific time. This is normally because they precede an operational Service Level Agreement (SLA), the starting of an online service, the availability of a database, the creation of a file to be sent to an external party. Jobs that can incur a penalty for the business if they do not run on time. These jobs, if they do not meet their schedules, are good candidates to define as critical end points. The scheduler calculates the critical path to a critical operation and if it detects that during the batch execution that a delay or failure in that path might affect the critical job, it moves the risk level of the critical job from Normal to Potential or High and issues a message to alert the operators.

The scheduler calculates the critical path to those jobs on the path that have the most risk of affecting the critical deadline of the operation. It automatically promotes jobs on the critical path that are late.

Defining critical operations

CRITICAL options

- **P:** Critical path target
 - Is a critical operation of high importance that must run on schedule
 - A critical path is built by the daily planning batch process
 - Can be assigned to a Workload Manager service class
- **W:** Eligible for Workload Manager assistance
 - Can be assigned to a Workload Manager service class
 - No critical path is built
- **N:** Not a critical job

Workload Manager options

- Workload Manager policies
 - Long duration
 - Deadline exceeded
 - Latest start time exceeded
 - Conditional
- Workload Manager service class

```
EQQAMJBP ----- JOB, WTO, AND PRINT OPTIONS -----
Command ==>
Enter/Change data below:
Application   : U#CRITA          TEAM # Critical App
Operation     : U#CP 020 U#CRITJ4
JOB CLASS     ==> _
ERROR TRACKING ==> Y           Job class of corresponding job
HIGHEST RETURNCODE ==> _
EXTERNAL MONITOR ==> N           Y means error automatically tracked
CENTRALIZED SCRIPT ==> N         Highest return code not in error
COND RECOVERY JOB ==> N         Job monitored by external product (Y/N)
CRITICAL      ==> P           Centralized script Y/N (for FTW only)
POLICY ==> _ CLASS ==> _          Recovery job for a cond predecessor (Y/N)
                                Critical job: N P W
                                WLM Policy and Service Class
```



© Copyright IBM Corporation 2015

Defining critical operations

Critical path targets are operations that are critical to your business and must be completed by their deadline time. The daily plan job calculates a critical path for these target operations. The critical path consists of a specific dependency network that includes internal and external predecessors.

The scheduler calculates the planned end time for each critical path target during the planning process based on the information you set for the operation or occurrence (for example, the IA time, duration, and deadline). The scheduler also considers workstation parallel servers, open intervals, special resources, and their availability. For this reason, the more accurate the application definition is, the more accurate the critical path calculation. To have the scheduler update your estimates in the application description database with actual run values, implement dynamic feedback.

The scheduler integrates with the z/OS Workload Manager to help improve operation throughput. You can assign critical operations to Workload Manager service classes based on a specified policy. You can assign any operation to a Workload Manager scheduling environment.

The scheduler increases the internal scheduling priority of a ready critical path operation. Scheduling priority is increased when an operation is approaching its latest start time without being started. On already started operations, the current Workload Manager policy is applied and the service class assigned.

Required controller initialization parameter for Workload Manager activation is as follows:

OPCOPTS WLM(class name|WLMJOBPR, policy name, CONDITIONAL(20))

- **CRITICAL:** Specify whether the operation is considered critical and eligible for Workload Manager assistance if it is late.

- **P:** Critical path target.
- **W:** Eligible for Workload Manager assistance but no critical path to be built.
- **N:** Not critical or eligible for Workload Manager assistance.

If you specify **P**, the operation is considered critical and the daily plan job generates a critical path.

If you specify **W**, Tivoli Workload Scheduler for z/OS automatically sends a request to the Workload Manager to promote a job or started task into the defined service class. The service class must be one that is defined for batch jobs in the Workload Manager environment. The scheduler sends the request if the conditions of the specified assist policy are met.

If you specify **N**, the job is not critical.

- **POLICY:** Specify the policy to use for Workload Manager assistance if the job is defined as critical.
 - **L** (Long duration): The job is assisted if it runs beyond its estimated duration time.
 - **D** (Deadline): The job is assisted if it is not finished when its deadline time is reached.
 - **S** (Latest start): The job is assisted if it is submitted after its latest start time.
 - **C** (Conditional): An algorithm calculates whether to apply the Deadline or Latest start policy.
- **CLASS:** Specify the Workload Manager service class.

The Workload Manager parameter of the controller OPCOPTS statements must be specified to activate Workload Manager integration. This statement provides the global defaults for the service class and Workload Manager policy.

Running the critical path

- Applies to both z/OS and end-to-end operations
- If a ready operation on a critical path might miss its latest start time, the internal scheduling priority is raised to the highest possible value
- If a started operation is late, the operation is promoted to the Workload Manager service class according the following conditions:
 - The CRITICAL option is set to W or the operation is on the critical path. (CRITICAL option is set to P for some operations)
 - The Workload Manager keyword of the OPCOPTS statement is set
 - The requirements of the Workload Manager policy to be applied are met

© Copyright IBM Corporation 2015

Running the critical path

Tivoli Workload Scheduler for z/OS provides special attention for operations that have the CRITICAL option set to **W** or **P**.

When set to **W**, the operation does not cause a critical path to be generated. Option **W** functions in much the same way as the CRITICAL option **Y** did in Tivoli Workload Scheduler for z/OS 8.2. The operation is deemed critical in the sense that it is eligible for Workload Manager service class assistance.

When set to **P**, the operation is considered critical and the Daily Plan job generates a critical path. All of the operations on the critical path are eligible for automatic scheduling priority increases and Workload Manager assistance.

Monitoring the critical path

BROWSING CRITICAL PATH (left part)										Row 1 of 3	
										Scroll ==>	CSR
Command ==> -											
Application : U#CRITA											
Operation : U#CP 20											
WLM Class and Policy :											
Enter the row command S to select an operation for details.											
Press ENTER to refresh.											
Row cmd	Application id ws	Operation no.	Jobname	Promot U W	Flags L R	S	P	Latest start	Input arrival		
....	U#CRITA	U#CP 005	U#CRITJ1	Y N	Y N	E	7	27 07.13	27 07.00		
....	U#CRITA	U#CP 015	U#CRITJ3	N N	Y N	W	7	27 07.13	27 07.00		
....	U#CRITA	U#CP 020	U#CRITJ4	N N	N N	W	7	27 07.14	27 07.00		
***** Bottom of data *****											

ISPF option =6.7

Late predecessor

© Copyright IBM Corporation 2015

Monitoring the critical path

You can use ISPF option **=6.7** to monitor the critical path of a critical path target operation. In this example, the critical path target is operation 020. It is waiting on operation 015, which is late because operation 005 ended in error. Both operations 005 and 015 are in the critical path.

The Browsing Critical Path panel has a left side and right side. You can monitor two lists as follows:

- A list of all critical jobs that have the CRITICAL option set to P
- A sorted list of critical dependencies for each critical job

Critical path monitoring information is also available in the Tivoli Dynamic Workload Console and the Tivoli Enterprise Portal. You can also filter for missing deadline messages that are issued only for operations on critical paths:

BATCHOPT CRITOPMSG(S(No|Yes))

where

- NO is the default value.
- YES means that deadline missed messages are issued for critical path operations only.

Additional critical path considerations

Accurate deadline and duration times are important when calculating the critical path, consider these things when defining critical path operations:

- Implementation of the deadline and duration feedback options
- Availability of critical path reports
- Availability of critical path monitoring

© Copyright IBM Corporation 2015

Additional critical path considerations

The deadline feedback options determine how deadlines in the application description database are adjusted. The deadline feedback options can improve the accuracy of the deadlines. More accurate deadlines can improve the critical path calculations.

Running critical path reports

- A REPLAN, EXTEND, or TRIAL job creates a critical path report if there is critical job data to report
- The report lists all of the critical jobs
- Each job in the list has a CRIT TYPE value of ORIG or PRED
- CRIT TYPE of ORIG defines the job as the originating job of the critical path (the CRITICAL option is set to P)
- CRIT TYPE of PRED defines the job as a predecessor of a critical job
 - As a predecessor of a critical job, it is also identified as critical
 - A critical job's predecessors are identified when the critical path is built at daily plan creation

© Copyright IBM Corporation 2015

Running the critical path reports

The report is the result of running the daily planning batch job. It is created in the SYSPRINT DD output of the last job step in the daily planning batch job, also known as the current plan batch job.

Critical path report example

CRITICAL PATH REPORT					Line 00000496 Col 001 080	Scroll ==> CSR
CRIT PATH TYPE	APPL ID	APPLICATION INPUT	OPER ID	PLANNED END	LATEST OUT	
ORIG 000001	U#CRITA	140227 07.00	U#CP_020	140227 07.01	140227 07.14	
PRED 000001	U#CRITA	140227 07.00	U#CP_005	140227 07.00	140227 07.13	
PRED 000001	U#CRITA	140227 07.00	U#CP_015	140227 07.00	140227 07.13	
LISTINGS FROM SAMPLE						
TWSZ						
SUBSYSTEM TWCK						

© Copyright IBM Corporation 2015

Critical path report example

This example shows the first operation with an OPERATION ID of TED1_004 and its CRITICAL option set to **P**. The other operations in the critical path are predecessors. The CRIT TYPE of **ORIG** indicates that it is the originating critical path operation.

Lesson 5 Workload Manager scheduling environments

Lesson 5 Workload Manager scheduling environments

- Workload Manager is a component of z/OS that handles dynamic workload management
- In Workload Manager scheduling environments, users can set up resource requirements across multiple systems
- Workload Manager scheduling environments are a distinct feature and used separately from Workload Manager service classes

© Copyright IBM Corporation 2015

In this lesson, you learn how to define and use Workload Manager scheduling environments in Tivoli Workload Scheduler for z/OS. After completing this lesson, you should be able to define and use Workload Manager scheduling environments in Tivoli Workload Scheduler for z/OS.

Possible benefits of implementing the Tivoli Workload Scheduler for z/OS Workload Manager scheduling environment integration include the following items:

- Automatically resubmitting operations waiting for the scheduling environment.
- Providing automatic tailoring of JCL.
- Monitoring of Tivoli Workload Scheduler for z/OS jobs that are associated with a scheduling environment.
- Aiding in resolution of Workload Manager scheduling environment issues.
- Balancing the loads of scheduled workload.

Supported environments

- Can be a single system or sysplex
- More than one sysplex can be handled by a single Tivoli Workload Scheduler for z/OS controller
- Allows for workload balancing across one or more sysplexes
- Sysplex information is included for monitoring
- More than one JES (job entry subsystem) complex within a sysplex

© Copyright IBM Corporation 2015

Supported environments

Tivoli Workload Scheduler for z/OS interfaces with Workload Manager across the sysplex to detect the status of Workload Manager scheduling environments. It uses both Workload Manager macros and event notification facility (ENF) listening exits to detect the availability of Workload Manager scheduling environments.

Tivoli Workload Scheduler for z/OS checks the status of the Workload Manager scheduling environment that is associated with a job before submission. The availability of the Workload Manager scheduling environment determines whether the job is submitted, when it is submitted, and its status.

The SCHENV= keyword parameter on the JOB statement indicates Tivoli Workload Scheduler for z/OS jobs that integrate with Workload Manager scheduling environments. The parameter has the following characteristics:

- Identifies the Workload Manager scheduling environment to use for a job
- Is automatically added by Tivoli Workload Scheduler for z/OS or manually specified

Workload Manager scheduling environment availability

Before submitting a job, Workload Manager scheduling environment availability is checked

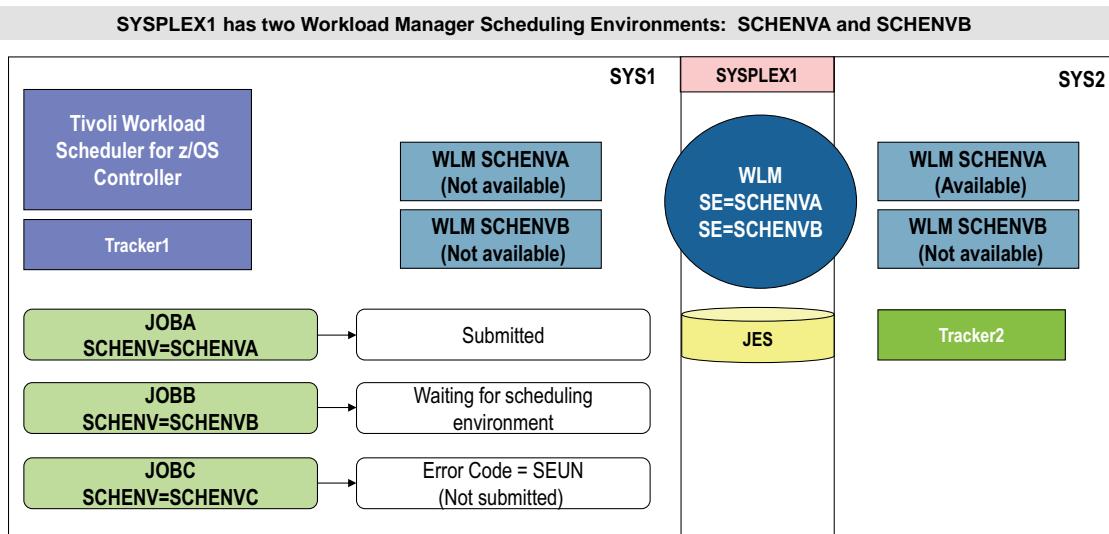
- If the Workload Manager scheduling environment is defined and available
 - The job is submitted
 - Workload Manager identifies the system where the job is to run
- If the Workload Manager scheduling environment is defined but not available
 - The job is not submitted
 - The job is put in waiting for scheduling environment extended status
 - Job submission is retried when the Workload Manager scheduling environment becomes available
- If the Workload Manager scheduling environment is not defined
 - The job is not submitted
 - The job is put in scheduling environment Unavailable status

© Copyright IBM Corporation 2015

Workload Manager scheduling environment availability

Tivoli Workload Scheduler for z/OS uses the defined Workload Manager interfaces to check the availability of Workload Manager scheduling environments. Jobs are not submitted if their defined scheduling environment is not available.

Single sysplex example



© Copyright IBM Corporation 2015

Single sysplex example

The slide shows a sysplex example with job statuses as follows:

- JOBA is submitted and run on system SYS2.
- JOBB must wait for the scheduling environment.
- JOBC ends in error because there is no scheduling environment that is defined by that name.

Setting up a Workload Manager scheduling environment

- Defining a Workload Manager scheduling environment for an operation
 - Specify in the application description database and current plan
You can use the ISPF panels, Batch Loader, Mass Update, Batch Command Interface Tool (BCIT), or PIF
 - Use EQQUX001 and EQQDP001 exits to define for current plan
 - Manually specify the Workload Manager scheduling environment name in JCL
- Activating the Workload Manager scheduling environment integration OPCOPTS SECHECK
- Identifying sysplexes for multiple sysplex configurations OPCOPTS SYSPLEXID
- Identifying system names for multiple JESplex in sysplex configurations OPCOPTS JESPLEX
- Improving performance OPCOPTS SUPPRESSENF

© Copyright IBM Corporation 2015

Setting up a Workload Manager scheduling environment

WLM OPCOPTS parameters are as follows:

- **SECHECK:** Use to activate WLM scheduling environments.
- **NO** (Default): Do not check for WLM scheduling environment.
- **ALL:** Always check for an associated scheduling environment name for all jobs.
 - Checks for a specified name in the current plan for the job.
 - Automatically tailors JCL with the WLM scheduling environment with the SCHENV= JOB card keyword parameter.
 - Checks for the scheduling environment on the JCL JOB card with the SCHENV= keyword parameter. A scheduling environment that is defined at operation level overrides a scheduling environment that is defined with the SCHENV= JOB card parameter.
- **OPERONLY:** Check whether the scheduling environment is specified at operation level in current plan.

Automatically tailors JCL with the WLM scheduling environment with the SCHENV= JOB card keyword parameter.

- **SYSPLEXID:** Use in multiple sysplex configuration. Specify the number that identifies the sysplex where the tracker belongs as an integer number, 1 - 9999. The default is 1.
- **JESPLEX:** Use in multiple JESplex within a sysplex configuration. Provide a list of the system names within the JESplex where the tracker belongs. These system names have 8 characters or fewer.
- **SUPPRESSENF:** Specify if activation of the event notification facility (ENF 57 and ENF 41 listener exits should be suppressed.
 - **YES:** Suppress ENF listener exits.
 - **NO:** Do not suppress ENF listener exits (the default).

Tivoli Workload Scheduler for z/OS user exits, used for defining WLM scheduling environment to the current plan are as follows:

- **EQQUX001:** Job submit user exit routine
- **EQQDPX01:** Daily planning scheduling environment exit routine

The z/OS ENF listener exits can detect changes in WLM scheduling environment. The Tivoli Workload Scheduler for z/OS controller can then act on changes. The notifications can be suppressed on specific trackers by using the OPCOPTS SUPPRESSENF parameter.

- **ENF 41 (EQQZNF41):** Informs the controller when a change in the WLM scheduling environment or definition takes place.
- **ENF 57 (EQQZNF57):** Informs the controller when a WLM scheduling environment becomes available.

Operations waiting on the WLM scheduling environment that just became available are submitted.

Using the SCHENV JCL keyword

Manually

- Use the SCHENV keyword parameter on a JCL JOB card to identify a scheduling environment to Workload Manager

```
//JOBTWSW JOB 1, 'TWSUSER', SCHENV=DB2LATE
```

Automatically

- Specify the Workload Manager scheduling environment in the application description
- Specify the OPCOPTS statement with the SECHECK parameter set to OPERONLY or ALL

© Copyright IBM Corporation 2015

Using the SCHENV JCL keyword

A scheduling environment that is defined at operation level overrides a scheduling environment that is defined with the SCHENV= JOB card parameter. You can specify a WLM scheduling environment at the operation level by using the **Extended info** option from the Operation Details panel.

Monitoring scheduling environment operations

- Using ISPF or Tivoli Dynamic Workload Console
- Providing a feature to track scheduling environment problems
- Waiting for scheduling environment, which is an extended status for operations
- Filtering operations by scheduling environment name
- Scheduling environment name, which has been added as a filtering criterion
- Identifying all operations that are blocked by a specific scheduling environment

© Copyright IBM Corporation 2015

Monitoring scheduling environment operations

The following information is useful during monitoring:

- **Extended status:** Waiting for scheduling environment
- Four tracker events:
 - **IJ4:** Job JCL submission failure because of a new scheduling environment
 - **VS:** Single scheduling environment status changes to YES
 - **VM:** Multiple scheduling environment status changes to YES (WLM query result)
 - **VL:** Log event to track VS or VM events
- **Error codes:**
 - **SEUN:** An operation was not submitted because its WLM scheduling environment is undefined in the sysplex.
 - **SERC:** An operation that was submitted in a restart and cleanup path was not submitted because the required WLM scheduling environment was not available.

Lesson 6 Advanced ISPF panels

Lesson 6 Advanced ISPF panels

- You can choose to use the advanced panels or basic panels
 - Use ISPF option **=0.8** to choose
 - Advanced panels are new with version 8.6
 - Basic panels, those provided in previous versions, are the default
- By using the advanced panels you can perform the following tasks:
 - Simplify the way in which you modify operations in the current plan
 - Enhance the way you view the list of all operations in the current plan
 - Browse comprehensive information about an operation in a single panel

```
EQQXPSTL ----- SETTING PANEL STYLE -----
Command ==> _

Enter Y to use the advanced ISPF panels, or N to use the basic ISPF
panels.

Advanced ISPF panels ==> N
```

© Copyright IBM Corporation 2015

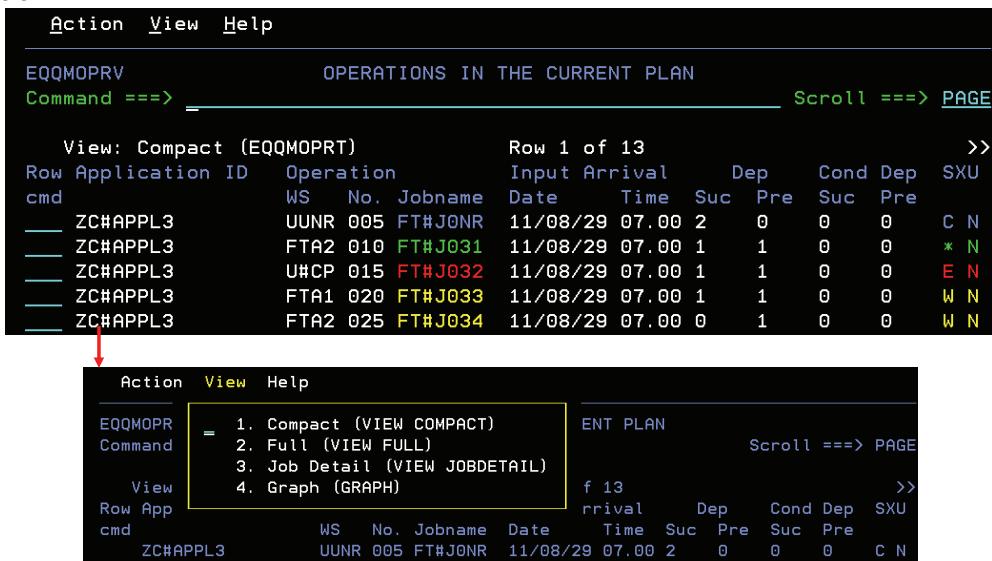
In this lesson, you learn how to initialize and use the advanced ISPF panels feature of Tivoli Workload Scheduler for z/OS. After completing this lesson, you should be able to initialize and use the advanced ISPF panels feature of Tivoli Workload Scheduler for z/OS.

If you chose to use the advanced panels instead of the basic panels, the content and layout of some ISPF panels are different. Several of the advanced panels have a menu bar in which there is a **View** menu. Depending on which view you select, the type and amount of information, changes.

The advanced panels are an optional feature Tivoli Workload Scheduler for z/OS. You enable the advanced panels by using the ISPF option **0.8**. Enter **Y** if you want to work with the advanced ISPF panels for both listing and browsing operations in the CP and application listing and browsing in the AD.

Advanced ISPF panels example

ISPF option =5.3



© Copyright IBM Corporation 2015

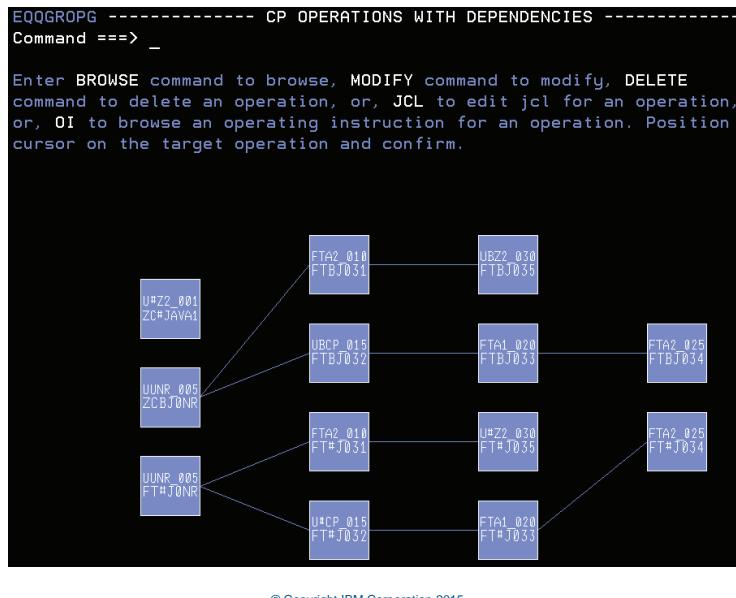
Advanced ISPF panels example

When listing and browsing operations in the current plan, from the **View** menu at the top of the panel you can choose from the following views:

- **Compact** is the default view and displays the concise, summarized information.
 - **Full** is an extended view that provides more in-depth information about dates and time, dependencies, and operation properties.
 - **Job Detail** is the view of detailed, job-centric data such as JCL execution and completion information, timings, and operational data.
 - **Graph** is the graphical view of the operations, from which you can click an operation that you want to browse or modify.

Advanced panels graph view

Graph view



Advanced panels graph view

The graphical display uses GDDM® to display a network of applications or operations graphically. Each node in the network represents an application or an operation, and each line a dependency. The dependencies are always represented from left to right: a dependency to a successor leaves the predecessor node on the right side and enters the successor node on the left side.

The commands select the node closest to the cursor to display, delete, or modify the node. The cursor must be in the graphical area. Enter the command first and set the cursor location after prompted to do so.

You can change the graphical attributes of the network dynamically, and select nodes for display and modification. Use the ATTR command to see the attribute definitions. You can also use the GDDM user control function on the network for zooming, scrolling, and printing.

The commands, GDDM, and ATTR, are available on all graphical displays. In addition, there are functional dependent commands to select nodes from the graph for display, deletion, and modification. The commands have the following characteristics:

- **GDDM:** Enters GDDM user control mode. In this mode you can scroll, zoom, or print the network that uses the GDDM user control functions.
- **ATTR:** Displays a panel where you can change the graphical attributes for the network. The attributes are saved for each individual graphical panel.

Student exercises



See the Student Exercise Guide Unit 5

© Copyright IBM Corporation 2015

Student exercises

Perform the exercises for this unit.

Summary

- Describe the process that Tivoli Workload Scheduler for z/OS uses to select operations for submission
- Describe the various ways that Tivoli Workload Scheduler for z/OS helps to manage the throughput of operations
- Implement the feedback function and its purpose
- Implement critical path management in Tivoli Workload Scheduler for z/OS
- Describe how Tivoli Workload Scheduler for z/OS integrates with the z/OS Workload Manager

© Copyright IBM Corporation 2015

Summary

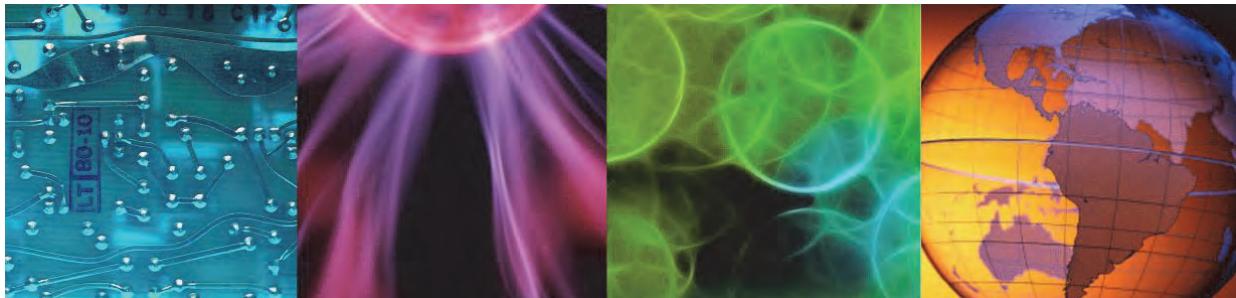


6 Long-term and current plans

IBM Tivoli Workload Scheduler for z/OS 9.2.0



6 Long-term and current plans



© Copyright IBM Corporation 2015

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

In this unit, you learn the relationship between the Tivoli Workload Scheduler for z/OS plans, and how to create and maintain the plans.

References

- SC32-1262 *IBM Tivoli Workload Scheduler for z/OS Managing the Workload*
- SC32-1265 *IBM Tivoli Workload Scheduler for z/OS Customization and Tuning*

Objectives

After completing this chapter, you should be able to perform the following tasks:

- Describe the functions of plans
- Describe the relationship between the databases and the plans
- Describe the relationship between the long-term plan and the current plan
- Implement the long-term plan and the current plan

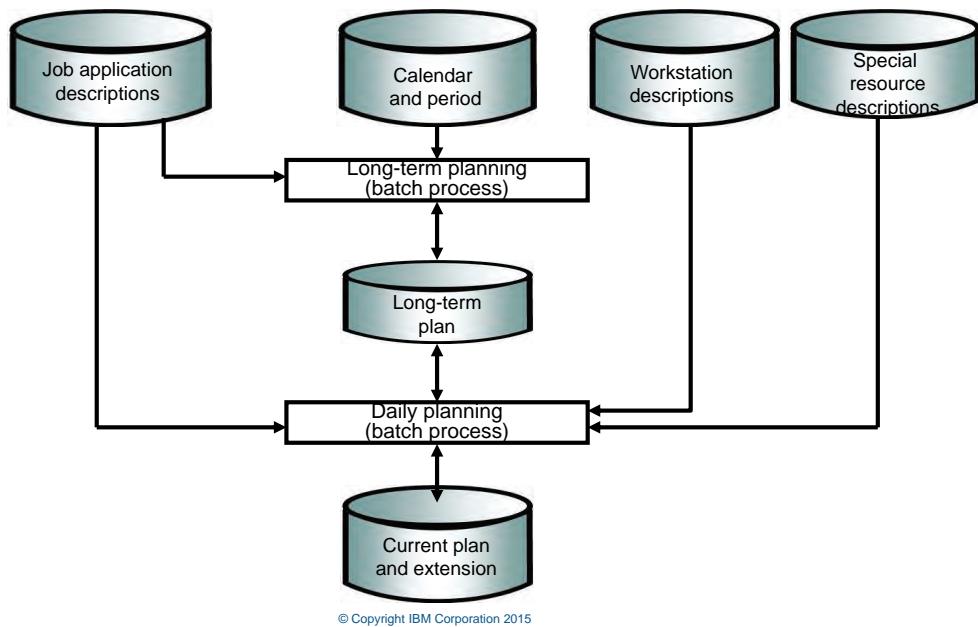
© Copyright IBM Corporation 2015

Objectives

Now that the databases contain the definitions built during the previous units it is possible to create the interactive plans from which the controller submits the jobs that are scheduled to run. The construction of the plans and the jobs that create them are explained. Once the long term and current plans are built you check that your schedules are ready to run. For the hands-on exercise in this unit you monitor and interact with the jobs you built, running them through to completion.

Lesson 1 Data bases and plans overview

Lesson 1 Data bases and plans overview



This lesson provides an overview of the data bases used to store the long-term and current plans. After completing this lesson, you should be able to describe the data bases used to store the long-term and current plans and determine their status.

Tivoli Workload Scheduler for z/OS maintains two plans: the **long-term** plan (LTC) and the **current** plan (CP). The LTP is created from the application, calendar, run cycle group, and period databases.

The CP is created from the LTP, AD, special resource (RD), and workstation (WS) databases. The CP holds occurrences that are created by the daily planning batch function. The online CP file can be one of two alternating data sets. The data definition names or these data sets are EQQCP1DS and EQQCP2DS.

The Tivoli Workload Scheduler for z/OS ISPF panel option **2.4** shows the Status of the Long-Term Plan panel. This ISPF panel shows the LTP status and shows when the current plan ends.

You can select the Tivoli Workload Scheduler for z/OS ISPF panel option **6.6** to enter the Browsing General Current Plan Information panel. This panel shows you which current plan file is in use.

Long-term plan status

```
EQQLSTAP ----- STATUS OF THE LONG TERM PLAN
Command ==>

View data below:

Long term plan start   : 14/03/10
Earliest non-completed
occurrence in
current plan          : 14/03/10
Latest update of
long term plan        : 14/03/10
Current plan end       : 14/03/11 06.00
Long term plan end     : 14/03/13
Completed occurrences
removal shift value    : 0
```

ISPF option =2.4

© Copyright IBM Corporation 2015

Long-term plan status

The slide shows the Tivoli Workload Scheduler for z/OS ISPF panel option **2.4**. This ISPF panel shows when the LTP started and when it ends. The oldest incomplete occurrence in current plan can affect the size of your long-term plan database. The long-term plan retains information about all occurrences after this date and time regardless of status. The panel shows the following items:

- **Long-term plan start:** Shows the earliest date that you can add an occurrence to the LTP if there is no current plan.
- **Earliest non-completed occurrence in current plan:** Shows the earliest occurrence that is not flagged as complete in the current plan. The long-term plan contains all occurrences since the date shown in this field.
- **Latest update of long-term plan:** Shows the date when you last updated the LTP.
- **Current plan end:** Shows the earliest date and time an occurrence can be added to the LTP. If you have no current plan, the earliest date becomes the LTP date.
- **Long-term plan end:** Shows the date when the LTP ends.

General CP information

```
EQQSGCPP ----- BROWSING GENERAL CURRENT PLAN INFORMATION -----
Command ==>
Current plan created      : 14/03/10 18.50
Planning period end        : 14/03/11 06.00

Backup information:
Last CP backup             : 14/03/10 18.51
First logged event after backup : 14/03/10 18.51   Time stamp: 0114069F 18513442

Daily planning status:
Under production           : No
NCP ready                  : No
Symphony status:
Symphony run number         :
Under production           :
New Symphony ready          :

In use ddname of:
Current plan                : EQQCP2DS
Job-tracking log             : EQQJT03
JCL repository               : EQQJS1DS
```

ISPF option =6.6

© Copyright IBM Corporation 2015

General CP information

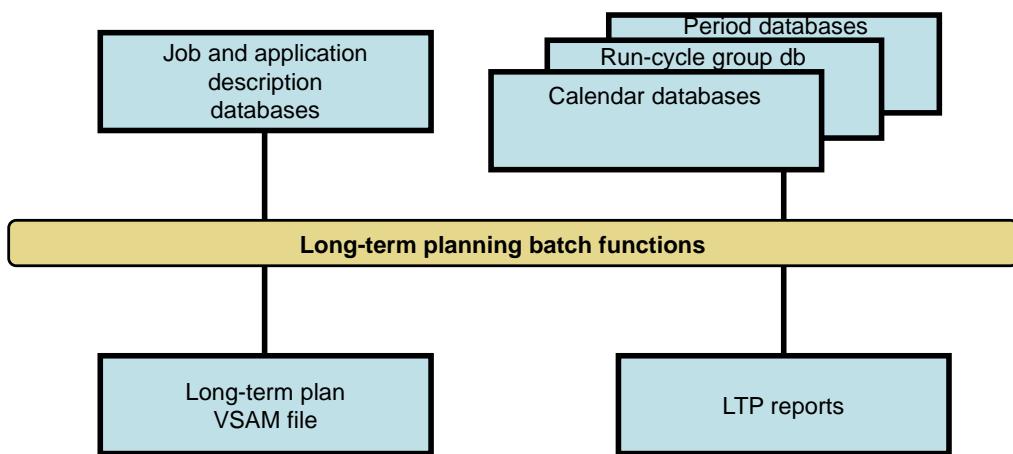
The Tivoli Workload Scheduler for z/OS ISPF panel option **6.6** shows important CP information:

- **Current plan created:** Shows the date and time when the current plan was created (might have been done by a daily planning or replan batch job).
- **Planning period end:** Shows the end date and time of the current planning period.
- **Last CP backup:** Shows the date and time when the most recent backup of the current plan was taken.
- **First logged event after backup:** Shows the date and time when the first event was written to the job-tracking-event log after the last backup of the current plan. The corresponding JTEVENT time stamp is also displayed. JT events are logged using GMT (Greenwich Mean Time).
- **Under production:** Shows whether a new current plan is under production:
 - **Yes:** Daily planning has started to build a new current plan.
 - **No:** No daily planning activity is occurring.
- **NCP ready:** Shows whether daily planning built the new current plan.
 - **Yes:** Daily planning built the NCP. The Normal Mode Manager subtask is activating the new current plan.
 - **No:** A new current plan is not ready to be activated.
- **Run number:** Shows the last Symphony run number.

- **Under production:** Shows whether a new symphony is under production:
 - **Yes:** Daily Plan processing started building a new symphony.
 - **No:** No Daily Plan activity is occurring.
- **New Symphony ready:** Shows whether Daily Plan processing built the new symphony:
 - **Yes:** Daily Plan built the symphony file. The Normal Mode Manager subtask is activating the new symphony file.
 - **No:** A new symphony plan is not ready to be activated.
- **Current plan:** Shows the DD name of the physical current plan data set currently in use.
- **Job-tracking log:** Shows the DD name of the current job-tracking log.
- **JCL repository:** Shows the DD name of the JCL repository that is in use.

Lesson 2 Long-term planning overview

Lesson 2 Long-term planning overview



© Copyright IBM Corporation 2015

In this lesson, you learn about the long-term planning process. After completing this lesson, you should be able to describe the long-term plan build process and functions.

The long-term planning process uses information from the databases to create a forecast of work that is scheduled to run. The process scans the application and job description databases for all active application and job descriptions with valid run-cycle information. It also uses the relevant run cycles and rules to calculate the run dates and generate occurrences. The calendar that is associated with the application is checked and occurrences are rescheduled according to the free-day rule.

Tivoli Workload Scheduler for z/OS saves the generated occurrences in the long-term plan file. Batch jobs that maintain the LTP are as follows:

- **LTC** (Long-Term Plan Create): Used for creating an LTP the first time Tivoli Workload Scheduler for z/OS initializes, and after an LTP refresh.
- **LTX** (Long-Term Plan Extend): Used for extending an existing LTP. If the job detects changes in the databases, it also modifies existing occurrences to reflect the changes. The Tivoli Workload Scheduler for z/OS batch schedule typically includes this process.
- **LTMOA** (Long-Term Plan Modify): Used for modifying occurrences in the LTP to reflect changes that are made in the databases. The end date of the plan does not change. LTP occurrences

that were modified manually do not change. Occurrences with input arrival times earlier than the end time of the current plan do not change.

LTP build process

Steps for long-term batch planning process

1. Determine the date range of the LTP
2. Apply the date range to calendar or calendars
3. Determine the free days for each application in turn
4. Generate occurrences
5. Reschedule using the free-day rule for occurrences that are generated on a free day
6. Bypass free days when calculating a deadline. For example, a next-day run deadline for an application scheduled on Friday would be extended to Monday if weekends are free days

© Copyright IBM Corporation 2015

LTP build process

Tivoli Workload Scheduler for z/OS examines every application and job description for run-cycle information. If an application is a member of a group, Tivoli Workload Scheduler for z/OS extracts the run-cycle and calendar information from the group definition.

When the planning process finds a valid run cycle, it checks for generated dates within the range of the long-term plan. The scheduler creates an occurrence in the long-term plan for each valid run cycle in the application or job description. The planning process checks the defined calendar to determine whether the free-day rule is applied. The scheduler detects and reports a duplicate occurrence, but does not store it in the long-term plan.

After the planning process creates the occurrences, the operations are examined for external predecessors. If an operation has one or more external predecessors, the planning process tries to resolve the dependency in the long-term plan based on the dependency resolution that is assigned to that operation. The operation input arrival time is used to look for an application occurrence within the specified time frame that contains the predecessor operation. When closest preceding is used (no range specified), then the dependency links to the closest past occurrence where the occurrence input arrival time is the same or earlier. When a range was specified, then the search is first done backward for the occurrence and then forward. If the dependency cannot be resolved, then none is created unless the predecessor was mandatory. Warning messages are issued for abnormal conditions.

Maintaining the long term plan panel

```
EQQLTOPP ----- MAINTAINING THE LONG TERM PLAN -----
Option ==> _

Select one of the following:

1 ONLINE      - Occurrence utility:
                  Browse, Create, Delete, List, and Modify
                  Occurrences and Dependencies in the long term plan.
                  Job setup for occurrences in the long term plan.

2 BATCH        - Plan utilities:
                  Modify, Create and Extend the long term plan
                  from run-cycle information.
                  Make a Trial long term plan.
                  Print long term plan.

3 ADD          - Create an occurrence in the long term plan

4 STATUS        - Display status of the long term plan

5 SET DEFAULTS - Set defaults to be used for browsing long term plan
```

ISPF option 2

© Copyright IBM Corporation 2015

Maintaining the long term plan panel

The long-term plan is the reservoir of planned work that is used by daily planning for creating the current plan. The functions that you use for producing the long-term plan are batch functions.

At the Maintaining the Long-Term Plan panel, select option **2**, and select BATCH to perform the following tasks:

- **Create** a new long-term plan.
- **Modify** the existing long-term plan.
- **Extend** the existing long-term plan.
- **Create trial** long-term plans.
- **Print** the long-term plan.

Long-term plan batch functions

The following options are in the **Selecting Long-Term Plan Batch Job** menu:

- **MODIFY**: This function modifies the contents of the LTP to reflect changes that are made in the Tivoli Workload Scheduler for z/OS databases. It does not modify occurrences with input arrival times later than the end time of the current plan. It also does not change the end date of the existing LTP. It does resolve external dependency links. The MODIFY batch function does not modify occurrences that were changed by using the online occurrence utility functions.
- **MODIFY ONE**: With one exception, this function does the same things as MODIFY. The exception is for a single named application. It does not resolve external dependency links.

- **EXTEND:** Along with all the functions of MODIFY, EXTEND changes the LTP end date and adds more days based on user specification. Users typically automate the long-term plan extension by scheduling the batch job in a Tivoli Workload Scheduler for z/OS application.
- **TRIAL:** Users create a test long-term plan without affecting the existing production long-term plan. The user specifies start and end dates and the plan is generated. The results are saved in a sequential data set chosen by the user.
- **PRINT** and **PRINT ONE:** With these print functions, you can send a list of applications to the printer.
- **CREATE:** You use this function the first time Tivoli Workload Scheduler for z/OS initializes and after an LTP Refresh only. You cannot use the CREATE function when a current plan exists.

The 24-hour day is the minimum time frame for a long-term plan. Be sure to enter start and end dates. The maximum length of the long-term plan is 4 years.

Generating JCL for batch functions

Tivoli Workload Scheduler for z/OS generates a listing whenever you run one of its batch functions. The listing shows the results of the function. For example, you receive a full listing of the long-term plan file contents when you create a long-term plan. You can send the listing to one of the following destinations:

- A system printer: SYSOUT CLASS
- A local printer: LOCAL PRINTER NAME
- A data set: DATASET NAME.

When you start a batch job function from the panel, you typically send the result to a data set. If the DATASET NAME field is blank, Tivoli Workload Scheduler for z/OS creates an output data set with the TSO user ID as the high-level qualifier. The controller subsystem name is the next qualifier. The function being run is the third qualifier, and LIST is the last qualifier.

The first time that you use the panel to start a Tivoli Workload Scheduler for z/OS batch function, type **E** in the **SUBMIT/EDIT JOB** field to edit. Then, you can review and save the JCL before you submit it.

You must also supply a job card for your JCL. You must enter it the first time only. When you terminate your session normally, ISPF saves the job card in your ISPF profile data set. EQQAPROF is the member that holds the job card and other Tivoli Workload Scheduler for z/OS user options. The batch job JCL is generated when you press Enter on this panel.

LTP batch job parameters

In the JCL that is created for you to run the long-term plan jobs, the EXEC PARM tells the EQQBATCH program that you are submitting a long-term plan create job. The SYSIN DD * provides the long-term plan date range. The first 6 digits are the starting date in YYMMDD format. The second 6 digits are the ending date in YYMMDD format.

Contents of the long-term plan

EQLSTOL ----- LONG TERM PLAN OCCURRENCES (left part) ----- Row 1 of 16											
Command ==> Scroll ==> CSR											
Enter the CREATE command above to create a new occurrence or enter the GRAPH command above to view occurrences graphically or scroll right, or, enter any of the commands below: B - Browse, D - Delete, J - Job setup, M - Modify, RG - Remove from Group											
Row cmd	Application id	Input arrival date	arrival time	Deadline date	time	P C	Pre	Suc	Cond Pre	Cond Suc	Pnd Man Pre
111	PRDAPPL1	14/03/10	07.00	14/03/10	08.00	5 N	0	0	0	0	0
111	TEST1	14/03/10	07.00	14/03/10	08.00	5 N	0	0	0	0	0
111	UHCRITA	14/03/10	07.00	14/03/10	07.15	7 N	0	0	0	0	0
111	EA73MST	14/03/11	06.00	14/03/12	06.00	5 N	0	2	0	0	0
111	EA73MST2	14/03/11	06.00	14/03/12	06.00	5 N	0	0	0	1	0
111	PRDAPPL1	14/03/11	07.00	14/03/11	08.00	5 N	1	0	1	0	0
111	TEST1	14/03/11	07.00	14/03/11	08.00	5 N	0	0	0	0	0
111	UHCRITA	14/03/11	07.00	14/03/11	07.15	7 N	0	0	0	0	0
111	PRDAPPL1	14/03/12	07.00	14/03/12	08.00	5 N	1	0	0	0	0
111	TEST1	14/03/12	07.00	14/03/12	08.00	5 N	0	0	0	0	0
111	UHCRITA	14/03/12	07.00	14/03/12	07.15	7 N	0	0	0	0	0
111	EA73MST	14/03/13	06.00	14/03/14	06.00	5 N	0	1	0	0	0
111	EA73MST2	14/03/13	06.00	14/03/14	06.00	5 N	0	0	0	1	0
111	PRDAPPL1	14/03/13	07.00	14/03/13	08.00	5 N	1	0	1	0	0
111	TEST1	14/03/13	07.00	14/03/13	08.00	5 N	0	0	0	0	0
111	UHCRITA	14/03/13	07.00	14/03/13	07.15	7 N	0	0	0	0	0

ISPF option =2.1

© Copyright IBM Corporation 2015

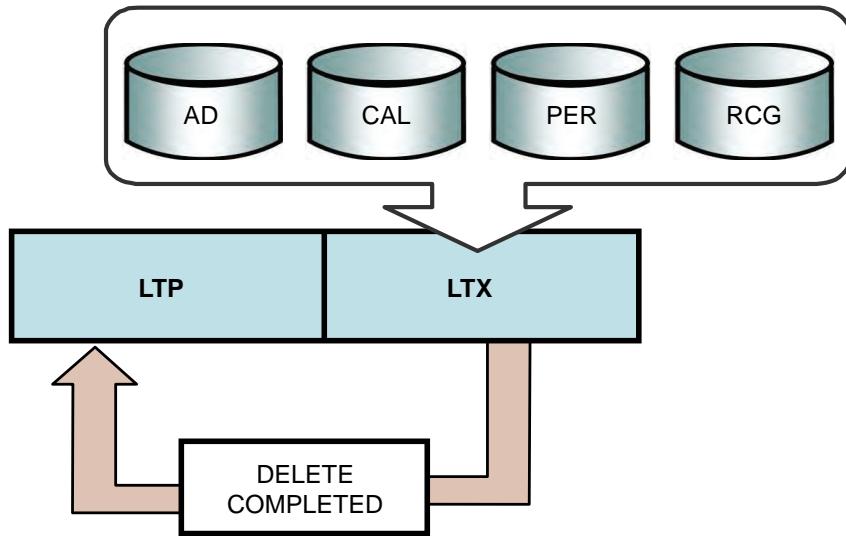
Contents of the long-term plan

You can view the contents of the LTP online from panel option **2.1**. The LTP file contains the following items:

- A list of application occurrences in chronological order by input arrival date and time.
- The deadline date and time of each occurrence. Tivoli Workload Scheduler for z/OS calculates the date from the deadline day value that is specified on the application run cycle.
- Resolved external dependencies: **Pre**decessors and **Suc**cessors.
- The status of each occurrence: **Completed**, **Yes**, or **No**.

The LTP does not store all the AD information in its own file. If you browse an occurrence in the LTP, you are looking at the AD database.

Long-term plan extension



© Copyright IBM Corporation 2015

Long-term plan extension

The LTP extension batch process is as follows:

1. Searches the AD database and uses valid run cycles to schedule application occurrences.
2. Checks the calendar database to ascertain free days, and reschedules according to the free-day rule.
3. Resolves external dependencies.
4. Deletes completed occurrences.

When you create a ***new current plan (NCP)***, part of the process is updating the long-term plan. The LTP is updated with any occurrences that were completed in the previous current plan.

When you extend or modify the long-term plan in batch, the LTP batch function deletes completed occurrences in scheduled-day lots. That is, either all or none of the occurrences for the day are removed for a particular day.

Assume that you have an occurrence in your long-term plan that is not completed. All other occurrences that are scheduled for that day and all following days are retained until the incomplete occurrence is completed. Avoid excessive growth of your long-term plan file by checking for incomplete occurrences regularly.

Normally, the long-term plan is extended by a batch job that the Tivoli Workload Scheduler for z/OS schedules. However, you can extend the LTP from the ISPF panels. Start the long-term plan extend batch function with ISPF option **2.3** to perpetuate an existing long-term plan.

The long-term plan extend function requires an EXTENSION LENGTH in days or a specific NEW END DATE. Use EXTENSION LENGTH when creating a batch job to automate the long-term plan extension. Fill a field and press Enter to see the panel for generating JCL for a batch job. From this panel, you can complete the details to submit a job that extends the long-term plan according to the requirements you entered.

LTP dependencies

Dependencies in the long-term plan

- Determine the range of the LTP
- LTP resolves dependencies by scanning backward in time until it finds the first predecessor occurrence if closest preceding was chosen when defining the operation in the application (default).
- If a range or same day was chosen for dependency resolution then the dependency is resolved by first scanning backward, and then forward within the limits of the range
- Start point for the scan is
 - The operation input arrival time, if it is a time-dependent job
OR
 - The input arrival time of the occurrence

© Copyright IBM Corporation 2015

LTP dependencies

LTP dependencies include only external dependencies. Only the long-term bplan create, extend, and modify all batch functions resolve external dependencies.

When resolving dependencies, Tivoli Workload Scheduler for z/OS uses the long-term plan that exists. Occurrences that fall within the current plan are also considered in the dependency resolution process.

After the LTP batch job generates its list of occurrences, it scans backwards and forwards (depending on the dependency resolution option) to find the most appropriate predecessor application. If an input arrival time (IA) is specified for the dependent operation, the operation IA is used. Otherwise, the scheduler uses the application occurrence IA from the run cycle.

If the AD has several external dependencies at different points, they are all resolved.

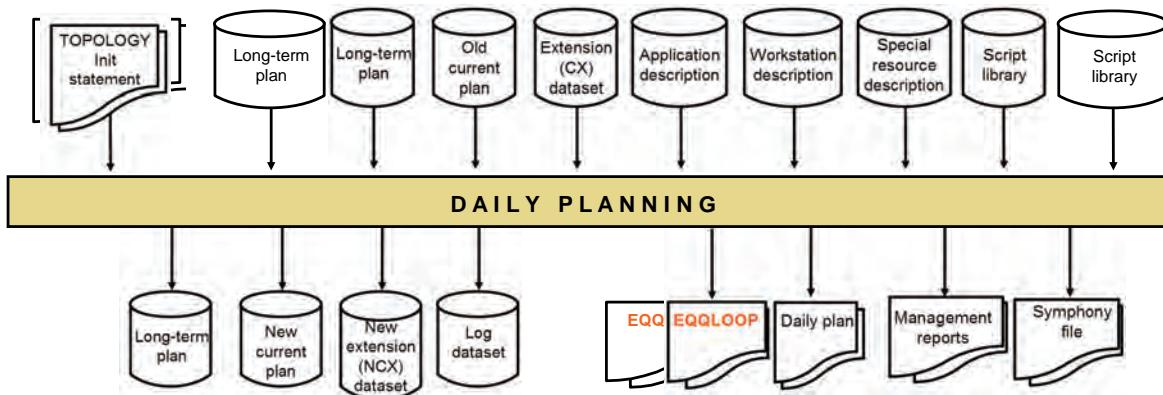
By using the ONLINE option in Maintaining the Long-Term Panel, you can browse, create, delete, list, and modify occurrences in the long-term plan. If you need to make special changes to the long-term plan, you can do it from here. From the Online panel, you can perform the following tasks:

- Add an application occurrence to the plan.
- Delete an application occurrence from the plan.
- Remove an occurrence from an occurrence group.
- List all the occurrences of an application in the plan.
- List all members of an occurrence group.
- Browse individual application occurrences in the plan.
- Browse the dependencies of application occurrences in the plan.
- Change application occurrences in the plan.
- Change the dependencies of application occurrences in the plan. You can perform these tasks:
 - Create or delete unconditional dependencies.
 - Delete conditional dependencies.
- Prepare job statements for an occurrence in the plan.
- Delete, complete, or add an occurrence group.

Lesson 3 The current plan

Lesson 3 The current plan

Daily planning: Requirements and results



© Copyright IBM Corporation 2015

In this lesson, you learn about the contents and purpose of the current plan. You also learn how to create and remove the current plan. After completing this lesson, you should be able to describe and create the current plan.

The process of producing the current plan is called **daily planning**. Daily planning requires input from the following sources:

- The long-term plan, which contains a list of application occurrences to run each day. The long-term plan details the input arrival and deadline times and external dependencies for every occurrence.
- The old current plan, if one exists. Incomplete applications carry forward into the new current plan, and completed occurrences are marked in the LTP.
- Description databases as follows:
 - The application description database, which contains the detail of applications at operation level.
 - The workstation description database, which shows the open intervals, parallel servers, and fixed resources available at each workstation.
 - The resource description database, which has details of special resources.
- Tivoli Workload Scheduler topology statements and the distributed environment script (job description) library, if the end-to-end feature is installed. The TOPOLOGY initialization statement is used for the Symphony file. The script library is used in the creation of the Symphony file.

Daily planning collects and processes data from the input data sets to update relevant data sets and produce current plan reports. The results are as follows.

- A new current plan with the following items:
 - Incomplete operations from the previous current plan
 - New occurrences
 - Potential predecessor records for each occurrence.
- Updates to the long-term plan for occurrences that are marked complete or deleted in the previous current plan
- Creation of the distributed environment Symphony file for the distributed agents, if the end-to-end feature is installed
- Daily planning reports and management reports

Daily planning batch functions

```
EQQDPLNP ----- PRODUCING TWSZ DAILY PLANS -----
Option ==> _

Select one of the following :

1 REPLAN           - Replan current planning period
2 EXTEND           - Extend the current planning period
3 TRIAL            - Produce a trial plan
4 PRINT CURRENT   - Print statistics for current planning period
5 SYMPHONY RENEW   - Create Symphony file starting from Current Plan
6 ARCHIVE          - Archive old current plans
```

ISPF option =3

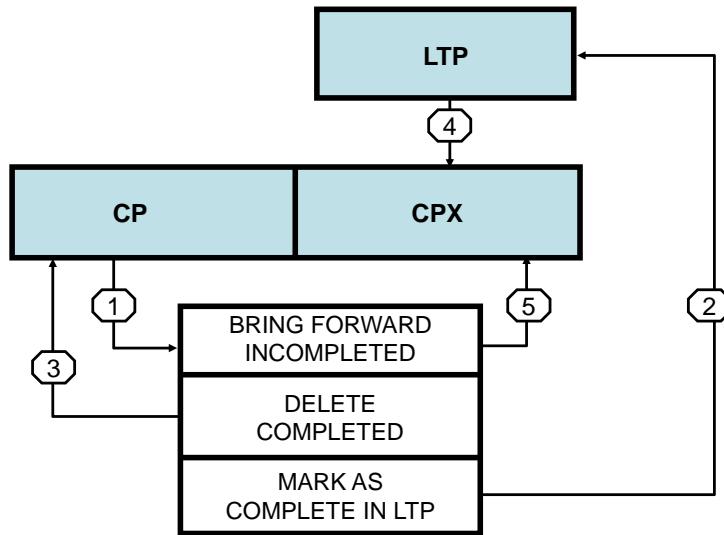
© Copyright IBM Corporation 2015

Daily planning batch functions

From the **DAILY PLANNING** option, you can perform functions as follows, which pertain to the current plan:

- **REPLAN:** Remove completed occurrences from the current plan. It updates the long-term plan with the occurrence data. It also recalculates the planned start times of the occurrences remaining in the current plan. It does not change the current plan end time or add new occurrences from the long-term plan.
- **EXTEND:** Create a new current plan if none exists. Extends the existing plan from its end time for a user-specified length of time, or to a specified time. This function performs the following tasks:
 - Updates the LTP with completed occurrences
 - Copies occurrences from the LTP whose input arrival times fall between the end of the existing current plan (CP) and the end time for the New current plan.
 - Carries forward all Incomplete occurrences from the existing current plan
- **TRIAL:** Create a trial current plan within the range of the long-term plan without affecting the existing in-production current plan.
- **PRINT CURRENT:** Print statistics for the current planning period.
- **SYMPHONY RENEW:** (Relevant for planned end-to-end with FTA workstations). Use for recovering from error situations when you add or change information in the current plan. You can also use this function after current plan recovery actions.

The daily planning process



© Copyright IBM Corporation 2015

The daily planning process

The sequence of events is as follows:

1. When the current plan is extended, the daily planning process brings forward the occurrences from the old current plan.
 2. Daily planning marks these brought-forward occurrences as locked in LTP until it has checked the old completed ones.
 3. They are deleted from the CP and marked as complete in the LTP.
 4. Daily planning examines the LTP for all occurrences that are scheduled to start during the life of the new CP and brings them into the CP.
 5. The new current plan now includes only new occurrences and the old, incomplete, occurrences.
- If the end-to-end feature is active, the daily planning process continues with these actions:
6. In the new current plan, the daily planning job flags the jobs and job streams that are added to the Symphony file.
 7. The job and job stream information is added to the new Symphony file.
 8. Tivoli Workload Scheduler for z/OS sends the Symphony file to the Tivoli Workload Scheduler distributed domain managers and directly connected agents.
 9. Tivoli Workload Scheduler applies all outstanding events of the old Symphony file to the new Symphony file.
 10. In Tivoli Workload Scheduler, the new Symphony file is distributed to other fault-tolerant agents.

Current plan batch job parameters

```

EQQDPEXP ----- EXTENDING CURRENT PLAN PERIOD -----
Command ==>

Enter/change data below and press ENTER

Current plan end date : 14/03/11 06.00

START DATE      ==> _____ YY/MM/DD If no current plan exists
TIME            ==> _____ HH.MM
END DATE        ==> _____ YY/MM/DD Specific END date
TIME            ==> _____ HH.MM Specific END time
EXTENSION LENGTH ==> 02400 HHHMM Extend plan by
TYPE            ==> A   A - includes all days
                           W - includes only work days

Report selection :
WS SUMMARY      ==> Y   Y if report wanted, otherwise N
OPERATING PLAN   ==> Y   Summary for all work stations
WS PLANS         ==> Y   Daily operation plan
INPUT ARRIVAL    ==> Y   Plans for all work stations
NON REPORTING   ==> Y   List of input arrival operations
CURRENT PERIOD   ==> Y   Plans for non reporting work station
PLANNED RESOURCE ==> Y   Print current period results
ACTUAL RESOURCE   ==> Y   Planned resource utilization
                           Actual resource utilization

```

© Copyright IBM Corporation 2015

Current plan batch job parameters

The current plan is normally extended 24 hours at a time, by a batch job that you schedule Tivoli Workload Scheduler for z/OS to run. You can create a new current plan or manually extend it from the **EXTEND** option on the **DAILY PLANNING** menu (option **3.2**).

On the Extending Current Plan Period panel, the **Current plan end date** field is blank when no current plan exists. If no current plan exists, Tivoli Workload Scheduler for z/OS uses the current date and time for the start of the current plan.

When creating or extending the CP, you can specify either a specific end date and time or an extension length in hours and minutes. You can tell Tivoli Workload Scheduler for z/OS to extend the CP to end on workdays only or on any day. Type **W** in the **TYPE** field to specify including workdays only. Type **A** in the **TYPE** field to specify that all days are acceptable end days.

When creating, extending, or replanning the current plan, you can produce reports on the results. Daily planning provides two printed reports:

- Plans
- Management reports

You can include all or some of the following items in a plan report:

- Histograms, that show the planned use of all workstations in the current plan.
- A list of all occurrences and their operations to be processed, in alphabetic order.
- Planned utilization of special resources.
- Workstation plans, which list all operations to be performed at each workstation, in order of planned start time.
- A list of all operations to be performed at each workstation, in order of input arrival time.

You can create management reports for the current period and for the previous period. The information that you can select to include in a management report includes the following items:

- Current period results, which show the following information about the old current plan:
 - Completed applications.
 - Operations that ended in error.
- Previous period results, which show information about the 24-hour period preceding the old current plan:
 - Summary of completed applications.
 - Completed applications.
 - Operations that ended in error.
 - Special resource actual utilization.
 - Error statistics on completed applications.
 - Workstation histograms: actual utilization.
 - Missed feedback report.

Creating and extending the current plan

- Service function Refresh – Destructive action – rarely used
 - ISPF option =9.5
 - Deletes existing current plan and resets the long term plan
- Use the Extend option =3.2 to create a current plan after a refresh
 - With Extend, you create a new plan or extend an existing plan
- To extend the current plan on a day-to-day basis, schedule a job to run
 - You can schedule the LTP and CP batch jobs together in an application to extend them both each day

```
EQQUTOPP ----- SERVICE FUNCTIONS -----
Option ==> _

Select one of the following:
Job submission: Current Status: Active
1 DEACTIVATE      - Deactivate job submission
2 ACTIVATE        - Activate job submission

Automatic recovery: Current Status: Active
3 DEACTIVATE      - Deactivate automatic recovery
4 ACTIVATE        - Activate automatic recovery

5 REFRESH          - Delete current plan and reset long term plan
```

© Copyright IBM Corporation 2015

Creating and extending the current plan

When you complete the Extending Current Plan Period panel, press Enter to view a panel for generating JCL to submit. In the JCL created, the EXEC statement PARM tells the EQQBATCH program that you are submitting a daily plan extension. The SYSIN DD * parameters provide the start date, start time, and duration in hours. The **W** specifies to ignore free days.

Deleting the current plan

To plan again from scratch, delete the current plan, and reset the long-term plan by using Tivoli Workload Scheduler for z/OS ISPF option **9.5**.



Note: This is a highly disruptive action. Deleting the current plan prevents Tivoli Workload Scheduler for z/OS from running any work. Nothing is scheduled or run without a current plan. The ability to issue this command is normally only available to a superuser id that needs to be activated before it can be used.

Student exercises



See the Student Exercise Guide Unit 6

© Copyright IBM Corporation 2015

Student exercises

Perform the exercises for this unit.

Summary

Now that you have completed this unit, you can perform the following tasks:

- Describe the functions of plans
- Describe the relationship between the databases and the plans
- Describe the relationship between the long-term plan and the current plan

© Copyright IBM Corporation 2015

Summary

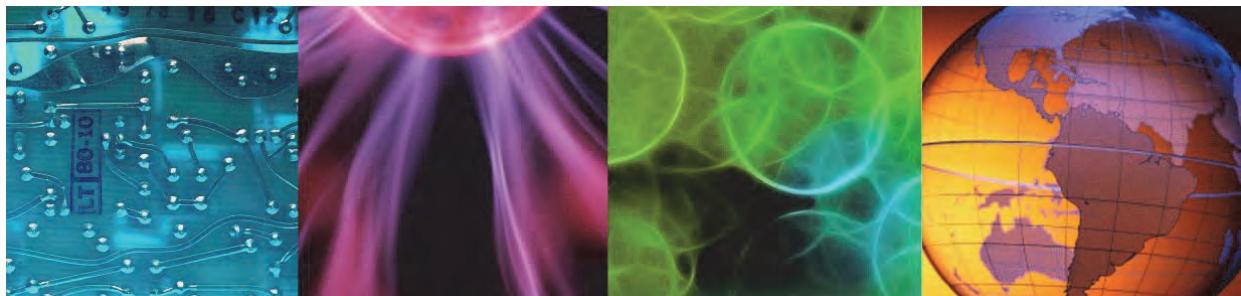


7 Restart and cleanup

IBM Tivoli Workload Scheduler for z/OS 9.2.0



7 Restart and cleanup



© Copyright IBM Corporation 2015

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

In this unit, you learn how to use the IBM Tivoli Workload Scheduler for z/OS restart and cleanup functions.

References:

- SC32-1262 *IBM Tivoli Workload Scheduler for z/OS Managing the Workload*
- SC32-1265 *IBM Tivoli Workload Scheduler for z/OS Customization and Tuning*

Objectives

- List the tasks that the restart and cleanup functions can perform
- Specify appropriate restart and cleanup options for selected operations
- Use the Restart and Cleanup panel for these tasks:
 - Perform job or step-level operation restarts
 - Override data set cleanup actions

© Copyright IBM Corporation 2015

Objectives

Not every job you schedule will work. This unit introduces you to the restart and cleanup feature which can update the system catalogs to allow jobs to be restarted. After this unit and its exercises you will have a good understanding of how this feature works and how it can be utilized in your environment.

Lesson 1 Introduction to restart and cleanup

Lesson 1 Introduction to restart and cleanup

- Included in Tivoli Workload Scheduler for z/OS
 - Restarts operations at the job level or step level
 - Cleans data sets for restarted operations
 - Applies to only z/OS jobs
- Requires the data store started task
 - Interfaces with JES
 - Builds its own copy of the job logs
- You can browse the job log of jobs from any Tivoli Workload Scheduler for z/OS user interface

© Copyright IBM Corporation 2015

In this lesson, you learn the basic concepts of the restart and cleanup feature of Tivoli Workload Scheduler for z/OS. After completing this lesson, you should be able to describe the basic concepts of the restart and cleanup feature in Tivoli Workload Scheduler for z/OS.

Restart and cleanup is a Tivoli Workload Scheduler for z/OS function that performs two tasks:

- Restarting an operation at the job level or the step level
- Cleaning up the associated data sets

You can run restart and cleanup for jobs in the z/OS environment only.

Restart and cleanup overview

- Restart an entire job, or just one or more steps
- Automatically identify the best restart step
- Modify the JCL before restarting
- Clean data sets before restarting
 - Catalog data sets
 - Uncatalog data sets
 - Delete data sets
 - Resolve generation data group (GDG) names
- Interface with Removable Media Manager (RMM)

© Copyright IBM Corporation 2015

Restart and cleanup overview

Use the restart and cleanup function for step-level restarts of z/OS jobs or started tasks even with no catalog modifications. Also, use restart and cleanup to catalog, uncatalog, or delete data sets for job restarts. You can also browse JES job logs and job logs from other operating platforms that do not have SDSF (System Search and Display Facility).

Restart and cleanup actions can be initiated automatically. You can also manually initiate them from the Tivoli Workload Scheduler for z/OS panels. If catalog modifications are required, the scheduler resets the catalog to its run state before the job. This function applies to both generation data group (GDG data sets) and data sets allocated within the JCL.

The restart and cleanup function manages resolution of GDG names, JCL containing nested INCLUDEs or PROCs, and IF-THEN-ELSE statements. Tivoli Workload Scheduler for z/OS interfaces with RMM, which assists with the management of data sets.

For some restarts, JCL modifications might be necessary. The restart and cleanup function can handle these modifications and even identify the best restart step in a job or procedure.

Job log retrieval data sets

- Data store VSAM data types
- EQQUDFxx are linear data sets
 - They save job logs for job log retrieval
 - They are unstructured SYSOUT
 - Their job log retrieval option must be set (EQQJOBS utility)
- EQQSDFxx are linear data sets that are used for saving structured job log information
- EQQPKIx and EQQSKIx are KSDS data sets used as the primary and secondary indexes
- Other data sets that the controller maintains
- EQQSDFxx
- EQQPKIx
- EQQSKIx

© Copyright IBM Corporation 2015

Joblog retrieval data sets

The data store facility collects structured and unstructured information for all submitted jobs for restart and cleanup functions.

- **Structured** data is the job log information that is associated with job steps and data sets.
- **Unstructured** data is SYSOUT data. Collecting unstructured data is optional.

Starting restart and cleanup

- Initialize restart and cleanup in the controller
 - Include restart and cleanup data sets in the controller started task
 - OPCOPTS RCLEANUP(YES)
- Configure and start the data store started task
 - Set the DSUTIL initialization statement to configure the cleanup utility to run as a subtask to do job log dataset maintenance
 - Set up the EQQCLEAN JCL procedure in the JES proclib concatenation
 - The EQQUXCAT user exit routine allows further customization
- Specify restart and cleanup options for operations
 - Use the ISPF panels
 - Available in the Tivoli Dynamic Workload Console

© Copyright IBM Corporation 2015

Starting restart and cleanup

The data store component has utilities that can run in batch mode only when the data store is not running. An exception is made only for the cleanup utility, which can run also as a subtask of the data store. In this case, called **online mode**, the utility runs periodically and deletes selected records from the database.

The data store EQQPARM includes the DSOPTS initialization statement, which uses the CINTERVAL and CLNPARM parameters to control the utility subtask, as in this example:

```
DSTOPTS CINTERVAL(60)  
CLNPARM(DSCLEAN)
```

CINTERVAL(interval time|0) defines the cleanup task wake-up interval time, expressed in minutes. Valid values are 0 - 1440. The default is 0, which means that the data store CleanUp subtask starts at data store initialization, but never runs.

CLNPARM(member name|EQQCLNPA) defines the parameter member name that contains the Cleanup task criteria. The default is EQQCLNPA. This parameter is meaningful only if CINTERVAL is greater than 0. The member that is specified contains the DSUTIL initialization statement.

Here is an example of a **DSUTIL** statement:

```
DSTUTIL DELSTRUC  
SEARCH1(OLDRDD1)  
DELUNSTR  
SEARCH2(OLDRDD1)
```

DELSTRUC|DELUNSTR|DELBOTH defines the criteria for deleting SYSOUT from the data store database. You can specify whether structured data (**DELSTRUC**) or unstructured data (**DELUNSTR**) is deleted. Specify **DELBOTH** to delete both types.

The **SEARCHnn** keyword (where *nn* can assume a value from 1 to 10) identifies the criteria that are selected. The string value has the following format:

CodeNameComparisonOperFieldValue

See the *IBM Tivoli Workload Scheduler for z/OS Customization and Tuning Guide* for details.

EQQCLEAN is a data store sample JCL procedure the scheduler provides and places in the JES proclib concatenation. EQQCLEAN is inserted into jobs that are defined to use the restart and cleanup function, as in the following example:

```
//*****  
//EQQCLEAN PROC  
//EQQCLEAN EXEC PGM=EQQCLEAN, REGION=0M,TIME=1440,PARM='&EQQPASS'  
//STEPLIB DD DISP=SHR,DSN=  
//EQQCLNDD DD DUMMY  
//EQQSIMDD DD DUMMY  
//EQQGDGDD DD DUMMY  
//EQQROOTD DD DUMMY  
//EQQDUMP DD SYSOUT=*  
//SYSPRINT DD SYSOUT=*  
//SYSUDUMP DD SYSOUT=*  
//SYSOUT DD SYSOUT=*  
// PEND
```

Use Tivoli Workload Scheduler for z/OS ISPF panel to configure restart and cleanup to run. Specify Restart and Cleanup options for individual options.

The **EQQUXCAT** user exit routine is available for EQQCLEAN to call for discarding specific cleanup actions at job run time. The sample EQQDELDS or EQQCLEAN program calls EQQUXCAT before the start of any simple cleanup action.

In some cases, EQQCLEAN might delete a data set you did not want deleted. Protect critical data sets from deletion by using either the RCLOPTS parameters (DDPROT, DDPRMEM, DSNSPROT, DSNSPRMEM) or the EQQUXCAT exit routine. EQQDELDS/EQQCLEAN does not run the action when the return code is set to a value other than 0. For details, see the *IBM Tivoli Workload Scheduler for z/OS Customization and Tuning Guide*.

Lesson 2 Restart and cleanup options

Lesson 2 Restart and cleanup options

- The operation must be a z/OS JCL job running on a computer workstation
- Cleanup options
 - Delete data sets created in a job
 - Uncatalog data sets cataloged in a job
 - Catalog data sets uncataloged in a job
- Datasets can be protected against cleanup actions
- Restart and cleanup can occur together or separately
- Operators can override cleanup actions
- Cleanup and automatic recovery can work together

© Copyright IBM Corporation 2015

In this lesson, you learn how to define restart and cleanup options. After completing this lesson, you should be able to define restart and cleanup options.

Restart and cleanup options are specified for computer workstation-based operations. The operations are z/OS JCL jobs. When you perform a cleanup, you take previously allocated data sets and catalog, uncatalog, or delete them.

You can run cleanup actions together with restarts or run cleanup separately. For example, you can specify the step restart and cleanup of an operation, which are completed at job run time. In other situations, you can run the data set cleanup separately, before the job reruns. In the second case, the step restart begins after the cleanup completes.

Two types of cleanup are as follows:

- Cleanup that is based on the history of a failed job run.
- Preventive cleanup that you can perform before a job runs. This type of cleanup runs a general cleanup of data sets, which is useful if you do not know the exact state of the data sets. This type of cleanup is not described in this unit. If you want to clean up data sets before a job first runs, look at the EQQDELDI member in the supplied sample library. The EQQDELDI member describes a program that deletes data sets with the NEW disposition that are not referred to with a different disposition in previous job steps. A different disposition can be OLD or SHR.

EQQDELDS is an optional part of the product, which can be used to avoid not-cataloged situations.

Cleanup and automatic recovery cooperate to ensure that cleanup actions are completed or revoked before any automatic recovery action starts for an operation. Automatic recovery is presented in detail in [Unit 10, “Automatic recovery”](#) on page 291.

The RC row command

```

EQQMOPRL --- MODIFYING OPERATIONS IN THE CURRENT PLAN (left Row 16 to 21 of 21
Command ===> PAGE

Enter the GRAPH command above to view list graphically,
enter the HIST command to select operation history list, or
enter any of the following row commands:
J - Edit JCL           M - Modify       B - Browse details
DEL - Delete Occurrence MH - Man. HOLD   MR - Man. RELEASE oper
O - Browse operator instructions NP - NOP oper   UN - UN-NOP oper
EX - EXECUTE operation D - Delete Oper RG - Remove from group
L - Browse joblog      K - Kill         RI - Recovery Info
RC - Restart and CleanUp FJR - Fast path JR
TCJ - Target Critical Job FSR - Fast path SR
BND - Reset bind information FSC - Fast path SC

Row Application id Operat Jobname Input Arrival Dep Cond Dep S
cmd ws no.          Date Time  Suc Pre Suc Pre
*** U#APPL1A U#CP 025 SM40#J5 11/10/22 06.00 0 2 0 0 W
*** U#CRITA U#CP 005 U#CRITJ1 11/10/22 07.00 2 0 0 0 C
rc' U#CRITA U#CP 010 U#CRITJ2 11/10/22 07.00 0 1 1 0 E
' U#CRITA U#CP 015 U#CRITJ3 11/10/22 07.00 1 1 1 1 W

```

© Copyright IBM Corporation 2015

The RC row command

From ISPF option =5.3, you can use the RC row command to select an operation for restart and cleanup. You can also edit the JCL or browse the job log from this panel. When you use the RC command, you are presented with the Operation Restart and Cleanup panel, as shown on the next slide.

Operation restart and cleanup panel

```
EQQRCLSE ----- OPERATION RESTART AND CLEANUP -----
Option ==>
Application : ADD1TOPLAN      15/08/21 05.49
Operation   : TWCL 10
Jobname and jobid : CPEXTND1      JOB00694
Expanded JCL used : N

Clean Up Result :
Restart and Cleanup pending: Collecting Joblog data

Edit JCL    ==> Y           Edit JCL before Restart (SR/JR only)

Select one of the following:
1 STEP RESTART          - Request a Step Restart
2 JOB RESTART            - Request a Job Restart
3 START CLEANUP          - Request Cleanup
4 START CLEANUP WITH AR - Request Cleanup with AR task
5 DISPLAY CLEANUP        - Display Cleanup result
```

© Copyright IBM Corporation 2015

Operation restart and cleanup panel

This menu panel also provides the restart and cleanup values in effect for the job and status information:

- **Expanded JCL used:** Shows the value of the **Expanded JCL** field as it is stored in the current plan.
- **Clean Up Result:** Shows the cleanup result related to the last run:
 - (blank): No clean up done
 - Completed: Clean up completed. You can have this status also when no cleanup actions were performed but the Clean Up request was confirmed.
 - Ended in error: Clean up ended in error
- **Restart and Cleanup pending:** Displayed only while the process is not yet completed after the request. It shows a status that is dynamically updated at each step of the process. The status can be:
 - Collecting job log data
 - JCL tailoring in progress
 - Process started
 - Ended in error

Input field description

- **Edit JCL:** Specify whether you want to edit the JCL before restarting the operation.
 - Y: Edit the JCL before restart.
 - N: Do not edit the JCL before restart

Menu items

1. **STEP RESTART:** Restart the operation, allow the selection and exclusion of steps to be included in the new run.
2. **JOB RESTART:** Restart the operation from beginning.
3. **START CLEANUP:** Only start the cleanup of the specified operation. Operation is not restarted.
4. **START CLEANUP WITH AR:** Only start the cleanup of the specified operation according to the AR restart step when available. Operation is not restarted.
5. **DISPLAY CLEANUP:** Display the result of the cleanup actions.

Restart and cleanup operation details panel

```
EQQAMRCL ----- RESTART AND CLEANUP OPERATION DETAILS -----
Command ==> _
Enter/Change data below:
Application : U#CRITA           TEAM # Critical App
Operation   : U#CP 010
Job name    : U#CRITJ2

Clean Up Type ==> N   Clean up Type (A/I/M/N)
Expanded JCL  ==> N   Use expanded JCL for Restart (Y/N)
User Sysout   ==> N   Log user sysout too (Y/N)
```

From ISPF option =1.4.3, select an operation and from the operation details Panel, select option 9 - cleanup options

© Copyright IBM Corporation 2015

Restart and cleanup operations detail panel

You can only specify that cleanup action is taken on computer workstations that run on z/OS. Also, you can specify whether Tivoli Workload Scheduler for z/OS uses the JCL extracted from the JES JCL SYSOUT. If necessary, you can specify whether user SYSOUT support is needed.

After the restart and cleanup function is activated on your system, you must apply it to operations to use it. You must specify options on the Restart and Cleanup Operations Details panel. The restart and cleanup options are as follows:

- Cleanup type:
 - **A** (Automatic). When the operation is ready to be submitted and the controller selects it, the controller automatically finds the cleanup actions to be taken. The controller also inserts these actions as the first step in the JCL of the restarted job.
 - **I** (Immediate). Data set cleanup is immediately performed if the operation ends in error.
 - **M** (Manual). Data set cleanup actions are deferred for the operation. They are performed when initiated manually from the panel.
 - **N** (None). No data set cleanup actions are performed.
- The JCL to use for restarts:
 - **Y**: Use the fully expanded JCL.
 - **N**: Use the JCL contained in the libraries of Tivoli Workload Scheduler for z/OS.
- Storing of user SYSOUT:
 - **Y**: Data store logs user SYSOUT.
 - **N**: Data store does not log user SYSOUT.

Automatic

Cleanup actions start either by a predefined trigger or from the panels. The controller automatically determines the cleanup actions to be taken and also inserts them as the first step in the JCL of the restarted job. With this option, the cleanup automatically reruns an occurrence when the job is ready and no other conditions keep the job from rerunning.

Immediate

The job is cleaned up when it fails. Assume that a tracked job ends in error and that immediate is specified for the operation. The controller checks for data set information in the current plan according to the following criteria:

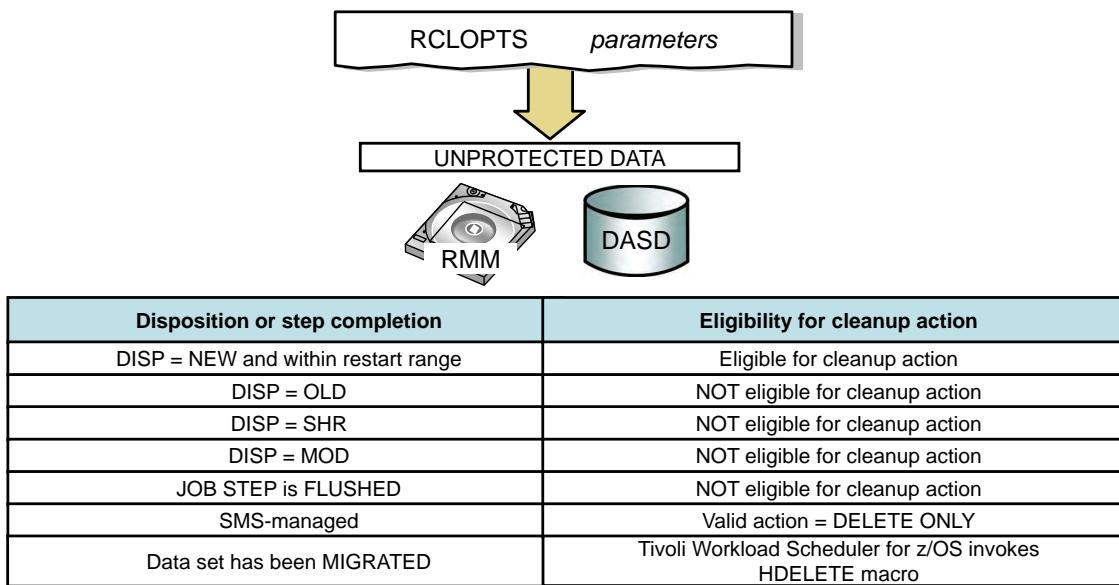
- If the automatic recovery action is to restart the operation, the controller checks for data set information. The controller starts with the step indicated by the automatic recovery function.
- If the automatic recovery action is not to restart the operation, one of two other scenarios is considered:
 - If **RCLOPTS IMMEDLOGIC(FIRSTSTEP)** was specified, which is the default, the controller checks for data set information. It starts with the first step and goes to the last step of the job.
 - If **RCLOPTS IMMEDLOGIC(BESTSTEP)** was specified, the controller starts with the best step suggested.

If data set information is found, the controller submits a stand-alone cleanup job. Otherwise, it sets the cleanup status to complete.

Manual

Cleanup is controlled manually if a tracked job ends in error and manual cleanup actions are defined for the operation. It is also controlled manually when a rerun is requested for an operation that specifies Manual. In each of these situations, you must initiate the cleanup action from the Modifying the Current Plan panel. The panel shows what action the scheduler takes for each of the affected data sets.

Data set selection eligibility



© Copyright IBM Corporation 2015

Data set selection eligibility

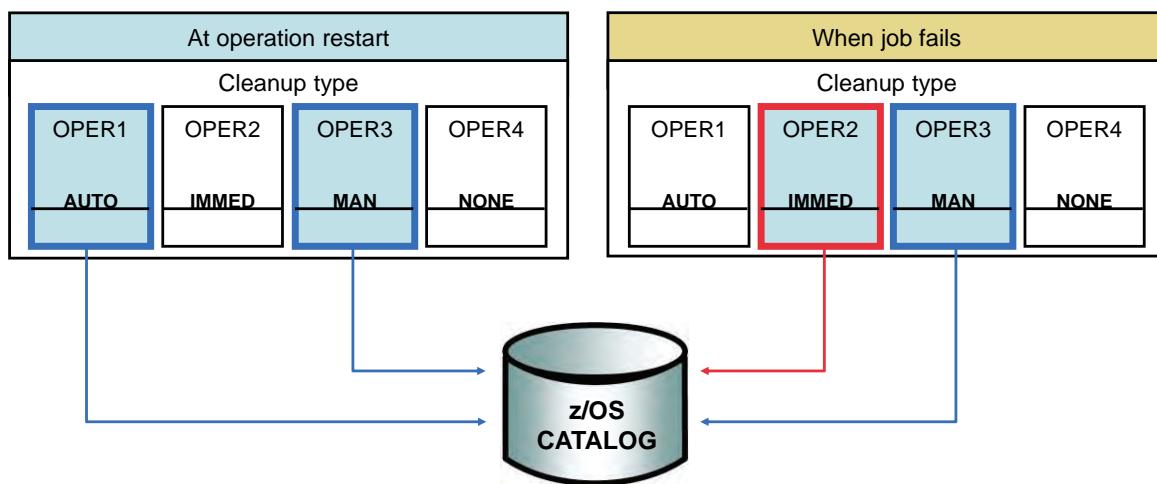
When implementing the scheduler restart and cleanup function, you can protect data sets from scheduler deletion. To prevent deletion, use one or more of these keyword parameters in the RCLOPTS initialization statement of the controller:

- **DDPROT(DD List)** defines a list of DD names that identify protected data sets.
- **DSNPROT(DSN List)** defines a list of the protected data set names.
- **DDPRMEM(member name)** specifies the name of a member of the scheduler parameter library that contains a list of protected DD names. DDPRMEM is mutually exclusive with DDPROT. You can refresh the list dynamically by issuing the z/OS **modify** command.
- **DSNPRMEM (member name)** specifies the name of a member of the scheduler parameter library. This library contains the list of protected data set names. DSNPRMEM is mutually exclusive with DSNPROT. You can refresh the list dynamically by issuing the z/OS **modify** command.

All other data sets, including tape volumes that are defined to Removable Media Manager (RMM), are eligible for cleanup. The table in this slide shows the data sets that are then eligible for cleanup actions.

To activate cleanup of tape volumes that are defined to RMM, specify **RMM(YES)** in the RCLOPTS initialization statement of the controller.

When cleanup actions are initiated



© Copyright IBM Corporation 2015

When cleanup actions are initiated

Initiation for cleanup actions occur at the following times:

- **AUTO** (Automatic): Tivoli Workload Scheduler for z/OS automatically initiates cleanup actions at operation restart only.
- **IMMED** (Immediate): Tivoli Workload Scheduler for z/OS automatically initiates cleanup actions when the job ends in error.
- **MAN** (Manual): A panel user initiates cleanup actions either at operation restart or when the job fails.

To view the result of a cleanup action, select option **4 (DISPLAY CLEANUP)** on the Operation Restart and Cleanup panel. The View Cleanup Results panel opens and displays the following status information:

- The data set cleanup that is completed or ended in error.
- The job name and job ID that performed the cleanup actions.
- The scheduler removed the data set from the catalog, uncataloged the data set, or cataloged the data set.

When the **AUTOMATIC CHECK** panel option is set to **YES**, the cleanup actions are shown for confirmation, whenever the operation is started from the panels.

Requesting notification of cleanup actions

Option 0.7 (CLEANUP CHECK)

```
EQQXCHKP ----- SETTING CHECK FOR AUTOMATIC CLEANUP TYPE -----
Command ===> _

CLEANUP CHECK    ===> Y      Specify Y to check the list of the cleanup
                           actions returned by OPC when you request a
                           RESTART function with cleanup type automatic.

                           Specify N to bypass the check.

                           Default is N .
```

© Copyright IBM Corporation 2015

Requesting notification of cleanup

Select the Tivoli Workload Scheduler for z/OS primary panel option **0.7** to request notification of cleanup actions and make one of the following choices:

- Specify **Y** if you want to check and possibly modify the Cleanup data set list that is displayed on the Modifying Cleanup Actions panels. You might want to take action even if the **Automatic** option is specified at operation level.
- Specify **N** to bypass the check. In this case, the Confirm Restart panel is directly displayed when you request a RESTART function with cleanup type automatic. The default is **N**.

Exceptions to immediate cleanup actions

At or before submission	After submission
OSEQ	MCP
OSUB	OSSQ
OSUF	OSSS
OSUP	OFSQ
OJCV	OFSS
JCLI	OFSC

© Copyright IBM Corporation 2015

Exceptions to immediate cleanup actions

If an operation ends with one of these errors, the scheduler does not perform any immediate cleanup actions. The error codes in the After submission column indicate that workload restart failed, or that the operation was manually set to error. The scheduler automatically changes the cleanup action from immediate to manual.

Lesson 3 Restarting the operation

Lesson 3 Restarting the operation

- Restarting the operation using job restart or step restart
- Setting expanded JCL option to N*
- Editing the JCL before restart
- Overriding the cleanup actions if needed
- Running cleanup, with or without automatic recovery
- Reviewing the cleanup results

© Copyright IBM Corporation 2015

In this lesson, you learn how to use restart and cleanup to restart an operation that ends in error. After completing this lesson, you should be able to use restart and cleanup to restart an operation that ends in error.

When an operation restarts, the restart point might vary, depending on the reason for the restart. Restarts can occur for the entire job or for specific steps within the job.

The JCL used for restarting the job can be the expanded JCL that was used in the original job submission. An operator can edit the JCL before the job is restarted.



Note: Use expanded JCL with extreme caution. Consider this scenario: The job originally ran on a computer workstation, not on the controller. The tracker that ran the job is on a different partition than the controller. JCL-containing nested procedures are expanded on the tracker, but the cleanup functions on the controller. The controller expands the JCL differently than the tracker, causing the restart failures when expanded JCL is used.

No resolution of GDG relative numbers occurs when the original unexpanded JCL is used. If expanded JCL is used, GDG resolution is always applied.

Selecting step restart

The screenshot shows the ISPF Operation Restart and Cleanup panel. The title bar reads "EQQRCLSE ----- OPERATION RESTART AND CLEANUP -----". The message area says "Option ==> 1_". Below it, the system displays information about the operation: Application U#CRITA, Operation U#CP 10, Jobname and jobid U#CRITJ2, Expanded JCL used N, Clean Up Result Completed. A note on the right says "Use ISPF option 5.4.0 and then the RC row command". The next section, "Edit JCL ==> Y Edit JCL before Restart (SR/JR only)", is followed by "Select one of the following:" with five options: 1 STEP RESTART, 2 JOB RESTART, 3 START CLEANUP, 4 START CLEANUP WITH AR, and 5 DISPLAY CLEANUP. Option 1 is highlighted with a red arrow pointing down to the "OPERATION RESTART AND CLEANUP" section. This section is circled in red, and another red arrow points down to the message "JOBLOG INFO REQUESTED". The final message at the bottom is "10/26 16.01.01 EQQM641I OPERINFO FOR U#CRITJ2 (JOB00751) RETRIEVED. CN(INTERNAL)".

© Copyright IBM Corporation 2015

Selecting step restart

You can issue the RC row command for requesting cleanup options only for an operation with the cleanup type set to Automatic, Immediate, or Manual.

This slide shows an example of a step restart requested for an operation with a cleanup type of automatic. The job log is requested first and a message is displayed when the job log is retrieved.

If the cleanup type is Immediate and the operation ends in error, the cleanup action occurs immediately. You can select option 5 to show the cleanup results.

You can rerun a completed operation or restart an ended-in-error operation by requesting restart and cleanup by issuing the RC row command. After you issue the RC command from ISPF option **5.4.0**, the Operation Restart and Cleanup panel opens as shown in the slide. You can specify that you want to edit the JCL for the job before a restart and if you want to use the expanded JCL.

From the panel that is shown in this slide, you can perform the cleanup independently of the restart. You can also view the results or you can initiate the cleanup by selecting a job or step restart.

To perform cleanup actions only (no restart), select the **Start Cleanup** option on the Operation Restart and Cleanup panel EQQCLSE. All the data sets, starting from the first step, are considered for the cleanup actions. The type of cleanup action depends on the following criteria:

- The status of the data set when cleanup is requested
- The DISP parameter that is specified in the JCL

Standalone cleanup without restarting

You can run cleanup without restarting the job. Both options **3** and **4** provide this capability. Option **4** works with automatic restart. Automatic restart is described in detail in Unit 10 Automatic recovery, starting on [page 297](#).

In both situations, you cannot see or modify the EQQCLEAN pre-step when editing the JCL. The submitted JCL is stored in the JS VSAM data set without the pre-step. When you insert the EQQCLEAN pre-step in a JCL, consider the following points:

- You must insert it after the JOB statement.
- You must insert it immediately before the first EXEC statement.
- If there are INCLUDE statements before the first EXEC statement, insert the pre-step before the INCLUDE statements. However, if the INCLUDE statements are listed in the RCLOPTS SKIPINCLUDE keyword, this rule does not apply. In this case, insert the pre-step immediately before the first EXEC statement.

Step restart selection list panel

```

EQQMERSL ----- STEP RESTART SELECTION LIST ----- Row 1 to 4 of 4
Command ==> go_
Primary commands: GO - to confirm the selection, END - to save it,
                   CANCEL - to exit without saving,
                   STEP - to show JCL Step Info (Expanded JCL only)
User Selection:   S - Start restart step      E - Last restart step
                  X - Step excluded (simulated flushed)
                  F - Step excluded (simulated with specified RC)
                  I - Step included

Application       : U#CRITA           11/10/22 07.00
Operation        : U#CP 10
Jobname and jobid : U#CRITJ2        JOB00751

Best Restart Step :      S1      0002
Current selected Step :     S1      0002



| User | Action | Restart | Step | StepName | ProcStep | PgmName  | Step Type | Step Status | Completion Code |
|------|--------|---------|------|----------|----------|----------|-----------|-------------|-----------------|
| Sel  | Sel    | No      |      |          |          |          |           |             |                 |
| X    | X      | N       | 0001 | EQQCLEAN | EQQCLEAN | EQQCLEAN | Clean up  | Executed    | 0000            |
| S    | S      | B       | 0002 |          | S1       | IEFBR14  | Normal    | Executed    | 0000            |
| .    | I      | Y       | 0003 | STEP2    |          | OPCUTIL  | Normal    | Executed    | 0008            |
| .    | I      | Y       | 0004 |          | S3       | IEFBR14  | Normal    | Executed    | 0000            |


```

EQQCLEAN is inserted and default start step is selected

© Copyright IBM Corporation 2015

Step restart selection list panel

Select **Step Restart** to display the Step Restart Selection panel. After you select the restart step, issue the GO command. The Modifying the Cleanup Action panel opens if cleanup is necessary. You might have to rerun cleanup if you start from the selected step. The example shows that the EQQCLEAN utility inserted as the first step.

Modifying cleanup actions panel

```
EQQMCMDL ----- MODIFYING CLEANUP ACTIONS ----- Row 1 to 1 of 1
Command ==> go_                                         Scroll ==> PAGE

Enter GO to confirm the selection, DISCARD to exclude all the actions,
        END to save the selection, CANCEL to exit without saving.
Row commands:   I to include the data set in the actions
                  X to exclude the data set from the actions

Application      : U#CRITA          11/10/22 07.00
Operation        : U#CP 10
Jobname and jobid: U#CRITJ2       JOB00751



| Row cmd | Sel | ProcStep | Dataset name | Act | Volser | Prot |
|---------|-----|----------|--------------|-----|--------|------|
| I       | S1  |          | INGC103.JAY  | D   | SMS003 | N    |


```

Include this data set for cleanup

© Copyright IBM Corporation 2015

Modifying cleanup actions panel

After you select the job restart, you can override the scheduler default cleanup actions. The example has one data set allocated with a disposition of new in job step S1. The scheduler, by default, sets it for deletion. After you include or exclude the data sets, type **GO** and press Enter to continue.

You can modify or browse cleanup information from a number of places in the Tivoli Workload Scheduler for z/OS panel. In most cases, you can use the Handling Operations Ended in Error panel to browse, modify, run, or discard cleanup actions.

You have a choice of two row commands:

- **X**: command to exclude a data set entry from the cleanup action list
- **I**: command to insert a data set entry in the cleanup action list

Insert an **Act** command to show the action that Tivoli Workload Scheduler for z/OS should take for the data set:

- **D**: The data set is scratched and removed from the catalog. If a GDG, the generation of the data set is scratched, and the entry for the data set in the GDG base is removed.
- **K**: An uncataloged data set is deleted.
- **C**: The data set is recataloged.
- **U**: The data set is uncataloged.

The **Prot** option shows whether the data set is protected:

- **Y:** The data set is protected (specified in the initial parameters) and cannot be deleted.
- **N:** The data set is not protected.

Editing JCL from restart

```

EQQMMJCL ----- EDITING JCL FROM RESTART -----
Command ==> go_                                         Scroll ==> CSR
Edit JCL below: press PF3 to save, CANCEL to reject, GO to save and continue.

The confirmed JCL will be used for restart. The submitted JCL will be
also saved in JS whenever not expanded JCL is specified.

Application      : U#CRITA          11/10/22 07.00
Operation        : U#CP 10
Jobname and jobid : U#CRITJ2        JOB00751
JCL last updated by : INGC103       11/10/26 15.48
Expanded JCL     : N

000010 /**
000011 /**
000012 //*****                                                 *
000013 /**
000014 /* WHEN JOB FAILS WITH A JCL ERROR, DELETE THE DD1 STATEMENT AND THE *
000015 /* TWO LINES FOLLOWING IT                                     *
000016 /**
000017 //*****                                                 *
000018 /**
000019 //TIVDST00 OUTPUT JESDS=ALL,DEST=TWS           INSERTED BY TWS
000020 //TIVDSTAL OUTPUT JESDS=ALL                   INSERTED BY TWS
000021 //S1 EXEC PGM=IEFBR14
000022 /**
000023 //DATA01   DD DSN=INGC103.JAY,DISP=(NEW,KEEP,KEEP),
000024 //                  DCB=(RECFM=FB,LRECL=80,BLKSIZE=3120),
000025 //                  SPACE=(TRK,(1,1)),UNIT=SYSDA
000026 /**
000027 //STEP2   EXEC PGM=OPCUTIL,PARM=('/RC=08')

```

Include this data set for cleanup

© Copyright IBM Corporation 2015

Editing JCL from restart

The **Edit JCL** option was set to **Y** on the Operation Restart and Cleanup panel. Therefore, you can edit the JCL and correct the program name before the job is restarted. This slide shows an example JCL editing panel. The two lines that Tivoli Workload Scheduler inserted are for providing the job log to the Data Store started task.

Confirm restart panel

```
EQQMERTP ----- CONFIRM RESTART -----
Command ==> Y_                                         Scroll ==> PAGE

Enter Y in the command field to restart the operation.
The operation status will be set to Ready.
Enter N to reject the restart request.

Application      : U#CRITA      TEAM # Critical App
Operation        : U#CP 10     U#CRITJ2
Input arrival    : 11/10/22 07.00

Restart step      :          S1
End step          :          S3
Expanded JCL will be used : N
GDG simulation    : Y

Error code        ==> 0008
User data         ==> _____
Reason for restart ==> _____
```

© Copyright IBM Corporation 2015

Confirm restart panel

The Confirm Restart panel provides the opportunity to reject the restart action. Specify **N** to reject the restart, and **Y** to accept the restart.

Use these input fields in the following ways:

- **Error code:** Optionally change the error code that is assigned to the operation. Used for reporting purposes only.
- **User data:** Optionally change the user data field for the operation. Used for reporting purposes only.
- **Reason for restart:** Enter free-form text to describe the reason that the operation is being restarted. Used for reporting purposes only.

Cleanup results

```
EQQMCMBL ----- VIEW CLEANUP RESULTS ----- Row 1 to 1 of 1
Command ===> _
Scroll ===> PAGE

OPERATION DATA:

Application : U#CRITA          11/10/22 07.00
Operation   : U#CP 10
Jobname and jobid : U#CRITJ2      JOB00754

CLEAN UP DATA:

Clean up result : Completed
Clean up jobname and jobid : U#CRITJ2      JOB00754

Dataset name      Act    Volser   Result
INGC103.JAY       D      SMS003  DONE
```

Option 5 on the OPERATION RESTART AND CLEANUP panel

© Copyright IBM Corporation 2015

Cleanup results

Select option **5** from the Operation Restart and Cleanup panel to see the cleanup results. The job log is requested, and you enter option **5** a second time.

The **Action** column shows which of the following cleanup actions are taken for the data set:

- **D:** The data set was scratched and removed from the catalog.
- **U:** The data set was uncataloged.
- **C:** The data set was cataloged.

Lesson 4 Browsing the job log

Lesson 4 Browsing the job log

```
EQQMOPRL --- MODIFYING OPERATIONS IN THE CURRENT PLAN (left Row 16 to 21 of 21
Command ===>                                         Scroll ==> PAGE

Enter the GRAPH command above to view list graphically,
enter the HIST command to select operation history list, or
enter any of the following row commands:
J      - Edit JCL          M      - Modify          B      - Browse details
DEL   - Delete Occurrence MH    - Man. HOLD       MR    - Man. RELEASE oper
O      - Browse operator instructions NP   - NOP oper       UN   - UN-NOP oper
EX    - EXECUTE operation  D     - Delete Oper     RG    - Remove from group
L      - Browse joblog     K     - Kill            RI    - Recovery Info
RC   - Restart and CleanUp FJR   - Fast path JR   FSR   - Fast path SR
TCJ  - Target Critical Job FSC   - Fast path SC
BND  - Reset bind information

Row Application id  Operat Jobname Input Arrival Dep Cond Dep S
cmd ws no.        Date Time  Suc Pre  Suc Pre
... U#APPL1A        U#CP 025 SM40HJ5 11/10/22 06.00 0 2 0 0 W
... U#CRITA         U#CP 005 U#CRITJ1 11/10/22 07.00 2 0 0 0 C
L U#CRITA         U#CP 010 U#CRITJ2 11/10/22 07.00 0 1 1 0 E
... U#CRITA         U#CP 015 U#CRITJ3 11/10/22 07.00 1 1 1 1 W
```

Use ISPF option **5.3.0** and then the **L** row command

© Copyright IBM Corporation 2015

In this lesson, you learn how to use restart and cleanup to browse the joblog. After completing this lesson, you should be able to use restart and cleanup to browse the joblog,

In addition to restart and cleanup, you can use the data store to retrieve and browse the job log of an operation. You specify the **L** row command on the Modifying Operations in the Current Plan panel, which is at ISPF option =**5.3**.

Job log view (1 of 2)

```

ISRBROBA  SYS11300.T134625.RA000.INGC103.OPCAWRK1.H0 Line 00000000 Col 001 080
Command ==> _____ Scroll ==> PAGE
***** Top of Data *****
$SEQQFSWWU-START-JESMSG
J E S 2   J O B   L O G   --   S Y S T E M   C A 0 D   --   N O D E

13.16.11 JOB00754 ---- THURSDAY, 27 OCT 2011 ----
13.16.11 JOB00754 IRR010I USERID INGC109 IS ASSIGNED TO THIS JOB.
13.16.11 JOB00754 ICH70001I INGC109 LAST ACCESS AT 15:58:20 ON WEDNESDAY, OCTO
13.16.11 JOB00754 SHASP373 U#CRITJ2 STARTED - INIT 1 - CLASS A - SYS CA0D
13.16.11 JOB00754 EQQCN00I START CLEANUP AND/OR RET-CODE SIMULATION PROCESS(ES)
13.16.11 JOB00754 EQQCN12I DLTD D INGC103.JAY
13.16.11 JOB00754 EQQCN17I SMS003
13.16.11 JOB00754 EQQCN99I CLEANUP AND/OR RET-CODE SIMULATION PROCESS(ES) ENDED
13.16.12 JOB00754 - --TIMINGS (MINS.)--
13.16.12 JOB00754 -JOBNAME STEPNAME PROCSTEP RC EXCP CPU SRB CLOCK
13.16.12 JOB00754 -U#CRITJ2 EQQCLEAN EQQCLEAN 00 74 .00 .00 .00
13.16.12 JOB00754 -U#CRITJ2 S1 00 2 .00 .00 .00
13.16.12 JOB00754 -U#CRITJ2 STEP2 08 5 .00 .00 .00
13.16.12 JOB00754 -U#CRITJ2 S3 00 2 .00 .00 .00
13.16.12 JOB00754 -U#CRITJ2 ENDED, NAME-JOB # 4 TOTAL CPU TIME=
13.16.12 JOB00754 SHASP395 U#CRITJ2 ENDED
1500 105 00000000

```

Shows the EQQCLEAN step with return code 00

© Copyright IBM Corporation 2015

Job log view (1 of 2)

In the job log view for the restarted operation, you can see that EQQCLEAN ran with return code 00. Use the F8 key to scroll down in the job log.

Job log view (2 of 2)

```
ISRBROBA SYS11300.T134625.RA000.INGC103.OPCAWRK1.HO Line 00000122 Col 001 080
Command ===> _____ Scroll ===> PAGE
IEF237I JES2 ALLOCATED TO SYSDUMP
EQQCN00I START CLEANUP AND/OR RET-CODE SIMULATION PROCESS(ES).
IGD103I SMS ALLOCATED TO DDNAME SYS00001
IGD104I INGC103.JAY RETAINED, DDNAME=SYS00001
EQQCN12I DLTD D INGC103.JAY
EQQCN17I SMS003
EQQCN99I CLEANUP AND/OR RET-CODE SIMULATION PROCESS(ES) ENDED.
IEF142I U#CRITJ2 EQQCLEAN EQQCLEAN - STEP WAS EXECUTED - COND CODE 0000
IEF285I SYS2.OPC.LOADLIB PASSED
IEF285I VOL SER NOS= MVC106.
IEF285I INGC109.U#CRITJ2.JOB00754.D0000101. ? SYSIN
IEF285I INGC109.U#CRITJ2.JOB00754.D0000102. ? SYSIN
IEF285I INGC109.U#CRITJ2.JOB00754.D0000103. ? SYSOUT
IEF285I INGC109.U#CRITJ2.JOB00754.D0000104. ? SYSOUT
IEF285I INGC109.U#CRITJ2.JOB00754.D0000105. ? SYSOUT
IEF285I INGC109.U#CRITJ2.JOB00754.D0000106. ? SYSOUT
IEF285I INGC109.U#CRITJ2.JOB00754.D0000107. ? SYSOUT
IEF373I STEP/EQQCLEAN/START 2011300.1316
IEF032I STEP/EQQCLEAN/STOP 2011300.1316
```

Shows EQQCLEAN deleted the data set INGC103.JAY

© Copyright IBM Corporation 2015

Job log view (2 of 2)

EQQCLEAN deleted the INGC103.JAY data set as part of the cleanup action.

Student exercises



See the Student Exercise Guide Unit 7

© Copyright IBM Corporation 2015

Student exercises

Perform the exercises for this unit.

Summary

- List the tasks that the restart and cleanup functions can perform
- Specify appropriate restart and cleanup options for selected operations
- Use the Restart and Cleanup panel for these tasks :
 - Perform job or step-level operation restarts
 - Override data set cleanup actions

© Copyright IBM Corporation 2015

Summary

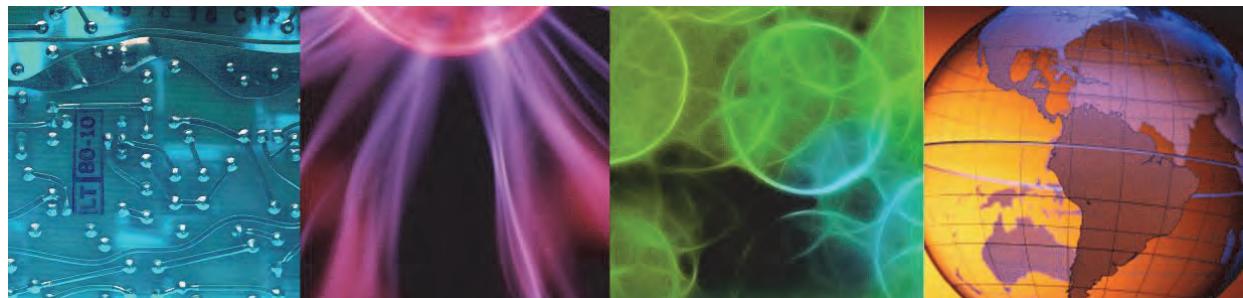


8 Special resources

IBM Tivoli Workload Scheduler for z/OS 9.2.0



8 Special resources



© Copyright IBM Corporation 2015

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

This unit describes the implementation and use of special resources.

References:

- SC32-1262 *IBM Tivoli Workload Scheduler for z/OS Managing the Workload*
- SC32-1265 *IBM Tivoli Workload Scheduler for z/OS Customization and Tuning*



Objectives

- Describe how to use special resources
- Define special resources in the database
- Use the Special Resource Monitor

© Copyright IBM Corporation 2015

Objectives

Sometimes a job has other dependencies than just another job, sometimes it waits for a file, or perhaps it cannot run when another job is executing. This unit shows how special resources can be used to cover any additional dependencies that your jobs may have. During the hands-on exercise you create special resources and use them on a new set of jobs that you add to the current plan and run.

Lesson 1 Special Resource Overview

Lesson 1 Special resources overview

- A special resource has a symbolic name for a logical entity that represents a limited resource of any type
- A special resource can logically represent physical computer system devices and data
 - Tape drives
 - Databases or data sets
 - You can define a special resource as part of a group of related or similar resources
 - You can synchronize and serialize the running of jobs
- A special resource differs from the R1 and R2 resources associated with a computer workstation

© Copyright IBM Corporation 2015

In this lesson, you learn the basics concepts behind special resources. After completing this lesson, you should be able to describe the basics concepts behind special resources.

You can use Tivoli Workload Scheduler for z/OS special resources instead of workstation fixed resources **R1** and **R2** to manage limited resources in your scheduling environment. The purposes include the following possibilities:

- Logical representations of any type of limited resource.
- Limited resources such as tape drives, communication lines, or databases. You create resources by using the Special Resources panel.
- Modeling of logical constructs, such as the following types of jobs:
 - Certain jobs, although they are not interdependent, that cannot be run at the same time.
 - A backup job that must not be started when a specified subsystem is active

Special resources are logical representations of scheduling requirements. You can use these logical entities to represent any type of limited resource, such as tape drives, communication lines, data sets, databases, and JES initiators.

Special resource characteristics

- Can restrict operations from running together:
 - Serialization of operations
 - Resource contention
- Can be used to implement special scheduling requirements
 - Can help in the coordination of running operations
 - Can synchronize running operations
- Can be dynamically created
- Availability can be dynamically adjusted
- Can be monitored separately
 - Tivoli Workload Scheduler for z/OS ISPF panels
 - Dynamic Workload Console (DWC - WebUI)
 - Program interface

© Copyright IBM Corporation 2015

Special resource characteristics

You use special resources to describe special scheduling requirements for your environment. You can also use special resources to serialize the running of work. Define these resources in the special resource database or dynamically create them.

Tivoli Workload Scheduler for z/OS daily planning functions generates utilization reports for special resources that are associated with operations in the current plan. You can use the Special Resource Monitor panel to monitor and modify special resources in the current plan.

You can use special resources with Tivoli Workload Scheduler for z/OS operations and Tivoli Workload Scheduler jobs. However, you lose the fault tolerance of the distributed agent if you use special resources with distributed jobs. Only the controller can determine the availability of a resource and signal the distributed agent to start the operation. You can switch the availability of a special resource at operation completion.

Special resources database properties

- Resource name
- Availability
- Amount
- Planning and control
- Action on error
- Action on complete
- Max usage limit
- Max usage type
- Hiperbatch
- Time intervals
- Workstations

© Copyright IBM Corporation 2015

Special resource database properties

You create resources by using the Special Resource panel. The Special Resource panel updates the resource database, which has these details of each resource:

- **Name:** Up to 44 characters
- **Availability:** Yes (Y) or no (N)
- **Quantity:** 1 - 999999
- **Used for:** Enter one of the following options to show how Tivoli Workload Scheduler for z/OS is to use the resource (do not use planning or both for resources that might become available or unavailable through external actions):
 - **P** (planning, when the current plan is extended)
 - **C** (control, when an operation starts)
 - **B** (both)
 - **N** (neither)
- **On-error action:** Subsequent actions if use of the resource in an operation results in errors. The operation can have an overriding keep-on-error specification in the operation definition. You can also specify an option for the resource.

The possible options are as follows:

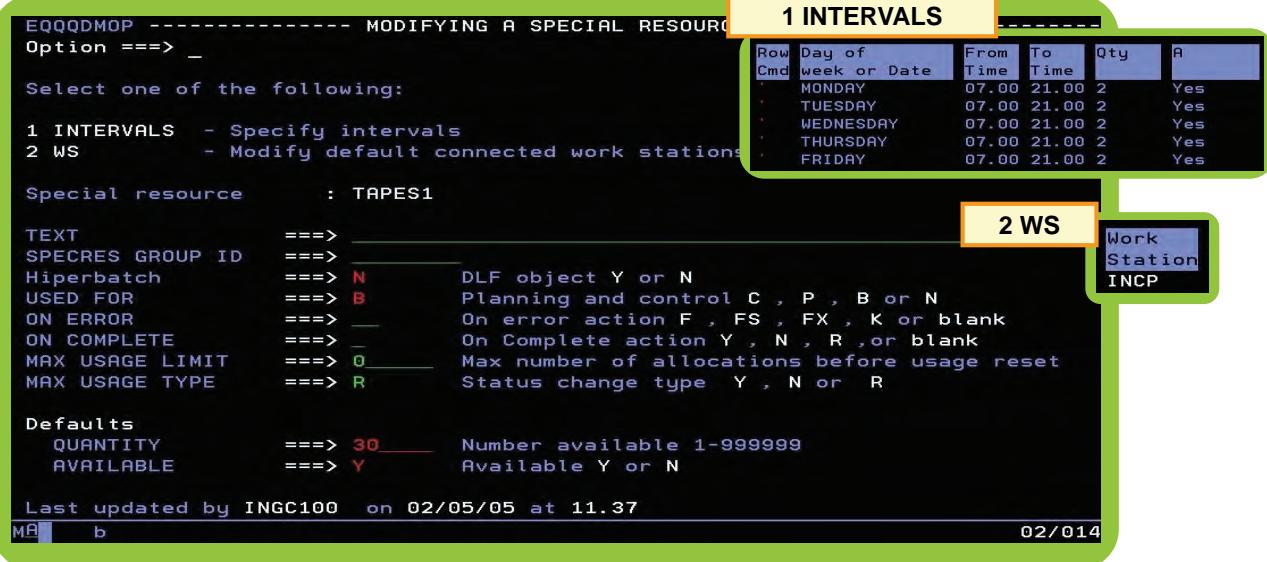
- **F:** Free its full allocation of this resource, resources allocated exclusive, and allocated shared.
- **FS:** Free its full shared allocation of this resource.

- **FX:** Free its full exclusive allocation of this resource.
- **K:** Keep its full allocation of this resource.
- **Blank:** Use the default that is specified in the ONERROR keyword of the RESOPTS statement. (Refer to the *Tivoli Workload Scheduler for z/OS Customization and Tuning Guide*.)
- **ON COMPLETE:** Set the global availability as follows when an operation that used the resource completes:
 - **Y:** Set the global availability to YES.
 - **N:** Set the global availability to NO.
 - **R:** Reset the global availability to blank.
 - **Blank:** Use the default from the operation, if it is not blank. If the operation default is blank, use the value from the special resource definition. If the special resource definition value is blank, use the value from the default that is specified in the RESOPTS initialization statement.
- **MAX USAGE LIMIT:** Specify the maximum number of allocations (regardless of quantity) that must be reached, before the resource global availability is changed to the MAX USAGE TYPE value. An internal usage counter increases each time that an operation allocates the resource. When this internal counter reaches the MAX USAGE LIMIT value, the global availability changes as specified by MAX USAGE TYPE. The default is **0**, and means that no usage counter check occurs.
- **MAX USAGE TYPE:** Indicate how to change global availability when the MAX USAGE LIMIT occurs. MAX USAGE TYPE is optional and is valid only if MAX USAGE LIMIT is not 0.
 - **Y:** Change global availability to Available.
 - **N:** Change global availability to Not Available.
 - **R:** Reset global availability to blank.
- **Connected workstations:** View a list of the workstations where operations can allocate the resource.

The quantity, availability, and list of workstations can vary with time. You can create time **intervals** to control each special resource.

Lesson 2 Creating special resources

Lesson 2 Creating special resources



© Copyright IBM Corporation 2015

In this lesson, you learn how to create special resources. After completing this lesson, you should be able to create special resources.

Use this panel to create special resources. You use the **Intervals** option to create time intervals that you can specify quantity and availability attributes for the resource. You use the **WS** option to specify the names of workstations that can allocate (be connected to) the special resource. If you do not use this option, all workstations allocate the resource. The default values specify the allocated availability and quantity of the resource for undefined time intervals.

Specifying availability intervals

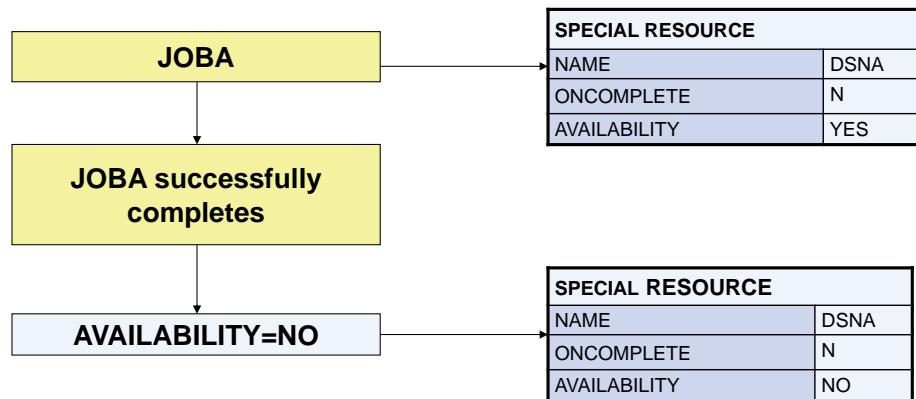
Use this panel to specify availability and quantity for time intervals you specify. Tivoli Workload Scheduler for z/OS automatically allocates the resource based on your specifications. The values that are specified on this panel override the defaults. If necessary, you can type the **Work stations** row command **S** for an interval. The **S** command shows the panel on that you can specify the workstations that are connectable to the special resource.

Specifying workstation connections

On the workstation connections panel, you specify the workstations that can allocate the special resource. You can specify workstations for a specific time interval. You can also select default workstations for the special resource with option **2 (WS)** on the Creating a Special Resource panel.

Special resource ON COMPLETE example

ONCOMPLETE processing at successful job completion



© Copyright IBM Corporation 2015

Special resource ON COMPLETE example

ON COMPLETE determines how special resource availability is to change for a successful job completion. You can specify the **ON COMPLETE** option at the special resource, operation, and initial parameter levels. At the *special resource* and *operation* level, the values for the **ON COMPLETE** option areas follows:

- **Y:** Set global availability to YES at end of job.
- **N:** Set global availability to NO at end of job.
- **R:** Reset the global availability to blank.
- **Blank:** Use the system defaults.

At the special resource and operation level, you can specify the values for the **ON COMPLETE** option by using the following interfaces:

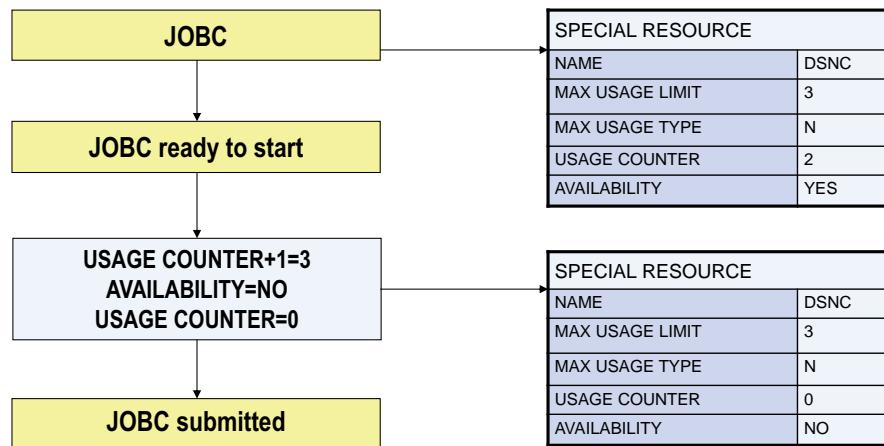
- ISPF panels
- Batch loader
- Batch Command Interface Tool (BCIT)
- Programming Interface (PIF)
- OCL (Command Language)

Using the RESOPTS initialization parameter, you can define default special resource options as follows:

- **ONCOMPLETE**: Statically defined special resources
- **DYNONCOMPLETE**: Dynamically defined special resources
- Values for both ONCOMPLETE and DYNONCOMPLETE:
 - **NOCHANGE**: Leave the global availability unchanged (the default).
 - **YES**: Set global availability to YES at the end of the job.
 - **NO**: Set global availability to NO at the end of the job.
 - **RESET**: Reset the global availability to blank. Blank is the default.

Special resource MAX USAGE example

Max usage processing at job start



© Copyright IBM Corporation 2015

Special resources MAX USAGE example

You can use a count of the number of jobs that used a special resource to switch its availability. At job start, the usage counter is checked to see whether the maximum limit is reached. If it is, the special resource availability is switched.

You can specify the special resource maximum usage (Max Usage) values at the special resource level by using the following interfaces:

- ISPF panels
- Programming Interface (PIF)

The controller updates and internally controls the usage counter and counts special resource allocations at the start of each operation. The usage counter updates only if the max usage limit is set to a value other than 0. The Browsing a Special Resource panel shows the counter.

Max Usage Limit indicates the maximum usage count to be exceeded before the special resource availability to switch to the status specified in Max Usage Type.

Max Usage Type indicates how special resource global availability status changes after the Max Usage Limit value is reached. Valid values are as follows:

- **YES**: Set global availability to YES at the start of the job.
- **NO**: Set global availability to NO at the start of the job.
- **RESET**: Reset the global availability to blank (default).

Creating resources dynamically

BATCHOPT
DYNAMICADD(YES)

RESOPTS
DYNAMICADD(YES/EVENT/OPER)

SRSTAT CREATE(YES)
LIFESPAN(*nnnnn*, new availability value)

© Copyright IBM Corporation 2015

Creating resources dynamically

If one of the three following situations arises, Tivoli Workload Scheduler for z/OS can dynamically create the specified special resource in the current plan:

1. Daily planning discovers that an operation needs a resource that is not defined in the special resource database.
2. An operation or event refers to a resource that is in the database but is not in the current plan.
3. An operation or event refers to a resource that is not in the database or the current plan.

In the first situation, if **YES** is specified for the DYNAMICADD parameter in the BATCHOPT statement, the scheduler creates a special resource during planning. The special resource database is not updated. However, the special resource is added to the special resource monitor in the current plan.

In the second situation, the scheduler creates a special resource in the current plan if it is defined in the database but was not added during daily planning. Resources are added only during daily planning if an operation in the plan refers to the resource. When you refer to the resource during the life of the plan, the scheduler adds the resource from the database.

In the third situation, you must set the keyword value for the DYNAMICADD parameter in the RESOPTS statement to **YES** or **OPER**. Change the keyword if you add an occurrence to the current plan by using the ISPF panels and one of the operations uses a special resource.

SRSTAT is a TSO-authorized scheduler command for managing special resources. You can use SRSTAT to create a special resource. To use SRSTAT to create special resources, set the CREATE

keyword of SRSTAT to **YES**. The DYNAMICADD keyword of RESOPTS must also be set to **EVENT** or **YES**. You can change the LIFESPAN parameter to control availability based on a time interval.

SRSTAT LIFESPAN parameter

SRSTAT LIFESPAN (*nnnnn, availability_value*)

- *nnnnn*: Number of minutes until expiration 0 to 99999 in minutes
- *availability_value*: Special resource availability is set to this value when the expiration value is reached
 - YES Set global availability to YES
 - NO Set global availability to NO
 - RESET Reset the global availability to blank (default)

© Copyright IBM Corporation 2015

SRSTAT LIFESPAN parameter

By setting the LIFESPAN parameter, you can switch the special resource availability based on an expiration time. Specifying LIFESPAN with a time interval of **0** deletes a pending LIFESPAN. Each special resource can have only one active LIFESPAN. A new LIFESPAN specification replaces a pending LIFESPAN. Specifying LIFESPAN with a time interval of **0** deletes a pending LIFESPAN.

Two ways to specify LIFESPAN for a special resource are as follows:

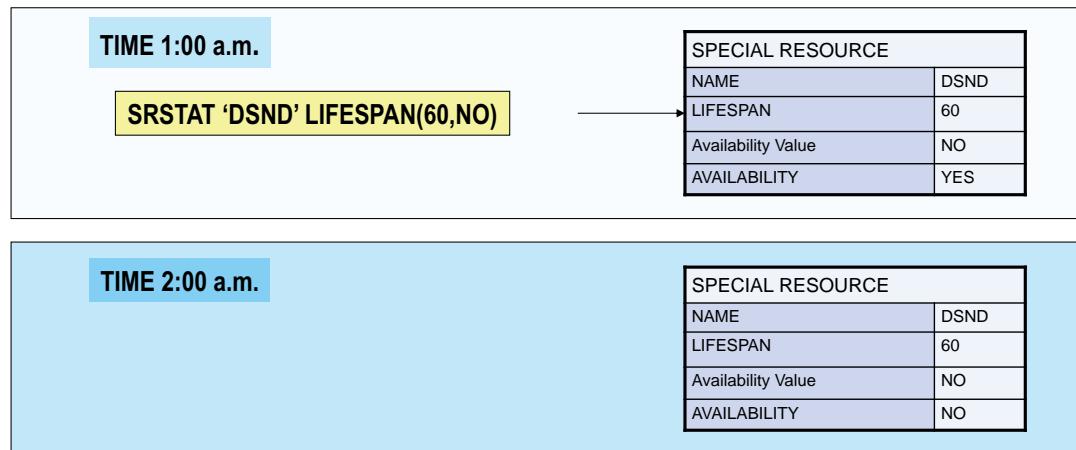
- **SRSTAT command**: A TSO command to change the status of a special resource has a new LIFESPAN parameter. The SUBSYS parameter of the SRSTAT command must specify the tracker subsystem that the SRSTAT command is directed to.
- **Data set triggering**: A special resource event generated by data set triggering can have an expiration time.

The availability of the special resource that is generated by dataset triggering can be switched based on specified LIFESPAN values. The LIFESPAN values are in the data set triggering selection table EQQDSLST.

RESOPTS LOOKAHEAD takes LIFESPAN into account when considering the future availability of a resource.

Special resource LIFESPAN example

A LIFESPAN processing example



© Copyright IBM Corporation 2015

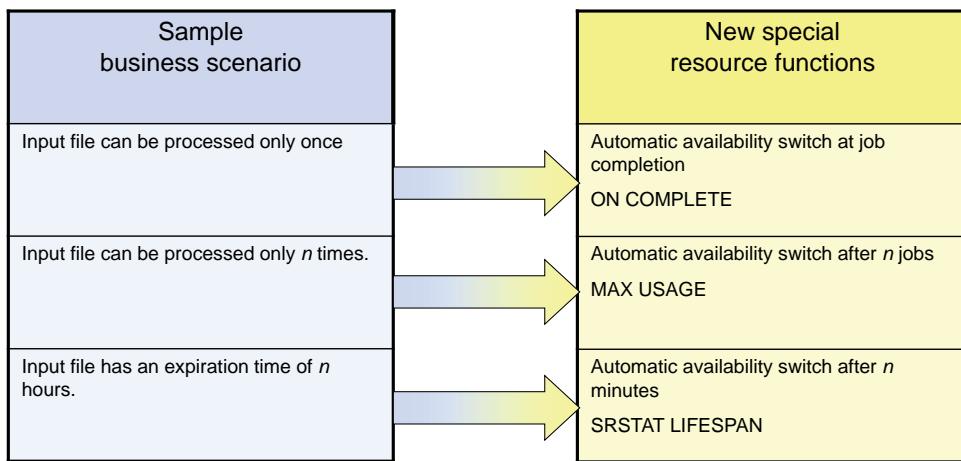
Special resource LIFESPAN example

This example shows how you can use the SRSTAT LIFESPAN parameter to change the availability of a special resource.

Lesson 3 Using special resources in operations

Lesson 3 Using special resources in operations

Special resource scenarios



In this lesson, you learn how to use special resources in operations. After completing this lesson, you should be able to use special resources in operations.

The purpose of a special resource is to assist with the scheduling of operations. This slide provides several possible scenarios for using a special resource to help control the scheduling of operations.

Specifying operation resource requirements

- Resource name
- Quantity required
- Exclusive or shared use
- Keep on error
- Available on complete

© Copyright IBM Corporation 2015

Specifying operation resource requirements

You use operation details option **3** to define special resources for an operation with these options:

- **Resource name:** Specify the name of the resource, up to 44 characters. If you are unsure of the name, you can use the global search characters of the percentage sign (%) and asterisk (*). Tivoli Workload Scheduler for z/OS lists matching names that you can select from the resources that you need.
- **Quantity required:** The number of resources that the operation allocates. If you leave this field blank, the operation is allocated the whole quantity currently available.
- **Exclusive or shared use:** Specifies whether an operations can use a special resource unit that is allocated to another operation as shared at the same time. The resource can not be used by operations that need exclusive use, as resource units that are allocated as exclusive can be used only by that operation. An **S** specifies that the resource can be shared. An **X** specifies that the operation requires the resource exclusively.
- **Action on error:** Specify whether the special resource remains allocated if the operation fails. If you leave this option blank, Tivoli Workload Scheduler for z/OS takes the default for the resource. Type **Y** in this field to keep the resource if the operation fails. Type **N** in this field to free the resource if the operation fails.
- **Avail On Complete:** Specify whether to set the global availability to **Y** (yes), **N** (no), or **R** (reset) at job completion. If you leave the field blank, the default is the value either from the special resource definition, if it is not blank, or from the RESOPTS initialization statement.

Assigning a special resource to operations

```
EQQAMSRL ----- SPECIAL RESOURCES ----- Row 1 to 1 of 1
Command ==> _

Enter/Change data in the rows, and/or enter any of the following
row commands:
I(nn) - Insert, R(nn),RR(nn) - Repeat, D(nn),DD - Delete

Operation : U#CP 015

Row  Special          Qty   Shr   Keep On   Avail on
cmd  Resource          Ex    Error  Complete

...  SM40#TAPES       3     X     Y      Y
```

Use operation details option 3

© Copyright IBM Corporation 2015

Assigning a special resource to operations

This example shows how to assign a special resource to an operation. The operation in this example shows settings for exclusive use of one instance of the special resource named SM40#TAPES.



Note: The daily planning batch function generates a planned special resource utilization report.

Special resource monitoring

- Monitors and manages resources in current plan
- Can change amount and availability
 - Manually
 - By other programs
 - By RODM
- Lists operations waiting for resource
- Lists operations using the resource

© Copyright IBM Corporation 2015

Special resource monitoring

You use the Special Resource Monitor to monitor and manage special resources in the current plan. The Special Resource Monitor shows a list of resources that operations in the current plan need. Even though operations automatically hold and release resources according to their descriptions in the current plan, you must monitor these resources. You monitor special resources in case events occur that you want to modify instructions for.

Assume that a special resource is created to represent the tape drives reserved for Tivoli Workload Scheduler for z/OS scheduled jobs. The current plan is built assuming that a specific number of tape drives are available. If one of the tape drives is taken physically offline, the scheduler no longer has a true picture of the environment. Unless you notify Tivoli Workload Scheduler for z/OS, it continues to allocate the offline tape drive to jobs that need a tape drive. After being submitted, the jobs wait for the offline tape drive.

You can prevent the scheduler from starting jobs that wait for an offline tape drive and use operating system resources. You can use the Special Resource Monitor panel to reduce the number of tape drives by 1. You specify a deviation of **-1** (minus 1) to show that the quantity is reduced. When the tape drive is online again, you reset the deviation from the Special Resource Monitor panel.

Special resource monitor: Waiting queue

```

EQQQMLSL ----- SPECIAL RESOURCE MONITOR ----- Row 1 to 1 of 1
Command ==> Scroll ==> PAGE

Enter any of the row commands below:
B - Browse, M - Modify, I - In use list, W - Waiting queue

R Special
Resource
W TEAM#DSN      A RDM Adjust Used Shared Used Excl W
N NNN 1          AAQD Qty 0 0 Y

EQQQMWM1 ----- SPECIAL RESOURCE MONITOR - WAITING QUEUE - Row 1 to 2 of 2
Command ==> _

Enter any of the row commands below:
S - Select details, D - Delete from queue

Special resource : TEAM#DSN
Text : Data Set used by TEAM#

Row Latest Out Operation Jobname Pri Qty Type Reason
cmd Date Time ws no.          8   1    X  UNAVL
... 11/10/22 23.01 U#CP 015 SM40#MS2
... 11/10/22 23.02 U#CP 005 SM40#MS1

```

© Copyright IBM Corporation 2015

Special resource monitor: Waiting queue

Use the Tivoli Workload Scheduler for z/OS ISPF option **5.7.0** to view or modify special resources in the current plan with these options:

- Browse (**B** row command)
- Modify (**M**)
- List the in-use (**I**)
- Show the waiting queue (**W**)

Modifying a special resource

```
EQQQMMOP ----- MODIFYING A SPECIAL RESOURCE -----
Option ==>
Select one of the following:
1 INTERVALS - Specify intervals
2 WS         - Modify default connected work stations

Special resource      : TEAM#DSN
Text                  : Data Set used by TEAM#
Specres group id    :
Hiperbatch           : No          Usage Counter   : 0

USED FOR             ==> B      Planning and control C , P , B or N
ON ERROR             ==> _      On error action F , FS , FX , K or blank
DEVIATION            ==> _____ Number to deviate -999999 to 999999 or blank
AVAILABLE            ==> N      Global availability Y or N or blank
QUANTITY             ==> _____ Global quantity 1 to 999999 or blank
ON COMPLETE          ==> _      On Complete action Y or N , R , or blank
MAX USAGE LIMIT     ==> 3      Max number of allocations before usage reset
MAX USAGE TYPE       ==> N      Status change type Y , N or R
Defaults
  QUANTITY           ==> 1      Number available 1-999999
  AVAILABLE          ==> Y      Available Y or N
Active LIFESPAN      : Action=   Expiration Date=
Last updated by TWSz on 11/10/22 at 15.16 Max Usage Limit change
```

© Copyright IBM Corporation 2015

Modifying a special resource

This example shows the results of a Modify row command (**M**).

Student exercises



See the Student Exercise Guide Unit 8

© Copyright IBM Corporation 2015

Student exercises

Perform the exercises for this unit.

Summary

- Describe how to use special resources
- Define special resources in the database
- Use the Special Resource Monitor

© Copyright IBM Corporation 2015

Summary

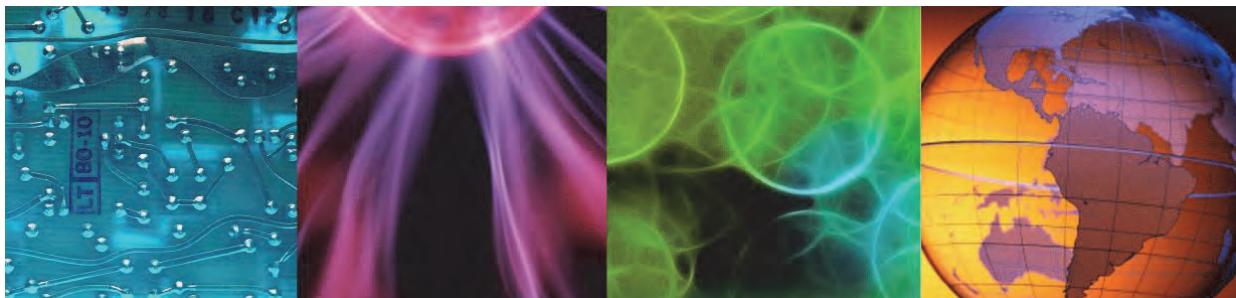


9 Automated job tailoring

IBM Tivoli Workload Scheduler for z/OS 9.2.0



9 Automated job tailoring



© Copyright IBM Corporation 2015

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

In this unit, you learn about automated job tailoring, to automatically change jobs by using Tivoli Workload Scheduler for z/OS Job Control Language (JCL) variables and directives.

References

- SC32-1262 *IBM Tivoli Workload Scheduler for z/OS Managing the Workload*
- SC32-1265 *IBM Tivoli Workload Scheduler for z/OS Customization and Tuning*

Objectives

- List the main goal of automated job tailoring
- Tailor the job before submitting it with JCL automation directives
- Create and use Tivoli Workload Scheduler for z/OS JCL variables

© Copyright IBM Corporation 2015

Objectives

Job Control Language, JCL, might need updating, daily, periodically, when a particular event happens. This unit teaches you how Tivoli Workload Scheduler for z/OS can automate much, if not all, of the update requirements. You will learn how to use the supplied variables, how to make conditional changes in the JCL, and how to build user defined variable tables. A hands-on exercise will build jobs to run that combine the JCL manipulation skills learned here with other skills learned on this course.

Lesson 1 Automated job tailoring overview

Lesson 1 Automated job tailoring overview

- Automatic editing of job JCL at job setup time or job submit time
- Dynamic inclusion and exclusion of inline job statements with special inline statements called directives
- Dynamic inclusion of job statements from the EQQJBLIB data set or from an user exit routine
- Automatic or manual (operator prompt) variable substitution

© Copyright IBM Corporation 2015

In this lesson, you are presented with an overview of job tailoring features that are provided by Tivoli Workload Scheduler for z/OS. After completing this lesson, you should be able to describe the job tailoring features that are provided by Tivoli Workload Scheduler for z/OS.

Tivoli Workload Scheduler for z/OS provides automatic job tailoring facilities for the automatic editing of jobs by using information that is known only at job setup or submit time. Tivoli Workload Scheduler for z/OS provides the following automated job tailoring uses:

- Dynamic inclusion and exclusion of inline job statements.
- Dynamic inclusion of JCL from the Tivoli Workload Scheduler for z/OS job library or a user exit routine.
- Dynamic creation of temporary variables.
- Date arithmetic by using Tivoli Workload Scheduler for z/OS supplied variables.
- Variable substitution of user-defined or Tivoli Workload Scheduler for z/OS supplied variables.

Automated job tailoring can reduce your dependency on manual editing of jobs. User-defined variable details are tables that are stored in the database. Access the database by using option 9 (JCLVAR) on the database panel.

For jobs submitted on a z/OS system, the job statements include z/OS JCL. However, you can include schedule JCL tailoring directives in jobs to be submitted on other supported operating systems. The directives have the same format for all supported operating systems.

Another method for substituting variables is using the Tivoli Workload Scheduler for z/OS program interface (PIF). User programs issue requests to the Tivoli Workload Scheduler for z/OS subsystem by using the PIF. These requests can perform variable substitution. See the PIF resource JCLPREPA in the Programming Interfaces manual.

JCL directives overview

Used for controlling the automatic editing of the JCL of a job

- The ///*%OPC prefix denotes directive statements in the JCL of a job
- Provides variable substitution control
 - Not required for variable substitution; optionally used for controlling variable substitution
 - Modifies the search path for finding variable replacement values
- Conditionally includes and excludes job statements
- Enables date manipulation in JCL

© Copyright IBM Corporation 2015

JCL directives overview

Tivoli Workload Scheduler for z/OS uses special comment statements, called directives, to manage the inclusion and exclusion of lines and to control aspects of variable substitution. See the IBM Tivoli Workload Scheduler Managing the Workload manual for a complete description of these job-tailoring features.

Based on the directives that are used, Tivoli Workload Scheduler for z/OS excludes lines in job JCL by skipping them at job setup or at job submit. By using directives, you can include job statements from an EQQJBLIB library member or supply them in a user-defined JCL exit routine.

JCL variables overview

- Activated with the OPCOPTS VARSUB initialization statement
OPCOPTS VARSUB(YES,NO,SCAN)
- Error handling is controlled with the OPCOPTS VARFAIL initialization statement
OPCOPTS VARFAIL(&, %, ?)
- Inline procedures can use variable substitution
OPCOPTS VARPROC(YES,NO)
- JCL variables and their values can be as follows:
 - Tivoli Workload Scheduler for z/OS supplied (standard)
 - User-defined with the ISPF panels or a user-written exit routine
 - Operator-supplied, with the operator being prompted for the value

© Copyright IBM Corporation 2015

JCL variables overview

You can create variables in job statements, in comment statements, and in any in-stream data within the job. You can use Tivoli Workload Scheduler for z/OS variables within cataloged or in-stream procedures if you set the VARPROC keyword of the OPCOPTS statement to YES. Variables occurring in comments are substituted (including comments to the right of job statements). When you create a variable by using the JCL variable definition panel, you also specify when you want substitution to occur. The scheduler performs the substitution at job setup time, at job submission, or both. It works the same for started tasks and system automation commands as it does for jobs.

You can specify that the variable values be supplied in any of the following ways:

- Manually.
- With a user-written exit routine.
- Automatically from information in the Tivoli Workload Scheduler for z/OS database user-defined tables.
- Using a variable that the operator must supply, which is called an operator-prompted variable.

Lesson 2 JCL directives

Lesson 2 JCL directives

- FETCH
- BEGIN and END
- SETFORM
- SETVAR
- SEARCH
- TABLE
- SCAN

© Copyright IBM Corporation 2015

In this lesson, you learn how to define and use JCL directives. After completing this lesson, you should be able to define and use JCL directives.

Tivoli Workload Scheduler for z/OS can dynamically include or exclude job statements during job setup and job submission. The directives that are shown on this slide can incorporate into the JCL of a job to control how the JCL is dynamically modified. See the Managing the Workload manual for detailed information about these directives.

The currently supported directives are as follows:

- FETCH: Names a PDS member containing job statements to be retrieved for inclusion.
- BEGIN and END: Mark the start and end.
 - JCL variable substitution
 - Lines of job statements to include or exclude
- SETFORM: Defines the format of supplied dynamic-format date and time variables.
- SETVAR: Creates temporary variables by using arithmetic or substring expressions together with supplied date variables or substring and concatenation actions on any variables.
- SEARCH: Defines the variable tables that are searched when attempting to resolve a variable.
- TABLE: Defines a variable table that is searched before the variable tables in any existing concatenation when resolving JCL variables.
- SCAN: Indicates the start of JCL substitution.

Identifying directive statements

In columns 1 to 7:

Before job tailoring

//%OPC

After job tailoring

/*>OPC

Changed by Tivoli Workload Scheduler for z/OS after successful execution of the directive

© Copyright IBM Corporation 2015

Identifying directive statements

The general syntax of a directive is as follows:

- Each directive must begin on a new 80-byte line.
- All directives begin with //%OPC in columns 1 - 7, followed by at least one space.
- You can code directive parameters in any order.
- Directive parameters can occur more than one time in the same directive.
- Directive parameters are separated by commas with no embedded blanks between parameters on the same line.
- If more than one parameter value is specified, parentheses are required. For example:

This example is correct: NAME=TABLE1

This example is incorrect: NAME=TABLE1, TABLE2

Instead, define the parameter as follows: NAME=(TABLE1, TABLE2)

- A directive specification cannot exceed 71 characters. It can continue on a new line if the directive continued by a comma after a complete parameter or subparameter.
Positions 72 - 80 are ignored.
- Each continuation line must begin with //%OPC in columns 1 - 7, followed by at least one space.
- If the directive completes successfully, the //%OPC is changed to /*>OPC

Dynamic inclusion and exclusion

Two methods available

FETCH

BEGIN
job statement
END

© Copyright IBM Corporation 2015

Dynamic inclusion and exclusion

You can use two methods of directives for including and excluding job statements when you use the scheduler automated job tailoring:

- With the FETCH Directive, you include JCL statements that the scheduler fetches from a partitioned data set member. A JCL-specified exit routine can also supply the JCL.
- BEGIN and END are used in pairs. You use them with the ACTION keyword and a value to include or exclude the job statements within the lines that are marked by the directives.

The FETCH directive

Conditional fetching of job statements from the following locations:

- Job library in EQQJBLIB concatenation
- A user exit routine

Example:

```
// *%OPC FETCH MEMBER=AUDTFILE, PHASE=SUBMIT,  
// *%OPC     COMP=(&OADDID..EQ. (SM4P#DLYW#OPC01) )
```

© Copyright IBM Corporation 2015

The FETCH directive

MEMBER=member_name defines the PDS member that is retrieved from a data set in the scheduler job library concatenation, EQQJBLIB. The lines in the member are included immediately after the FETCH directive. You can use an exit routine to fetch the member instead. Use the keyword EXIT=exit_name instead of MEMBER=member_name.

PHASE=SUBMIT specifies that the FETCH takes effect during the submit phase of the operation. The alternative is at setup time, which involves manual intervention.

The FETCH directive is normally used with the keyword COMP. Specify the keyword COMP=(expression1.EQ.(expression2)) to test expressions for the conditional invocation of directives. This example compares the name of the application occurrence with the name of a specified application. For more information about the COMP keyword, See Managing the Workload.

If the application occurrence name is SM4P#DLYW#OPC01, the scheduler performs the fetch.

BEGIN and END directives

- Marks sections of inline statements
- Conditionally includes or excludes marked job statements

Example:

```
//*%OPC BEGIN ACTION=INCLUDE, PHASE=SUBMIT,  
//*%OPC           COMP=(&OWW..NE.(&CWW.))  
//EQQTROUT DD DSN=%GDGDS.TRKLOG(+1),  
//               DISP=(NEW,CATLG),SPACE=(CYL,(2,2),RLSE),  
//               UNIT=SYSDA,  
//               DCB=(LRECL=32000,RECFM=VBS)  
//*%OPC END ACTION=INCLUDE
```

© Copyright IBM Corporation 2015

Begin and End directives

Unlike the FETCH directive, the job statements are not retrieved; they are already in the job. The BEGIN and END pair are typically used with the COMP keyword. BEGIN and END directives that specify ACTION=INCLUDE or ACTION=EXCLUDE cannot be nested and cannot overlap.

- BEGIN ACTION=INCLUDE marks the start of a set of job statements that you want Tivoli Workload Scheduler for z/OS to include in the job.
- END ACTION=INCLUDE marks the end of the set of job statements that you want Tivoli Workload Scheduler for z/OS to include in the job.

Date manipulation directives

SETFORM

SETVAR

© Copyright IBM Corporation 2015

Data manipulation directives

You can use Tivoli Workload Scheduler for z/OS supplied date variables and two date manipulation directives to automatically set up dates in jobs. The two date manipulation directives are as follows:

- SETFORM Defines the format of dynamic-format supplied variables.
- SETVAR creates a temporary variable by using an arithmetic expression together with supplied date variables.

The SETFORM directive

Dynamically formats dates to a user-specified form

Example:

```
/* This example demonstrates using the SETFORM directive
/* to dynamically change the printed format of dates
/* and times.
/*
/*%OPC SETFORM OCDATE=(MM:DD:YY)
/* The IA date for this occurrence is: &OCDATE.
/*
RESULT
/* The IA date for this occurrence is: 05:07:07
```

© Copyright IBM Corporation 2015

The SETFORM directive

The SETFORM directive format is SETFORM OCDATE=(MM:DD:YY), with the following elements:

- OCDATE is a dynamic-format supplied variable that, when substituted, provides the occurrence input arrival date.
- The (MM:DD:YY) is the user-specified format, by using date-pertinent keywords, in which the occurrence input arrival date is shown.

The set form directive format expression can contain a combination of time-pertinent keywords, date-pertinent keywords, and delimiters. The date-related keywords are as follows:

- CC: Represents the century; it is used with YY to define the format of a full year, such as 2014.
- YY: Represents the last two figures in the year.
- MM: Represents the month.
- DDD: Represents the day in the year. It is substituted before DD.
- DD: Represents the day in the month.

The time-pertinent keywords are as follows:

- HH: Represents the hour.
- MM: Represents the minutes.

Any other characters in the format expression are interpreted as delimiters.

The SETVAR directive

- This directive creates a temporary variable by using one of the following values: An arithmetic expression together with supplied date or time variables.
- A substring of another variable.
- The result of an arithmetic addition or subtraction.
- Concatenated strings or variables set to an alphanumeric value.

```
/* This example demonstrates date arithmetic
/* The IA date for this occurrence is: &OYMD2.
/* Then on checks to be printed set the date to 2 work days
/* from the IA date.
/*%OPC SETVAR TVAR=(OYMD2+2WD)
/* The date on the checks will be: &TVAR.

RESULT
/* The IA date for this occurrence is: 02/05/07
/* Then on checks to be printed set the date to 2 work days
/* from the IA date.
/*>OPC SETVAR TVAR=(OYMD2+2WD)
/* The date on the checks will be: 02/05/09
```

© Copyright IBM Corporation 2015

The SETVAR directive

Two arithmetic operators are supported for SETVAR: addition (+) and subtraction (-)

TT1 - Date arithmetic keywords are as follows:

- WD: workdays, as defined for the calendar that is used by the occurrence
- CD: calendar days
- WK: weeks, converted to days before the arithmetic is performed
- MO: months
- YR: years

TT1 - Time arithmetic keywords are as follows:

- HH: Hours
- MM: Minutes
- SS: Seconds

TT2 - Valid only for time-related variables:

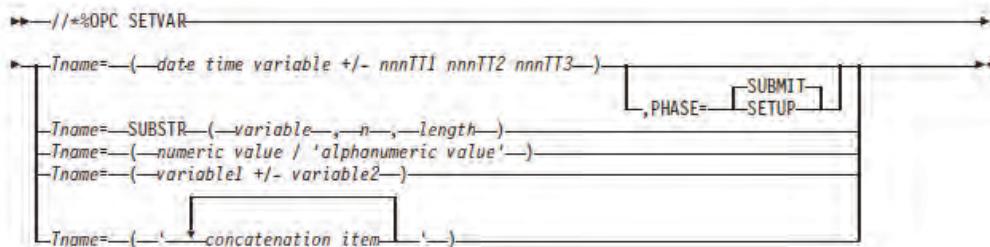
- MM: Minutes
- SS: Seconds

TT3 - Valid only for time-related variables:

- SS: Seconds

The SETVAR directive Syntax

Syntax



You can use the format *nnnTT1 nnnTT2 nnnTT3* only for time-related variables, to add or subtract hours, minutes, and seconds to or from a given time. Specify this triple format only if you want to specify hours, minutes, and seconds.

Using duplicated types, as for example in 6HH, 5MM, 7MM, is not allowed.

© Copyright IBM Corporation 2015

The SETVAR directive Syntax

The SETVAR format that is shown above is SETVAR TVAR=(OYMD2+2WD), with the following elements.

TVAR is the name of the temporary variable you create in the job. The name must begin with a T

- In the example OYMD2 is a supplied date variable that resolves to the occurrence input arrival date in the format YY/MM/DD.
- 2WD Specifies two workdays.
- OYMD2+2WD is an expression that tells the scheduler to add two work days to the occurrence input arrival date.

SETVAR Can also be used with Dynamic format variables and SETFORM as follows:

```
// *%OPC SCAN
// *%OPC SETFORM CDATE=(ACCURATE DATE CCYY MM DD)
// *%OPC SETVAR TDATE=(CDATE + 1CD)
```

If the occurrence input arrival date is 15/02/28, the expressions are substituted as follows:

```
CDATE = 'ACCURATE DATE 2015 02 28'
TDATE = 'ACCURATE DATE 2015 03 01'
```

SETVAR: can also be used to set a value, do normal addition or subtraction on numbers, substring a variable or concatenate values. CD: calendar days

Substring example that uses a predefined variable

```
/*%OPC SETFORM OCDATE=(YYMMDD)
/*%OPC SETVAR TVAR1=(&OCDATE')
/*%OPC SETVAR TVAR2=SUBSTR(&TVAR1,3,2)
/*%OPC SETVAR TVAR3=(OCDATE + 1MO)
/*%OPC SETVAR TVAR4=SUBSTR(&TVAR3,3,2)
```

If the occurrence input arrival date is 14/05/22, the expressions are substituted as follows:

```
TVAR1 = 140522
TVAR2 = 05
TVAR3 = 140622
TVAR4 = 06
```

In fact the SUBSTR parameter identifies a substring of TVAR1 and TVAR3 values, starting from position 3 for a length of 2 characters. According to the format set by the SETFORM directive, it identifies the MM part of the date value.

An example of arithmetic with temporary variables

```
/*%OPC SETVAR TX=(1)
/*%OPC SETVAR TY=(2)
/*%OPC SETVAR TZ=(&TX+&TY)
TZ is a temporary variable set to the result of the arithmetic addition.
An example of concatenating temporary variables
/*%OPC SETVAR T001=('STRING1')
/*%OPC SETVAR T002=('STRING2')
/*%OPC SETVAR T003='&T001 &T002 CONCATENATED STRINGS'
```

T003 is a temporary variable set to the following value: STRING1 STRING2 CONCATENATED STRINGS

Search-related directives

Two search-related directives

SEARCH

Syntax:

`/*%OPC SEARCH NAME(table name,..., APPL|NOAPPL, GLOBAL|NOGLOBAL)`

TABLE

Syntax:

`/*%OPC TABLE NAME=(table name)`

© Copyright IBM Corporation 2015

Search-related directives

The SEARCH directive defines the variable tables that are searched when assigning a value to a variable. Use the SEARCH directive to specify the variable tables you want searched. You can specify up 16 tables, including the application and global tables. The default search order is as follows until the search finds the variable:

1. Variable tables that are specified by table name, in the order specified
2. Application variable table, if it exists
3. Global variable table



Note: A SEARCH directive cannot contain any Tivoli Workload Scheduler for z/OS variables.

The TABLE Directive defines a variable table that is searched before the variable tables in any existing concatenation when resolving JCL variables. You use the TABLE directive to include a variable table at the front of any existing table concatenation. You must have created the table specified by table name by using the OPC JCL Variable Tables panel. The table name that is specified can be the name of a table or an ampersand (&) or percent sign (%) variable. This variable follows the standard scheduler format and can be operator-prompted or not.

You can use more than one TABLE directive in the operation JCL. When each table statement is found, the table name it specifies is added to the front of the table search list. You can concatenate up to 16 tables in this way, including application and global variable tables.

Searching for variables examples

SEARCH example:

```
// *%OPC SEARCH NAME=(GLOBAL, TABLE1, NOAPPL)
```

TABLE example:

VARABC and DATASET2 are both defined in TABLE1 and in TABLE2

```
// *%OPC SCAN
// *%OPC SEARCH NAME=(TABLE2)
// DDNAME1 DD DSN=&VARABC..FINANCE,DISP=SHR
// *%OPC TABLE NAME=(TABLE1)
// DDNAME2 DD DSN=&VARABC.&DATASET2.,DISP=SHR
```

TABLE1 is searched first, and then **TABLE2** only for the variables used following the **TABLE** statement

© Copyright IBM Corporation 2015

Searching for variables examples

Variables and TABLE statements process sequentially. After a variable is assigned a value, all instances of the variable are set to that value. Later table concatenations do not affect the variable value.

Lesson 3 JCL Variables

Lesson 3 JCL variables

- Usable in jobs that are running on any operating platform
- Automatic substitutions are performed before job submission
- Possibilities:
 - User-created
 - Tivoli Workload Scheduler for z/OS-supplied (built-in)
 - Temporary
- If user-created, stored in tables in the variables database
- Support for the following items:
 - Interdependence of values, or dependent variables
 - Validation of user-supplied values, or operator-prompted variables

© Copyright IBM Corporation 2015

In this lesson, you learn how to use JCL variables to do automated value substitutions in the JCL of a job. After completing this lesson, you should be able to use JCL variables to do automated value substitutions in the JCL of a job.

You can use variables in job statements for any operating platform that are directly connected to the controller, including z/OS systems and supported tracker agents. Tivoli Workload Scheduler for z/OS does not support variable substitution for jobs that run on fault-tolerant agents in the distributed environment. For jobs running on z/OS systems, you can use JCL variables as follows:

- Executable job statements.
- Comment statements.
- In-stream data.
- In-stream procedures.

You can test and verify variable substitution at job setup with a user-tailored PIF program. Usually substitution takes place at job submit time as defined by the user (the default). You create and save user-defined variables in the Tivoli Workload Scheduler for z/OS variables database.

You can create temporary variables by using an arithmetic expression on the date-pertinent or time-pertinent variables. You use the SETVAR directive to create temporary variables.

Some jobs require information that is not available until just before job submission. These jobs require manual setup. You can create operator-prompted or dependent variables, or both, for these jobs. These types of variables can reduce manual data entry errors.



Note: If you give a variable a value of b (blank), the variable substitution does not occur if the variable definition states that no value is required. These variables can then undergo the usual operating system substitution after tailoring. If a variable is set to \neg (logical not) or has it as a default setting, the variable is deleted from the line.

Using variables in a job

- Specify the SCAN directive before the first variable in the job.
Also specify the OPCOPTS VARSUB(SCAN) initialization statement
- A JCL variable can begin with any of the three following symbols:
 - & ampersand
 - % percent
 - ? question mark
- You can assign a value to an ampersand (&) or percent (%) variable that is itself another variable
- You can terminate a JCL variable with any of the following types of symbols:
 - A period
 - A blank
 - One of 13 other non-alphabetic characters

© Copyright IBM Corporation 2015

Using variables in a job

Set the VARSUB keyword of the OPCOPTS statement to YES. Variable substitution occurs from the beginning of the job for all operations that are defined on setup or computer workstations.

Set the VARSUB keyword of the OPCOPTS statement to SCAN (the default) and specify the directive ///*%OPC SCAN IN your job. Substitution in the job starts if the SCAN directive is found.

Use the SCAN directive before specifying any variables in the job. You can prevent variable substitution in a job by using the NOSCAN value for the ACTION keyword in a BEGIN and END directives pair. You denote the start of a variable with one of the following symbols:

- & (ampersand)
- % (percent sign)
- ? (question mark)

The symbol determines how Tivoli Workload Scheduler for z/OS carries out the variable substitution. Denote the end of a variable with a period, a blank, or one of 13 other symbols.

Ampersand (&) variables are substituted from left to right within the line. Ampersand variables correspond to the symbolic variables in z/OS JCL procedures and behave the same. If an ampersand variable is immediately followed by a percent sign (%) variable and there is no intervening termination character, a compound variable is formed. Any string that begins with double ampersands (&&) is not substituted.

Sometimes the JCL contains symbolic parameters in the format &symbolic and you might not want the scheduler to attempt resolution. In this case, you can define the variable in a scheduler JCL variable table. Define the variable with a blank default value and specify that a value is not required.

Percent (%) variables can be used to form simple variables and compound variables.

Question mark (?) variables are tabular. You can specify the column on the JCL statement that the variable value begins when substituted.

Variable concatenation

- Simple
 - One-to-one substitution
- Compound
 - Variables concatenated to each other
 - Variable concatenations that form new variables

© Copyright IBM Corporation 2015

Variable concatenation

Simple variables are preceded by a percent sign (%) and ended by a period or any termination character other than the percent sign. Values are assigned to the variable in one pass. These variables do not use any concatenation.

Using JCL substitution, you can form compound variables. A compound variable consists of a concatenation of variables as follows:

- A variable (of any type) followed by a percent variable with no intervening periods or other termination symbols.
- A question mark variable followed by an ampersand variable with no intervening periods or other termination symbols.

These are two or more simple variables that are concatenated to dynamically create new variables.

Using compound variables

Percent (%) variables are substituted from right to left:

```
//STEPLIB DD DSN=MY.%DATA%SET
```

On first pass:

```
//STEPLIB DD DSN=MY.%DATALIB
```

On second pass:

```
//STEPLIB DD DSN=MY.MAINFILE
```

Example of a variable table

Variable name	Default value
SET	LIB
DATALIB	MAINFILE

© Copyright IBM Corporation 2015

Using compound variables

This example shows two simple variables that are defined in the variable table. The variables that are used in the job statement have a percent sign prefix. Therefore, the two variables that are concatenated to form the compound variable are resolved, starting with the right most variable. The percent symbol (%) indicates a right-to-left substitution sequence.

Because two variables form the compound variable, Tivoli Workload Scheduler for z/OS resolves them in two passes: the %SET is resolved in the first pass, and the value is placed in the column where the variable was specified. A new variable called %DATALIB is created, which Tivoli Workload Scheduler for z/OS resolves on the second pass.

Using tabular variables

- Question mark (?) variables are tabular
- User can specify a column to place the default value in
- Specification can be placed in one of two places:
 - The variable definition in the database
 - In the job when the variable is used
- Usage:
 - ?VARNAMe uses column specified in database
 - ?nnVARNAMe uses column number specified by *nn*

© Copyright IBM Corporation 2015

Using tabular variables

Question mark variables are tabular. You can specify in the column on the JCL statement that the variable value begins when substituted.

The position where the value is placed can be defined in the Tivoli Workload Scheduler for z/OS database. You specify the position when the variable is defined or in the job where the variable is used. You might use these variables primarily within in-stream data, although they can be used anywhere in your job.

Operator-prompted variables

- Values that users supply before a job is submitted
- User can view a customized prompt that requests input of a valid value
- The user's input can be validated based on a criterion from the following list
 - ALPHA
 - NUM
 - ENUM
 - HEX
 - BIT
 - PICT
 - NAME
 - DSNAME
 - RANGE
 - LIST

© Copyright IBM Corporation 2015

Operator-prompted variables

When variable values are known only just before job submission, you can use operator-prompted variables to validate operator entries. Tivoli Workload Scheduler can check input that is entered by the operator at job setup time or by a substitution exit routine and validate input based on criteria that you specify. See *Managing the Workload* for details on the criteria.

Tivoli Workload Scheduler for z/OS supplied variables

- Occurrence-related variables
- Operation-related variables
- Date-related variables
- Dynamic-format variables

© Copyright IBM Corporation 2015

Tivoli Workload Scheduler for z/OS supplied variables

Tivoli Workload Scheduler for z/OS supplies variables that you can use in your business. These variable names are reserved by the scheduler. The scheduler does not read variable definitions for these variables from a user-defined variable table. The variables are split into four groups as follows:

- Occurrence-pertinent variables
- Operation-pertinent variables
- Date-pertinent variables
- Dynamic-pertinent variables

Examples:

Variable name: OADID

Length (in bytes): ADID

Description: Application ID

Variable name: OCALID

Length in bytes: 16

Description: Calendar name

For a complete list of standard variables see the Managing the Workload manual .

User-created variable tables

You can specify variable tables in the following locations:

- Period definitions
- Run cycles in applications
- The MODIFY CURRENT PLAN panel
- The MODIFY LONG-TERM PLAN panel
- Jobs
- Tivoli Workload Scheduler for z/OS initialization statement

© Copyright IBM Corporation 2015

User-created variable tables

You can control where Tivoli Workload Scheduler for z/OS looks to find a value for the variables that you use in a job. You can specify the name of your variable table in places as follows:

- Period definitions.
- Run cycles in applications.
- The MODIFY CURRENT PLAN panel.
- The MODIFY LONG-TERM PLAN panel.
- Jobs, by using the TABLE and SEARCH directives.
- A Tivoli Workload Scheduler for z/OS initialization statement (global variable table).

Defining variables in tables



The screenshot shows a terminal window with the command EQQJVVML running. The title bar says "MODIFYING VARIABLES IN A TABLE". The message area says "Row 1 of 13" and "Scroll ==> CSR". Below that is a help message about row commands. The variable table is titled "TEAM#VAR1" and owned by "STUDENT". The table description is "Student jobcard vars". The table has columns: Row, Variable, Subst., Setup, Val, Default, cmd, Name, Exit, Val req, Value. The data rows are:

Row	Variable	Subst.	Setup	Val	Default	cmd	Name	Exit	Val req	Value
1	RCCT	N	N	9999						
2	ACCTNUM	N	Y	99999999999						
3	IPADDR	N	Y	255.255.255.255						
4	IPNAME	P	Y	MVSC11						
5	MYID	N	Y	INGC103						
6	MYIP	N	N	0.0.0.0						
7	MYPDM	N	N							
8	TEAM	N	Y	D						
9	TERMID	N	N	TMHRA						
10	TEAMIDX	P	N							

© Copyright IBM Corporation 2015

Defining variables in tables

Use Tivoli Workload Scheduler for z/OS ISPF option 1.9.2 to define variables in tables. Subst.Exit shows the name of an exit routine that can set a variable, validate a variable, or both.

Setup shows how variable substitution is handled:

- N: No interaction. The variable is substituted at submit. Default.
- P: An interaction with the operator occurs at job setup.
- Y: No interaction. The variable is substituted at submit if it is not already set at job setup.

Val Req indicates whether the variable is assigned a blank value:

- N: Not required. The variable can be assigned a blank value. Default.
- Y: Value required. The variable cannot be assigned a blank value.

Default Value shows the value that the variable has if another value is not explicitly assigned.

Assigning variable tables to applications

```
EQQAMRPL ----- RUN CYCLES ----- Row 1 to 1 of 1
Command ==> _ Scroll ==> PAGE

Enter/Change data in the rows, and/or enter any of the following
row commands:
I(nn) - Insert, R(nn),RR(nn) - Repeat, D(nn),DD - Delete
S - Specify run days/Modify rule

Application : SM40#MAX3 Appl using TEAM#DSN
  Name of   Input Deadline F day In Out of
Row period/rule HH.MM day HH.MM Type rule effect Effect
cmd Text
  RULE3 15.14 00 23.00 R 4 07/11/11 71/12/31 TEAM#
  TEAM# Run Cycle using Every Option
```

Variable table assigned to the run cycle definition

© Copyright IBM Corporation 2015

Assigning variable tables to applications

Use the run cycle definition of an application description to assign variable tables to the application.

Scanning order for variable resolution

Variable tables are scanned in this order until the end or the variable is found

- Job-specific tables
 - Name in the TABLE directive
 - Names in the SEARCH directive
- Application-related tables
 - Name on the MCP panel
 - Name on the MODIFY LTP panel
 - Name on the run cycle
 - Name associated with the period definition
- Global table: Value specified in the OPCOPTS GTABLE initialization statement
OPCOPTS GTABLE(table ID|GLOBAL)

© Copyright IBM Corporation 2015

Scanning order for variable resolution

You can refer to variable tables in several places. A job can refer to more than one table. You can refer to tables by using directives in the job, the run cycle that is used to schedule the application, or a period that is used by the run cycle.

This slide shows the search order for the variable tables. The tables are searched in this order until the variable is found. If a table is not found, the search proceeds to the next table. If a value for the variable cannot be found, then substitution does not take place. The job fails with none of the variables substituted. The error code for failed variable substitution is OJCV.

Summary

- List the main goal of automated job tailoring
- Tailor the job before submitting it with JCL automation directives
- Create and use Tivoli Workload Scheduler for z/OS JCL variables

© Copyright IBM Corporation 2015

Summary

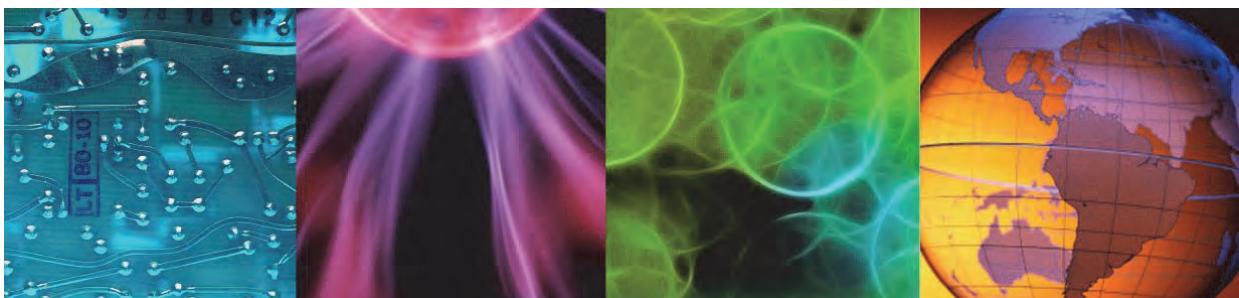


10 Automatic recovery

IBM Tivoli Workload Scheduler for z/OS 9.2.0



10 Automatic recovery



© Copyright IBM Corporation 2015

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

In this unit you learn about the Tivoli Workload Scheduler for z/OS automatic recovery function and how to code the control statement for it.

References

- SC32-1262 *IBM Tivoli Workload Scheduler for z/OS Managing the Workload*
- SC32-1265 *IBM Tivoli Workload Scheduler for z/OS Customization and Tuning*

Objectives

- Describe the automatic recovery function
- List the types of automatic recovery actions
- Code automatic recovery statements

© Copyright IBM Corporation 2015

Objectives

At three o'clock in the morning you put a call out to support to fix a failed job and the solution is to restart from failed step. The time lost between failure and fix can be removed by using Automatic recovery. This unit shows how this feature works and how it can allow known recoveries to be built into the jobs. You use this function in a following hands-on exercise.

Lesson 1 Automatic recovery

Lesson 1 Automatic recovery

- Automatic recovery for FAILED jobs and started tasks
- Automatic recovery of jobs and started tasks for system failures within the sysplex
 - Requires XCF
 - OR
 - Manual recovery
- Specifies the recovery criteria as part of the JCL
 - Requires special control statements in the JCL
- Works with restart and cleanup
 - Data set cleanup is performed

© Copyright IBM Corporation 2015

In this lesson, you learn about the automatic recovery feature of Tivoli Workload Scheduler for z/OS. After completing this lesson, you should be able to describe and implement the automatic recovery feature of Tivoli Workload Scheduler for z/OS.

Tivoli Workload Scheduler for z/OS supports the automatic recovery of failed jobs or started tasks. You specify the recovery criteria as part of the JCL for the operation in the form of special control statements. If a failing job operation has a cleanup action defined and contains automatic recovery statements, the cleanup action is performed before automatic recovery. For automatic recovery, you can use only the Immediate or None cleanup types.

The automatic recovery function takes over when a job or started task ends in error. Then, the following information is available:

- The error code for the operation. The error code can be as follows:
 - The abend code of an abending step
 - The return code of the last step
 - An error code set by Tivoli Workload Scheduler for z/OS, such as JCLI, CCUN, JCL, CLNO, CLNA, CLNC, CAN, PCAN, CLNP, OFxx, or OSxx
 - An error code set by the job completion checker

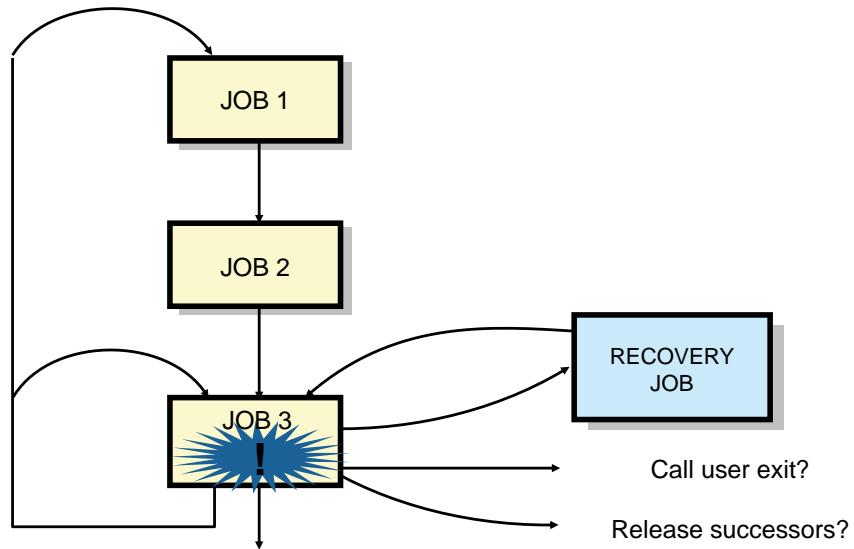
- The name of the abending step, if the error is associated with a step
- Step completion codes and step names for all steps that ran. The step completion code is either an abend code or a return code.

If the error occurs in the initialization phase or in the completion phase of the job or started task, no step information is available. Statements that specify recovery actions for certain steps are not applicable to such errors.



Note: Automatic recovery is not applicable for error codes, such as OSUP. These error codes are for jobs that have not yet reached the job queue.

Automatic recovery diagram



© Copyright IBM Corporation 2015

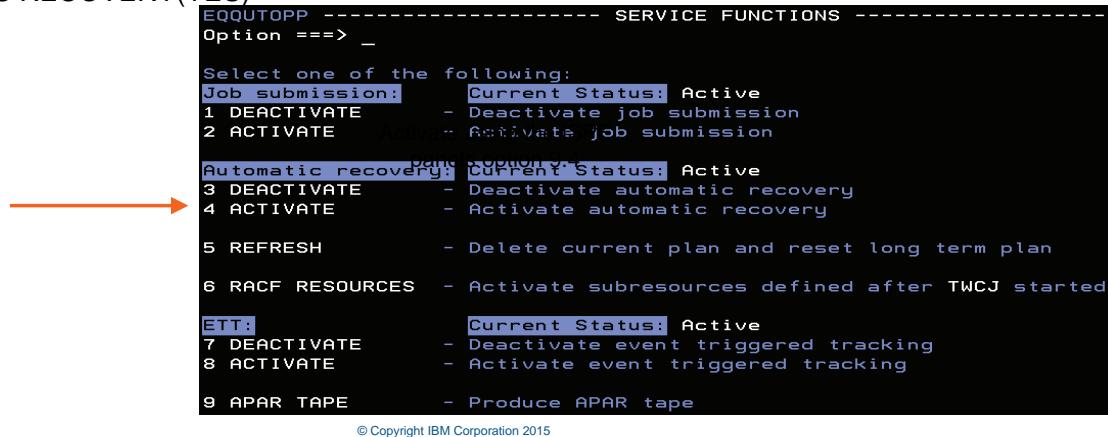
Automatic recovery diagram

If activated, the automatic recovery function in the scheduler attempts to take the appropriate actions when a job fails. This feature of Tivoli Workload Scheduler for z/OS provides automated recovery without manual intervention.

You can switch the automatic recovery function on or off by using the Tivoli Workload Scheduler for z/OS Service Functions panel, ISPF options 9.3 and 9.4. As part of the JCL of a job, you can define a RECOVER statement that specifies one or more automatic recovery actions. The actions can include rerunning the job, starting a user-written exit, or starting a successor application.

Implementing automatic job recovery

- Specify recovery criteria and recovery actions in job JCL
 - Control statements in JCL
//%OPC RECOVER
- Use controller initialization statements
 - OPCOPTS RECOVERY(YES)
 - AROPTS



```
EQQUTOPP ----- SERVICE FUNCTIONS -----
Option ==> _

Select one of the following:
Job submission: Current Status: Active
1 DEACTIVATE - Deactivate job submission
2 ACTIVATE Active Activate job submission

Automatic recovery: Current Status: Active
3 DEACTIVATE - Deactivate automatic recovery
4 ACTIVATE - Activate automatic recovery

5 REFRESH - Delete current plan and reset long term plan
6 RACF RESOURCES - Activate subresources defined after TWCJ started

ETT: Current Status: Active
7 DEACTIVATE - Deactivate event triggered tracking
8 ACTIVATE - Activate event triggered tracking

9 APAR TAPE - Produce APAR tape

© Copyright IBM Corporation 2015
```

Implementing automatic job recovery

You must activate automatic job recovery by using the ISPF dialogs, option 9.4. For the activation to succeed, the OPCOPTS RECOVERY(YES) initialization statement parameter must be specified for the controller.

You can also provide automatic recovery options in the AROPTS initialization statement when RECOVERY(YES) is specified. See the IBM Tivoli Workload Scheduler for z/OS Customization and Tuning Guide for details on the AROPTS initialization statement. You can specify run time options for automatic recovery, such as start and end times for the feature. You can also specify error codes that are excluded from automatic recovery processing.

You can set up automatic recovery procedures for many predictable failures by using //%OPC RECOVER directives in your JCL. The basic contents of the recovery directives in the JCL are as follows:

- Matching criteria that identifies a failure for which there is a recovery.
- The actions to be taken.

Recovery actions

- Restart at failed operation
- Restart at another operation, with or without JCL changes
- Add another application into CP
- Release a successor dependency
- Restart at failed or another step, with or without JCL changes
- Leave in error

© Copyright IBM Corporation 2015

Recovery actions

In the automatic recovery statements in your JCL, you specify recovery actions for the possible failures. The JCL can contain multiple recovery actions for different or recursive failures. Each recovery statement can be used only once. Recovery actions can include JCL and job execution modifications before restarting. Some of these actions are as follows:

- Deleting steps
- Including procedures
- Restarting at a step other than the first step

The recovery parameters are as follows:

- **RESTART:** Specifies whether the occurrence is restarted.
- **RESJOB:** Specifies the name of the job or started task from where the occurrence is rerun.
- **ADDAPPL:** Specifies an application or a list of applications that are added as occurrences in the current plan.
- **RELSUCC:** Specifies the application ID of a successor occurrence or a list of IDs.
- **ALTWS:** Specifies the name of an alternative workstation on which to run the operation.
- **ALTJOB:** Specifies an alternative job or started-task name to use when the job is restarted. You use it in a MAS complex to support the restart of a job that has not ended according to JES.

Recovery statements: Conditional criteria

- Can be conditional
- Can contain the following items:

CONDITIONAL CRITERIA

- **ERRSTEP** Step in error
- **JOBCODE** Error code for job
- **STEPCODE** Error code for step
- **TIME** Time selection parameters

© Copyright IBM Corporation 2015

Recovery statements: conditional criteria

The selection parameters in the recovery statement, which are all optional, are as follows:

- **ERRSTEP:** Restricts recovery actions to the steps that are listed.
- **JOBCODE:** Restricts recovery to be valid only for the completion codes or return codes that are specified.
- **STEPCODE:** Specifies that only the step return codes that are listed are valid.
- **TIME:** Restricts recovery actions to specific times.

You can put several combinations of selection criteria into the RECOVER statement. You can, for example, take different actions at different times of the day.

Recovery statements: JCL rebuild and actions

JCL
REBUILD
OPTIONS

- **DELSTEP** Step to delete
- **RESSTEP** Step at which to restart
- **ADDPROC** Call a procedure
- **CALLEXIT** Program name

RECOVERY
ACTIONS

- **ADDAPPL** Add an application into CP
- **RESTART** YES or NO
- **RESJOB** Restart at *jobname x*
- **RELSUCC** Release successor application
- **ALTWS** Alternative workstation
- **ALTJOB** Alternative job or ST name

© Copyright IBM Corporation 2015

Recovery statements: JCL rebuild and actions

JCL rebuild options

In the OPC RECOVER statement, you can include one or more of the following optional parameters to rebuild the JCL before you submit it:

- **DELSTEP:** Specifies a step, a list, or a range of steps to delete from the inline JCL before rerunning the failed operation.
- **RESSTEP:** Specifies the name of the job or started-task step where the operation is to restart.
- **ADDPROC:** Specifies a name or a list of names of JCL procedure library members to be added to the inline JCL before rerunning the failed operation.
- **CALLEXIT:** Specifies the name of a program exit routine that is called before the restart.

Automatic job recovery example JCL

```
//RECOVZA JOB ....  
//**%OPC RECOVER JOBCODE=S*37,ADDPROC=SPACECHG  
//**%OPC RECOVER JOBCODE=(*,16-4095),RESTART=N,  
//%OPC ADDAPPL=REORG ← RECOVERY statement continuation  
//**%OPC RECOVER JOBCODE=12  
//STEP01 EXEC PGM=MYPROG,REGION=256K  
//SYSOUT DD SYSOUT=A  
//TSTIN DD DSN=TI94237.IN.DATA,DISP=SHR  
//TESTWK1 DD UNIT=3380,SPACE=(CYL,(1,1))  
//TESTWK2 DD UNIT=3380,SPACE=(CYL,(1,1))  
//TESTWK3 DD UNIT=3380,SPACE=(CYL,(1,1))  
//TSTOUT DD DSN=TI94237.OUT.DATA,DISP=SHR
```

© Copyright IBM Corporation 2015

Automatic job recovery example JCL

The first RECOVER statement specifies that, for space problems in any of the steps, both of these actions occur:

- The member SPACECHG is added.
- The failed job is restarted (that is, rerun the failed operation).

The second RECOVER statement specifies that the failed job is not restarted if both of the following conditions exist:

- Any other error code not in the AROPTS EXCLUDECC list
- A return code of 16 or higher

However, the REORG application starts.

The third RECOVER statement specifies that, for return code 12, the failed job restarts. The preceding actions apply to the time range specified by the STARTTIME and ENDTIME parameters of the AROPTS automatic recovery initialization statement.

If the job fails a subsequent time, it receives the ended-in-error status and remains on the ended-in-error list. The scan for recovery statements can be repeated later.

The ARC row command

```
EQQMEE1L ----- HANDLING OPERATIONS ENDED IN ERROR(left part) Row 1 to 1 of 1
Command ==> _                                         Scroll ==> PAGE

Scroll right or enter the SUPPRESS command to suppress full row command
information, enter the HIST command to select operation history list
or enter any of the row commands below:

OPERATION RELATED COMMANDS :
I query information, O browse operator instructions, J edit JCL,
C complete, MH manual hold, MR manual release, SJR simple job restart,
RC restart and cleanup, FSR Fast path SR, FJR Fast path JR, FSC Fast path SC
L Browse joblog, RI Recovery info

OCCURRENCE RELATED COMMANDS:
RER rerun, ARC attempt automatic recovery, WOC reset to waiting, CMP complete,
MOD modify, DEL delete, RG Remove from Group, DG Delete Group, or
CG Complete Group

LAYOUT ID          ==> OPCESA_     Change to switch layout id

Cmd Ended      time Application      ws   no. Jobname Errc
arc 11/10/27 13.16 U#CRITA        U#CP  10 U#CRITJ2 0008
```

© Copyright IBM Corporation 2015

The ARC row command

A job or started task that ends in error is automatically scanned for a RECOVER statement that corresponds to the reported error. You can start this recovery scan manually from the Handling Operations Ended in Error Panel in the MODIFY CURRENT PLAN panel. Selecting an operation with the row command **ARC** starts the scan.

If you start recovery from the Modifying the Current Plan panel, time restrictions on the RECOVER statements are overridden. The recovery occurs immediately, regardless of the TIME parameter that is specified on the RECOVER statement. The RECOVER statement that is used matches the original error condition of the operation, the same as if the scan were started automatically.

Automatic recovery messages and comments

```
EQQMJCLE ----- EDITING JCL FOR A COMPUTER OPERATION -----
Command ==> _                                         Scroll ==> CSR

Edit JCL below and press END to finish or CANCEL to reject:

Application      : ARTEST1                      Test Automatic Recovery
Operation        : UHCP 1
Status of operation : Ended in error          S806
Jobname          : ARJOB1                       JCL last updated by: INGC103

***** ***** Top of Data *****
000001 //ARJOB1 JOB (999), 'INGC103', CLASS=A, MSGCLASS=H, NOTIFY=&SYSUID
000002 //*>OPC RECOVER JOBCODE=S806,RESTART=N
000003 /* OPC MSG:
000004 /* OPC MSG: I *** R E C O V E R Y   A C T I O N S   T A K E N   ***
000005 //STEP01 EXEC PGM=MYPROG,REGION=256K
000006 //SYSOUT DD SYSOUT=A
000007 //TSTIN DD DSN=INGC103.IN.DATA,DISP=SHR
000008 //TESTWK1 DD UNIT=3380,SPACE=(CYL,(1,1))
000009 //TESTWK2 DD UNIT=3380,SPACE=(CYL,(1,1))
000010 //TESTWK3 DD UNIT=3380,SPACE=(CYL,(1,1))
000011 //TSTOUT DD DSN=INGC103.OUT.DATA,DISP=SHR
```

Edit the JCL (J row command) from the MODIFYING THE CURRENT PLAN panel after running the ARC command
[View the results](#)

© Copyright IBM Corporation 2015

Automatic recovery messages and comments

You can also specify a RESTART=N parameter on a RECOVER statement, which causes a RECOVER statement scan to occur. With this action, you see the effect that starting the RECOVER statement has without causing the operation to be resubmitted. The JCL created in this way is the JCL that the scheduler uses if the operation reruns. To return to the original JCL, you must manually edit the JCL to undo the effects of the recovery statement. This editing includes amending the RECOVER statement because it automatically changes to a comment statement.

If you want the original job from EQQJBLIB, you can delete all the job statements and press F3 (End) to save. This action forces Tivoli Workload Scheduler for z/OS to read a fresh copy of the JCL from the original EQQJBLIB library.

The automatic recovery function updates the JCL of the failing operation with information in the form of message statements about the recovery action taken. Also, if an error occurs during the automatic recovery function, the scheduler uses message statements to insert an error description in the JCL file that is saved on the JS data set. This example shows the format of the message statement:

```
/* OPC message-text
```

Logging and error reporting

Automatic recovery actions are logged in three places:

- On the job-tracking log, for Tivoli Workload Scheduler for z/OS restart purposes
- On the Tivoli Workload Scheduler for z/OS message log
- In the Tivoli Workload Scheduler for z/OS JS data set
 - In the JCL record of the failed job
 - OR
 - In the started task in the form of message statements

© Copyright IBM Corporation 2015

Logging and error reporting

The slide shows that the successful start of recovery actions is recorded in three places. Message statements are added to the JCL for you to reconstruct the recovery activities performed.

If an error is detected in a RECOVER statement, the scheduler inserts messages that describe the error into the JCL. The messages identify the incorrect statement and possibly the position within the statement where the error was detected. The letter A in the message indicates the position of the error, as shown in the following example:

```
//*>OPC SCAN
//SAMPLEA JOB (885002,NOBO),SAMPLE,NOTIFY=XMAWS,MSGCLASS=Q,
// CLASS=B,MSGLEVEL=(1,1),PRTY=1
//*%OPC RECOVER JOBCODE=JCL,RESTART=N,ADAPPL=RECOV
//* OPC MSG: A
//* OPC MSG: E INCORRECT PARAMETER
//STEP1 EXEC PGM=IEFBR14
//DD1 DD DSN=XMAWS.NOT.THERE,DISP=(OLD,DELETE,DELETE)
//
```

Summary

- Describe the automatic recovery function
- List the types of automatic recovery actions
- Code automatic recovery statements

© Copyright IBM Corporation 2015

Summary

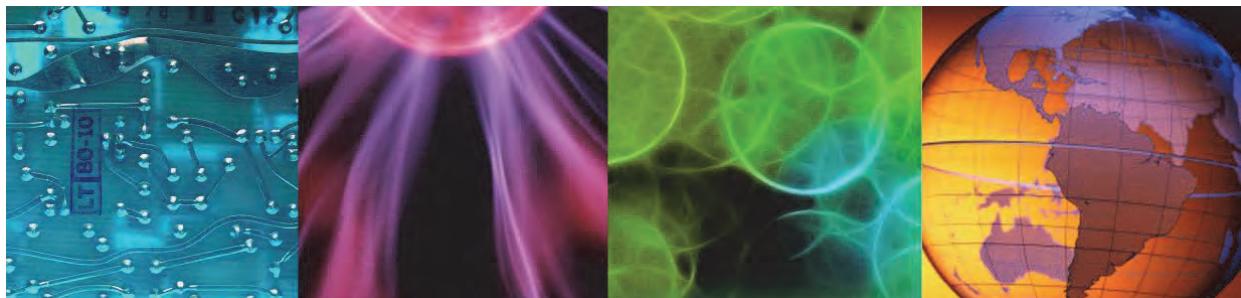


11 Managing unplanned work

IBM Tivoli Workload scheduler for z/OS 9.2.0



11 Managing unplanned work



© Copyright IBM Corporation 2015

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

In this unit, you learn about TSO-authorized Tivoli Workload Scheduler for z/OS commands. You also learn about event-triggered tracking and the data set triggering functions. You can use TSO commands, event-triggered tracking, and data set triggering to handle unplanned work.

References

- SC32-1262 *IBM Tivoli Workload Scheduler for z/OS Managing the Workload*
- SC32-1265 *IBM Tivoli Workload Scheduler for z/OS Customization and Tuning*
- SC32-1266 *IBM Tivoli Workload Scheduler for z/OS Programming Interfaces*

Objectives

- Issue the OPSTAT and SRSTAT commands to handle unplanned work and control workflow
- Handle unplanned work with event-triggered tracking
- Create data set triggering table entries
- Enhance event-triggered tracking with data set triggers

© Copyright IBM Corporation 2015

Objectives

Although Tivoli Workload Scheduler for z/OS, schedules batch workload, not all batch workload can be scheduled. This unit introduces some of the solutions that can be used to bring the unplanned work under the control of Tivoli Workload Scheduler for z/OS. External commands that can alter the plans, event tracking and dataset triggering are explored and hands-on exercises show these solutions in a practical way.

Lesson 1 Managing unplanned work overview

Lesson 1 Managing unplanned work overview

- On-demand or unpredictable operations
- Commands
- Event-triggered tracking
- Data set triggering

© Copyright IBM Corporation 2015

This lesson provides an overview of managing unplanned work that uses Tivoli Workload Scheduler for z/OS. After completing this lesson, you should be able to describe the managing of unplanned work that uses Tivoli Workload Scheduler for z/OS.

Unplanned, on-request, or on-demand - dynamic work forms a large part of the work that is handled by many scheduling departments. Tivoli Workload Scheduler for z/OS is equipped to handle unplanned work.

This unit covers some of the functions and commands that you can use to handle unplanned work:

- Tivoli Workload Scheduler for z/OS commands OPSTAT and SRSTAT
- Event-triggered tracking
- Data set triggering

Lesson 2 OPSTAT and SRSTAT commands

Lesson 2 OPSTAT and SRSTAT commands

TSO authorized commands SYS1.PARMLIB(IKJTSOxx)

- Issue directly from TSO or from a batch job both inside and outside Tivoli Workload Scheduler for z/OS
- Use OPSTAT to set the status of an operation at any workstation except a non-reporting workstation
- Use SRSTAT to change the overriding (global) availability, quantity, or deviation of a special resource

© Copyright IBM Corporation 2015

In this lesson, you learn how to use the OPSTAT and SRSTAT commands. After completing this lesson, you should be able to use the OPSTAT and SRSTAT commands.

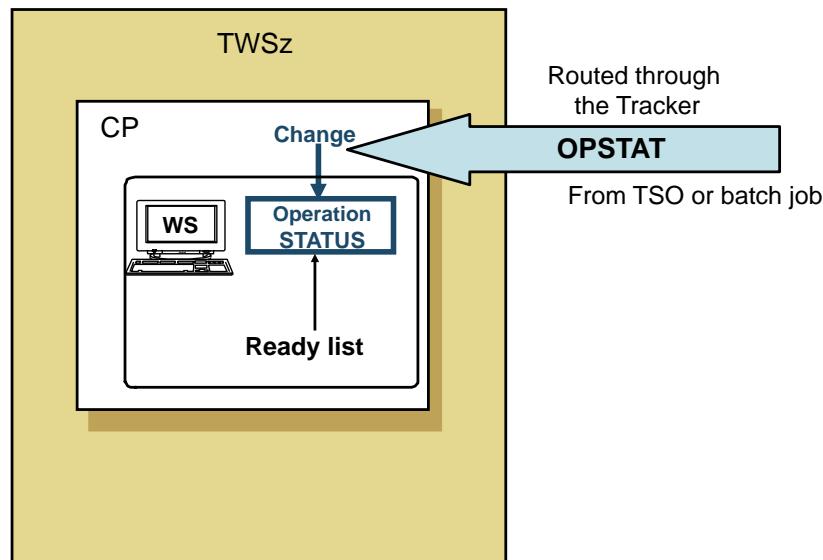
You can use the OPSTAT command to place the starting of an operation into the hands of a user who must control it. By submitting a job with the OPSTAT command, the user can set the status of a manual predecessor operation to complete and release successive operations.

You can use the SRSTAT command to prevent operations from allocating a particular resource. SRSTAT can also be used to add an occurrence to the current plan by using event-triggered tracking.

The OPSTAT command

CP = Current Plan

WS = Workstation



© Copyright IBM Corporation 2015

The OPSTAT command

You can use the OPSTAT (operation status) command to change the status of an operation to one of the following states:

- **C**: completed (probably the most common use)
- **E**: ended in error
- **I**: interrupted
- **S**: started
- **X**: reset the status to its previous logical status
- **Q**: extended status Q set
- **T**: extended status S set

Events that are generated by OPSTAT are matched against operations on the ready list. Events that are received for operations in Waiting (**W**) or Complete (**C**) status are ignored. You cannot change the status of a waiting operation because it has incomplete predecessors. If more than one operation is scheduled for a workstation, you can optionally specify more selection criteria. You can use the ADID, IA, OPNUM, or JOBNAME parameters to identify the specific operation whose status is to change.

Issuing the OPSTAT command

- User needs update authority to the ready list
- To identify the workstation, you must specify the WSNAME parameter
- Direct the command to the tracker, not to the controller
- User must specify enough parameters to avoid ambiguity
- Minimum recommended parameters:
 - ADID(application_description_name)
 - JOBNAME(jobname)
 - STATUS(new_status)
 - SUBSYS(MSTR)
 - TRACE(1)
 - WSNAME(workstation_name)

© Copyright IBM Corporation 2015

Issuing the OPSTAT command

You can run OPSTAT as a TSO command or by using a batch job step that runs program EQQEVPGM. The ID of the user who issues the command must be authorized to update the ready list.

Use minimum parameters and add others if necessary to ensure that the correct operation is updated. If you specify MSTR, the OPSTAT command is directed to all tracker subsystems on the z/OS system where the OPSTAT command is issued. The advantage with MSTR is that the job is not affected by changes in the subsystem name. The issue with MSTR is its blanket approach to issue the command to ALL tracker subsystems as it could affect the batch in the wrong Controller. Use full parameter names, not abbreviations, to prevent the command from failing because of ambiguity.

OPSTAT example

```
/*  
/* THIS JCL IS USED TO DEMONSTRATE USE OF THE OPSTAT COMMAND  
/*  
//STEP1 EXEC PGM=EQQEVPGM REGION=512K  
//EQQMLIB DD DSN=OPC.EQQMLIB, DISP=SHR  
//EQQMLOG DD SYSOUT=*  
//SYSIN DD *  
    OPSTAT WSNAME(GAUT) STATUS(C) JOBNAME(EA73NJOB) ADID(MYAPPL)  
    SUBSYS(MSTR) TRACE(1)  
  
/*
```

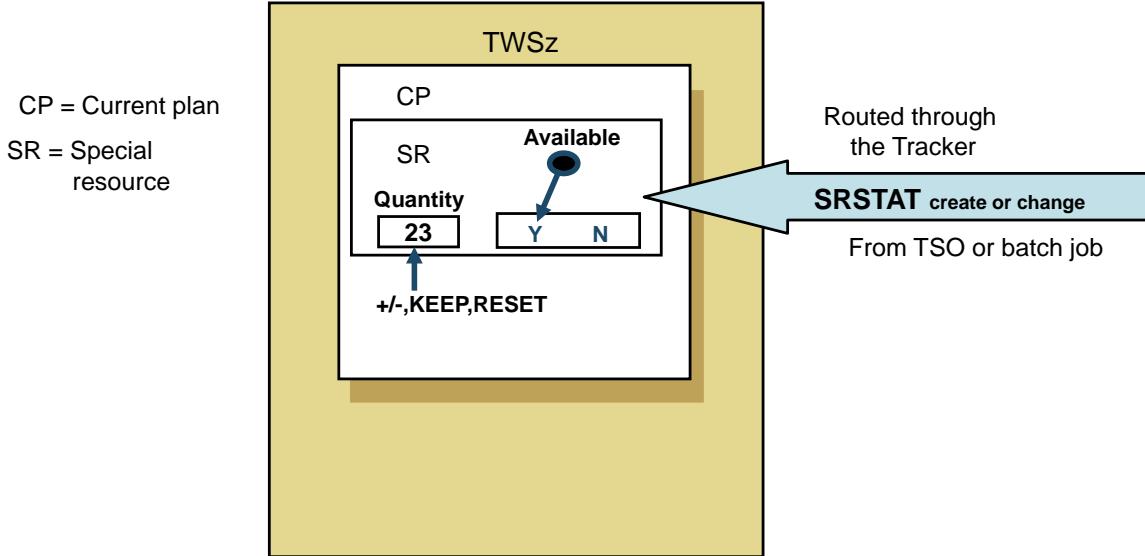
© Copyright IBM Corporation 2015

OPSTAT example

This example shows the required DD statements for the program you run when using a batch job. The Tivoli Workload Scheduler for z/OS program that you run is EQQEVPGM. The example sets the status of job EA73NJOB in the MYAPPL application on workstation GAUT to Complete.

The command can span multiple lines without continuation characters. For more information about the OPSTAT command and the parameters see the *IBM Tivoli Workload Scheduler: Programming Interfaces* manual

The SRSTAT command



© Copyright IBM Corporation 2015

The SRSTAT command

You can use the SRSTAT (special resource status) command to change the overriding (global) availability, amount, and deviation of special resources. As mentioned in [Unit 8, “Special resources” SRSTAT LIFESPAN parameter](#), you can also use the SRSTAT LIFESPAN parameter to set the life span time of a special resource.

You can issue the SRSTAT command directly from TSO or within a batch job. The batch job can be in a Tivoli Workload Scheduler for z/OS application or one submitted outside the control of Tivoli Workload Scheduler for z/OS.

You can include the SRSTAT command in a job step that creates a data set. The data set can also be a special resource and can be made available for other jobs to use. The other jobs can share the special resource and not wait until the original job is completed.

Jobs and started tasks that are running finish regardless of the status of the special resources.

Issuing the SRSTAT command

- User needs update authority to the named resource
- The command is to be directed to the tracker, not to the controller
- Minimum recommended parameters:
 - AVAIL(resource_availability)
 - SUBSYS(MSTR)
 - TRACE(1)

© Copyright IBM Corporation 2015

Issuing the SRSTAT command

The issuer of the SRSTAT command must be authorized to update the named resource. If you specify **MSTR**, the SRSTAT command is directed to all tracker subsystems on the z/OS system where the command is issued. Use the full parameter name, not abbreviations, to prevent the command from failing because of ambiguity.

SRSTAT example

ISPF Option 6

```

Menu List Mode Functions Utilities Help
ISRTSO ISPF Command Shell
Enter TSO or Workstation commands below:
==> SRSTAT 'SPECIAL RESOURCE TEAM1' SUBSYS(MSTR) AVAIL(YES) CRE(YES)

```

- **DYNAMICADD** keyword of the RESOPTS initialization statement must be set to **YES** or **EVENT** to create a new special resource by issuing SRSTAT
- **CRE(YES)** is the default
- Created in the current plan, **not** in the database
- Default quantity is 1

© Copyright IBM Corporation 2015

SRSTAT example

This example shows **SRSTAT** being run as a TSO command from ISPF option **6**. Tivoli Workload Scheduler for z/OS adds the resource to the current plan of the receiving subsystem if the resource is not already in the database. It uses the following values:

- **Text:** Blank.
- **Specres group ID:** Blank.
- **Hiperbatch:** No.
- Used for Control.
- **On error:** Blank. If an error occurs, Tivoli Workload Scheduler for z/OS uses the value that is specified in the operation details. If the value in the operation details is also blank, the value of the ONERROR keyword of RESOPTS is used.
- **Overriding availability, quantity, and deviation:** The value that is specified by SRSTAT, or blank.
- **Default quantity:** 1. The default quantity is automatically increased if contention occurs.
- **Default availability:** Yes.
- **Intervals:** No intervals are created.
- **Workstations:** * (All workstations can allocate the resource).

Lesson 3 Event-triggered tracking

Lesson 3 Event-triggered tracking

- Is usable for tracking work submitted outside of the Tivoli Workload Scheduler for z/OS environment
- Uses job-start or special resource events as triggers
- Adds application occurrences to the current plan:
 - Real applications
 - Dummy applications
- ETT values usable for automatic variable substitution in JCL
- Variables usable for passing event-related information to the triggered job

© Copyright IBM Corporation 2015

In this lesson, you learn how to implement and use event-triggered tracking. After completing this lesson, you should be able to implement and use event-triggered tracking.

The ETT function helps to improve the handling of unplanned work. Some batch work, by its nature, cannot be planned into a timed schedule. It depends on other work being done outside of Tivoli Workload Scheduler for z/OS, such as the following examples:

- The receipt or preparation of a data set or other file.
- Work that is submitted from another system that is not part of the z/OS complex.
- The finance department always running its own updates.

Whatever the reasons for handling some of the work outside of the scheduler, ETT can control it. The unplanned work is set up as an application description in the scheduler. A triggering event then adds it to the current plan. The event can be a job-start event or a special resource that becomes available.

JCL variables are used for automatic variable substitution in the JCL of a job. Examples are JCL variables associated with the triggering event, such as job name or the data set name that trigger.

Event types

- Triggering event type J for job
- Triggering event type R for special resource
- Special resource events, which are generated as follows:
 - The SRSTAT command
 - Data set triggers that occur when a data set is closed
- Job-start events, which are generated by jobs started outside of the current plan

© Copyright IBM Corporation 2015

Event types

Resource events and job events as follow are the two types of events that can act as triggers:

- **R (resource events).** Tivoli Workload Scheduler for z/OS generates a resource event when the status of a named special resource is set to Available. The resource event can occur when a user issues the SRSTAT command. It can also occur when the scheduler issues its own SRSTAT command internally to satisfy a data set trigger.
- **J (job events).** Tivoli Workload Scheduler for z/OS generates a job event when a named job is submitted outside of its control.

For both event types, you must specify the triggers in the event-triggered tracking criteria table.

ETT occurrence-related JCL variables

- JCL variables are automatically substituted at job submission
- Usage can be simulated with the Programming Interface (PIF) JCLPREPA statement

ETT JCL Variable	Value: type R if data set triggered	Value: type R if not data set triggered	Value: type J
OETTYPE	R (resource triggered)	R	J (job triggered)
OETCRIT	Data set name	Resource name	Triggering job name
OETJOBN	Closing job name or TSO ID	Blank	Job name
OETJNUM	Blank	Blank	JES job #
OETEVNM	Full data set name	Expanded resource name	Job name
OETGROOT	GDG root data set (GDG's only)	Blank	Blank
OETGGEN	GDG generation number <i>GnnnnVnn</i> , GDG's only	Blank	Blank

© Copyright IBM Corporation 2015

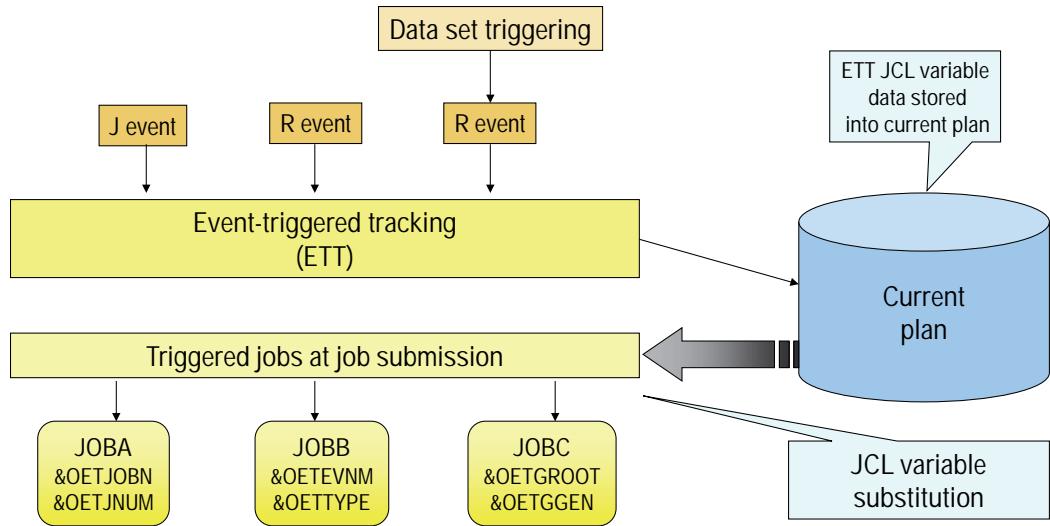
ETT occurrence-related JCL variables

The variables are as follows:

- **OETTYPE**: The event type of the event-triggering criteria name in the ETT table entries
 - Event type **J** for job
 - Event type **R** for special resource
- **OETCRIT**: The event-triggering criteria from the ETT table entries that are used for matching
 - Event type **J** for job: Job name of the triggering job
 - Event type **R** for special resource: The special resource or the data set name
- **OETJOBN**: The full job name that triggered the event
 - Event type **J** for job: Job name of the triggering job
 - Event type **R** for special resource: If not generated with data set triggering, it is blank. If generated by data set triggering as a special resource event, it is the job name or TSO user ID that closed the data set.
- **OETJNUM**: The JES job number of the job that triggered the event. If OETJOBN is blank, this variable is also blank.
 - Event type **J** for job: JES job number
 - Event type **R** for special resource: Blank

- **OETEVNM:** The full triggering event name
 - Event type **J** for job: Job name of the triggering job.
 - Event type **R** for special resource: The full resource name. If it is a GDG data set, it is the full GDG name, including the generation and version (G0000V00). For data set triggering, it is the full data set name.
- **OETGROOT:** The GDG data set root name
 - Event type **J** for job: Blank
 - Event type **R** for special resource: If not generated with data set triggering for a GDG data set, it is blank. If generated by data set triggering as a special resource event for a GDG data set, it is the GDG data set root name.
- **OETGGEN:** The GDG data set generation number (*GnnnnVnn*)
 - Event type **J** for job: Blank
 - Event type **R** for special resource: If not generated with data set triggering for a GDG data set, it is blank. If generated by data set triggering as a special resource event for a GDG data set, it is the GDG data set generation number.

ETT variable substitution overview



© Copyright IBM Corporation 2015

ETT variable substitution overview

This diagram depicts how ETT events provide data values for variable substitution in triggered jobs.

ETT event type J: Sample JCL job

```
//XAPLETT1 JOB 'TRIGGERED JOB',CLASS=A,REGION=4096K,  
//           MSGLEVEL=(1,1),MSGCLASS=R,NOTIFY=&SYSUID  
//*****  
// THIS JOB IS TRIGGERED BY THE COMPLETION OF ANOTHER JOB  
// USING AN ETT EVENT TYPE OF J.  
// THE VARIABLES USED BELOW IN THE DATA SET NAMES WILL BE SUBSTITUTED  
// BEFORE SUBMITTING THIS JOB.  
//*****  
//**%OPC SCAN ← Required Statement  
//*****  
//STEP1 EXEC PGM=IEFBR14  
//SYSPRINT DD SYSOUT=*  
//ETTVAR1 DD DISP=(OLD,DELETE),←  
//           DSN=TWS.&OETCRIT←  
//ETTVAR2 DD DISP=(OLD,DELETE),←  
//           DSN=TWS.&OETTYPE←  
//ETTVAR3 DD DISP=(OLD,DELETE),←  
//           DSN=TWS.&OETEVNM←  
//ETTVAR4 DD DISP=(OLD,DELETE),←  
//           DSN=TWS.&OETJOBN←  
//ETTVAR5 DD DISP=(OLD,DELETE),←  
//           DSN=TWS.&OETJNUM←
```

© Copyright IBM Corporation 2015

ETT event type J: Sample JCL job

The example job is triggered by the completion of a different job by using an ETT event type of **J**.
The **//*%OPC SCAN** statement is required for variable substitution.

Specifying ETT criteria

EQQJMTCL ----- MODIFYING ETT TRACKING CRITERIA ----- Row 1 to 2 of 2				Command ==> _	Scroll ==> PAGE			
Change data in the rows, and/or enter any of the following row commands: I(nn) - Insert, R(nn),RR(nn) - Repeat, D(nn),DD - Delete								
Row cmd	Name of triggering event	Id of associated application		E	J	R	D	A
'''	TRIGJOB1_____	XAPLETT1_____		J	N	N	N	
'''	SPECIAL.RESOURCE.TEAM#_____	XAPLETT2_____		R	N	N	Y	

© Copyright IBM Corporation 2015

Specifying ETT criteria

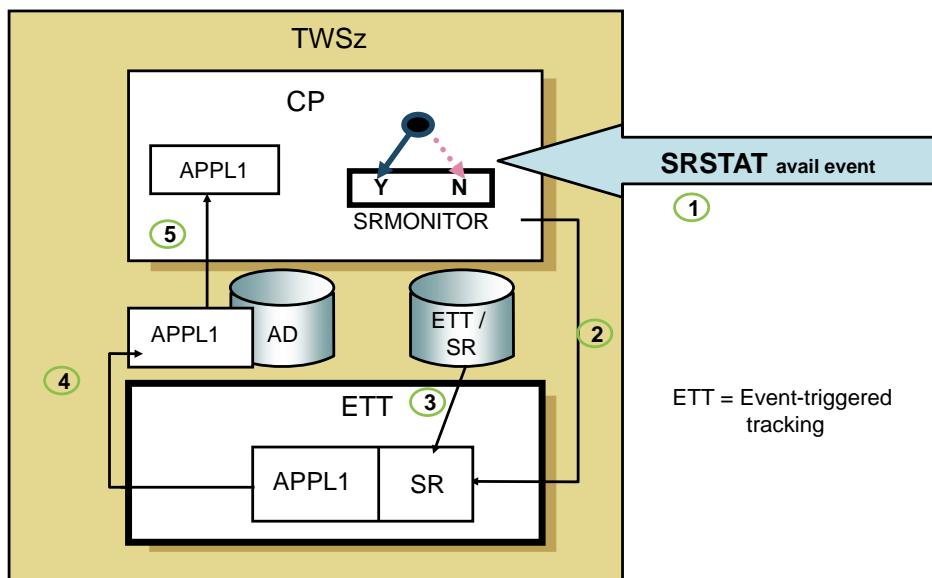
This example shows a job name and special resource trigger in the ETT table. You can use ISPF option **1.7.2** to modify ETT tracking criteria.

- **Row cmd:** The **D** (delete), **I** (insert), and **R** (repeat) commands operate in the usual ISPF edit mode.
- **Name of triggering event:** Specify the name of a job or a special resource, up to 44 characters. Required. You can also use the global search characters (*) and (%) to specify a generic name.
- **ID of associated application:** Specify the name of an application that is defined in the application description (AD) database. Required.
- **E T:** Specify the type of event that you want to cause a dynamic update of the current plan. Required.
 - **J:** A job reader event is the triggering event.
 - **R:** A special resource availability event is the triggering event.
- **J R:** Specify job name replace, which is valid only with event type **J**. Specifies whether the job name of the first operation in the associated application is replaced. Optional.
 - **Y:** The name of the first operation is to be replaced by the job name of the triggering job.
 - **N:** The application is added unchanged.

- **D R:** Specify the dependency resolution, which determines whether external dependencies are added when occurrences are added to the current plan. Optional.
 - **Y:** Add external dependencies.
 - **N:** Do not add external dependencies.
 - **P:** Add only predecessor dependencies.
- **S:** Add only successor dependencies.

Special resource event triggers

- Special resource switched to Available
- Special Resource Monitor notifies ETT processing
- ETT retrieves application ID from the ETT database
- ETT verifies the application is in the application database
- The application is added to the current plan



© Copyright IBM Corporation 2015

Special resource event triggers

When the SRSTAT command changes the status of a special resource to Available, ETT checks the ETT database. If it finds the SRSTAT name, it looks for the associated application description in the AD database and adds it into the current plan.

The data set triggering function can also be used for automatically creating a special resource event that sets the availability to **Yes**. A resource event might, for example, set the status of IMS.MAIN.DBASE to Available. This setting can be the trigger that schedules one or more Tivoli Workload Scheduler for z/OS applications that perform backups and consolidations. You can set this SR status from outside the scheduler if necessary.

Some example uses for ETT are as follows:

- Starting IMS housekeeping when IMS is closed.
- Starting work that needs data sets or files that are created by TSO or VM users.
- Starting work that depends on files transmitted from non-Tivoli Workload Scheduler for z/OS systems, even from other countries.
- Starting work that depends on jobs that are submitted by application owners from TSO.

You can use many events that take place outside the scheduler as triggers to add applications to the current plan. Applications that you want added to the current plan must exist as active applications in the scheduler application database.

Job name event trigger

- Triggering criteria:
 - Explicit job name
 - Generic job name using asterisk (*), percent sign (%) or both
- Job name of the first operation in the added application occurrence can be replaced with the job name of the triggering job

© Copyright IBM Corporation 2015

Job name event trigger

Using a job name as the ETT trigger can cause an application to be added into the current plan. ETT compares job names with names specified as triggering events in the ETT database. If one matches, it loads the associated application from the AD database into the current plan. The trigger name can be generic. For example, a job name that starts with *PAYR** can add application **MASTERPAYROLL**.

You can replace the first job name in the added application with the job name of the triggering job event for tracking jobs that are submitted outside of Tivoli Workload Scheduler for z/OS.

Lesson 4 Data set triggering

Lesson 4 Data set triggering

- Automatically creates a special resource availability event when a data set is closed
- The special resource representing the data set might be a predefined special resource
- Usually used in conjunction with event-triggered tracking
- Included in the IEFU83 system management facilities (SMF) exit routine
- Works with the special resource LIFESPAN-like options

© Copyright IBM Corporation 2015

In this lesson, you learn how to implement and use data set triggering. After completing this lesson, you should be able to implement and use data set triggering.

Data set triggering is a scheduler tool that you use for creating special resource availability events when specific data sets are closed. You can use data set triggering to further enhance the ETT function by using the special resource event as an ETT trigger. Implement the SMF exit routine **IEFUJI** to support data set triggering.

Technote 1020169

Tivoli Workload Scheduler for z/OS data set triggering works by intercepting and examining all SMF data set close records. To set up data set triggering, complete the following steps:

1. Collect SMF14, SMF15, and SMF64 records on the systems where this function is to work.
2. Compile and install the Tivoli Workload Scheduler-provided IEFU83 job-tracking exit routine in SEQQSAMP member EQQU831.

It is possible to trigger only when a data set is closed after creation or update, not after an open for READ (input). Set the SRREAD parameter in the EQQEXIT macro in EQQU831 to SRREAD=NO to disable READ triggers.

You can filter data set triggers based on the user ID used to create or modify a data set. To do this, set the SETUID=*parameter* of the EQQEXIT macro in the Tivoli Workload

Scheduler-supplied IEFUJI exit routine (EQQUJI1) to SETUID=Y. This assumes that the SMFxxUID field is blank for SMF records on the target system. The Tivoli Workload Scheduler for z/OS subsystem sets the JMRUSEID to the value found in the ACEEUSRI. SMF then moves this value into the SMFxxUID field. Tivoli Workload Scheduler for z/OS can then compare it to the value specified in the USERID field of the EQQLSENT macro.

3. Code a series of EQQLSENT macros as described in the *Installation Guide*.
4. Compile the macros from step 3 using the EQQLSJCL member of the SEQQSAMP library.
5. Name the output file EQQDSLST.
6. Place it in the PDS specified by the EQQJCLIB DD statement in the TRACKER start process.
7. The first time a tracker is started following an IPL (Initial Program Load), it automatically attempts to load EQQDSLST. However, it is possible to force the load at any time by issuing the appropriate command. See Appendix B of the *Planning and Scheduling the Workload* manual.
8. Define special resource ETT triggers in the Tivoli Workload Scheduler for z/OS dialogs. Specify the data set names from the EQQLSENT macros as the special resource names.

When the steps complete, whenever any data set is closed, the sequence of events is as follows:

1. The IEFU83 exit routine is called to examine the SMF CLOSE record.
2. The exit routine starts the tracker subsystem. The tracker subsystem compares the data set name and the specific data from the SMF record against the entries in the EQQDSLST.
3. If a match is made, the tracker subsystem creates a special resources status change event record and places it on the WTRQ (Writer Queue) in ECSA (Extended Common Storage Area).
4. The tracker event writer task removes the record from the WTRQ, writes it to the event data set, and passes it to the controller.
5. The controller processes the event in the same way as one created by using an SRSTAT batch job. The data set name is compared to the ETT trigger definitions. If a match is made, an occurrence of the specified application is added to the current plan.

Creating a data set triggering table

EQQLSENT

- STRING = *string/LASTENTRY*
- POS = *numeric position*
- USERID = *user ID filter criteria*
- JOBNAME = *jobname filter criteria*
- AINDIC = (Y/N)
- LIFACT = {Y|N|R}
- LIFTIM = *interval*

Assemble to create the **EQQDSLST** table member in EQQJCLIB

© Copyright IBM Corporation 2015

Creating a data set triggering table

You specify the names of the data sets on which to trigger by creating entries in a data set triggering table. The table is a PDS member. You create entries in the PDS member by assembling the EQQLSENT macro with the required parameters. These parameters specify the selection criteria for each data set to be placed in the data set triggering table.

The EQQLSENT parameters are as follows:

- **STRING**: A required keyword that specifies the character string to be searched for. The string can be from 1 to 44 characters long.
- **POS**: A required keyword that specifies the numeric position within the data set name where the specified string begins.
- **USERID**: An optional keyword that specifies a character string, which is the user ID. The user ID is the one associated with the job, started task, or TSO session that performed the data set close. The string can be from 1 to 8 characters long and can be generic.
- **JOBNAME**: An optional keyword that specifies a character string that is the name of the job or started task that requested the data set close. The string can be from 1 to 8 characters long and can contain wildcard characters.
- **AINDIC**: An optional keyword that specifies whether the special resource is set to available (Y) or unavailable (N). The default is to make the resource available.

- **LIFACT={Y|N|R}**: Optional keyword that specifies the value that the global availability of the special resource resets to. The reset occurs after the interval of time specified by LIFTIM has expired. Possible values are as follows:
 - **Y**: Sets the global availability to Yes.
 - **N**: Sets the global availability to No.
 - **R**: Sets the global availability to blank.

This keyword is valid only if you specify LIFTIM. The default is **R**.

- **LIFTIM=interval**: Optional keyword that specifies the interval of time, in minutes, after which, the global availability of the special resource is reset. It is reset to the value specified by LIFACT. The possible range is from 1 to 999999.

The last entry in the table must be defined as the last entry, for example:

```
EQQLSENT STRING=LASTENTRY
```

The entries are then assembled and loaded into a table named **EQQDSLST**. You can use the sample **EQQLSJCL** in the SEQQSAMP library to specify the EQQLSENT macro and assemble the selection table into the EQQDSLST member of EQQJCLIB. The selection table loads into ECSA when the IBM Tivoli Workload Scheduler for z/OS event writer starts.

When an SMF 14, 15, or 64 record matches a condition in EQQDSLST, a special resource availability event is created. The event is broadcast to all scheduler subsystems that were defined on the system where the SMF record was created.

Data set triggering table: Sample entries

```
EQQLSENT STRING=INGC100.DSTRI,POS=1,JOBNAME=EA73DTX1,USERID=INGC100
EQQLSENT STRING=SYS2.OPC.OPCX.G,POS=1,USERID=INGC100
EQQLSENT STRING=CASHWEST,POS=20,JOBNAME=PCP1D055

EQQLSENT STRING=HAZPULL.ONLINE.TRANS,POS=20,JOBNAME=PCPID100

EQQLSENT STRING=LASTENTRY
```

© Copyright IBM Corporation 2015

Data set triggering table: Sample entries

This slide shows a sample data set triggering table definition that uses EQQLSENT macro definitions.

Loading the data set triggering table

F subsystem_name,NEWDSLST

where *subsystem_name* is the tracker subsystem name

Example

F TWGC, NEWDSLST

© Copyright IBM Corporation 2015

Loading the data set triggering table

Use the z/OS operator MODIFY command to dynamically load the assembled data set triggering table. The format of the command is as follows:

MODIFY trackerSubsystemName,NEWDSLST

Check the tracker message log to verify that the table successfully loaded.

Data set triggering life span values

- You can specify values that are similar to the special resource LIFESPAN in the data set triggering selection table named EQQDSLST
- Two EQQLSENT macro optional parameters represent the LIFESPAN of the data set triggering special resource:
 - LIFTIM: Number of minutes until expiration
Values are 0 to 99999 in minutes.
 - LIFACT: Special resource availability that is set to this value when the expiration value is reached
 - YES: Set global availability to YES
 - NO: Set global availability to NO
 - RESET: Reset the global availability to blank, which is the default value

© Copyright IBM Corporation 2015

Data set triggering life span values

The LIFTIM and LIFACT parameters combine to create an option similar to that of the LIFESPAN parameter that you can use with the SRSTAT command.

Lesson 5 XML, an alternative method

Lesson 5 XML, an alternative method

- You can use XML definitions to provide EDWA data set triggering criteria
- You create XML definitions in a PDS member
 - Member EQQXML01 in SEQQSAMP is a Tivoli Workload Scheduler for z/OS provided sample
- With the ISPF edit feature, you can create or modify the XML definitions
- The XML member is processed by a Tivoli Workload Scheduler for z/OS-provided batch job that invokes the JZOS Batch Launcher
 - Configuration builder job uses zJava
- The output of the configuration builder batch job is one or more members in the EQQEVLIB PDS
 - Writes data to one member for every tracker destination in the XML
EQQEVLIB members are created by the Tivoli Workload Scheduler for z/OS controller at startup based on the destinations defined in the ROUTOPTS statement
 - Deployed centrally by the Tivoli Workload Scheduler for z/OS controller
 - Operators can use the MVS console modify command to deploy definitions

© Copyright IBM Corporation 2015

In this lesson, you learn how to use XML to build the data set trigger criteria. After completing this lesson, you should be able to use XML to build the data set trigger criteria.

XML definitions can be used to provide EDWA data set triggering criteria. XML definitions are created in a PDS member. The member EQQXML01 in the SEQQSAMP data set is a Tivoli Workload Scheduler for z/OS provided sample. You can use ISPF edit to create or modify the XML definitions. The XML member is processed by a Tivoli Workload Scheduler for z/OS provided batch job that invokes the JZOS batch launcher. The job is referred to as the configuration builder job, and it uses zJava. The output of the configuration builder batch job is one or more members in the EQQEVLIB partitioned data set. The Tivoli Workload Scheduler for z/OS controller started task creates one member in the EQQEVLIB data set for every destination that is defined in the ROUTOPTS initialization statement.

The configuration builder job writes data set triggering information to a member in EQQEVLIB (or EVLIB) for every tracker destination that is defined in the XML source PDS member. The member must exist in EQQEVLIB when the configuration builder job runs. The Tivoli Workload Scheduler for z/OS controller centrally deploys these processed XML definitions to tracker destinations. The controller deploys the members from EQQEVLIB to each respective tracker started task, where they are written into the EQQJCLIB data set of the tracker started task as a member name EQQEVLST. Operators can use the MVS console modify command to deploy definitions also. See the training module XML Implementation for details.

The historical run data that is archived for the Tivoli Dynamic Workload Console reporting feature, also requires zJava, does not use XML definitions on z/OS.

Student exercises



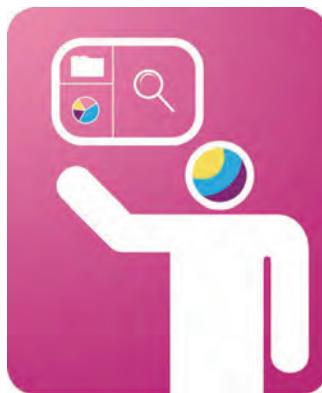
See the Student Exercise Guide Unit 11

© Copyright IBM Corporation 2015

Student exercises

Perform the exercises for this unit.

Instructor demonstration



© Copyright IBM Corporation 2015

Instructor demonstration

The instructor will now demonstrate starting and stopping the controller to make a change to the controller initialization statements.

So far in the student exercises you have not used one of the options made available with IBM Tivoli Workload Scheduler version 8.6.0. but not implemented during the student exercises.

You can configure the job logs to be automatically retrieved when a job ends in error. For MVS jobs, use the JOBLOGRETRIEVAL and RESTARTINFORETRIEVAL keywords in the RCLOPTS statement to activate this option. The instructor will demonstrate how to set, implement, and test this option using the JOBLOGRETRIEVAL keyword of the RCLOPTS statement. You then have the opportunity to test this feature using your application. The demonstration includes the following steps:

1. Modify the RCLOPTS initialization statement using the JOBLOGRETRIEVAL(ONERROR) keyword.
2. Stop and then restart the controller.
3. Verify the new keyword was used by viewing the controller EQQMLOG.
4. Test the automatic job log retrieval function.
5. Test using the T#U11APPL application.

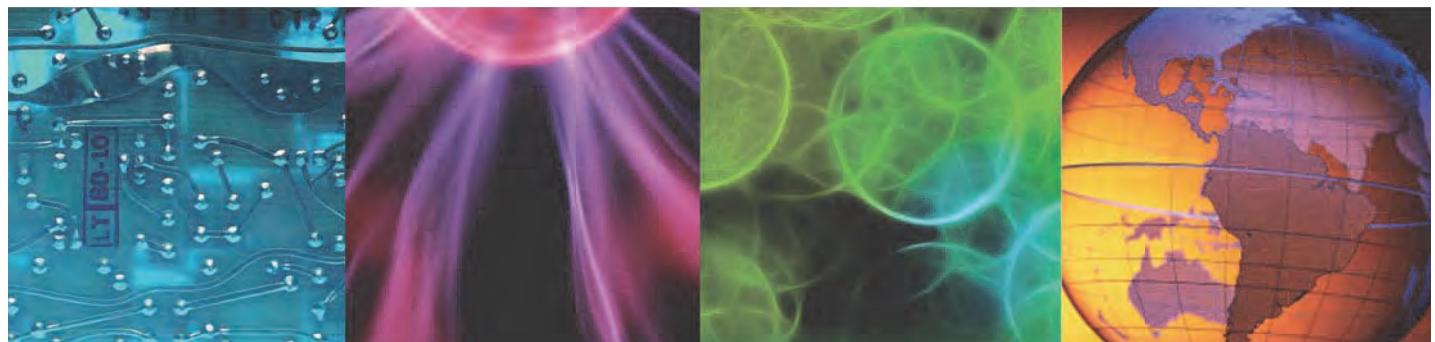
Summary

- Issue the OPSTAT and SRSTAT commands to handle unplanned work and control workflow
- Handle unplanned work with event-triggered tracking
- Create data set triggering table entries
- Enhance event-triggered tracking with data set triggers

© Copyright IBM Corporation 2015

Summary

TM405 1.0



ibm.com/training

Authorized
IBM | Training