IBM Workload Scheduler for z/OS

Customization and Tuning

Version 9.3 SPE (Revised January 2019)

IBM Workload Scheduler for z/OS

Customization and Tuning

Version 9.3 SPE (Revised January 2019)

Note

Before using this information and the product it supports, read the information in "Notices" on page 409.

This edition applies to version 9, release 3, modification level 0 of IBM Workload Scheduler for z/OS (program number 5698-T08) and to all subsequent releases and modifications until otherwise indicated in new editions.

Contents

Figures

Tables

About this publication

IBM Workload Scheduler for z/OS: Customization and Tuning shows you how to customize or tune IBM Workload Scheduler for z/OS. You can use this publication for reference during installation.

Your workload can run on various platforms, but you control it from a central z/OS® system that runs the IBM Workload Scheduler for z/OS controller.

The term *scheduler*, when used in this publication, refers to IBM Workload Scheduler for z/OS. The term *DB2®*, when used in this publication, refers to DATABASE 2 and DB2 Universal Database™.

What is new in this release

Learn what is new in this release.

For information about the new or changed functions in this release, see *Overview*, section *Summary of enhancements*.

For information about the APARs that this release addresses, see the Program Directory and the Dynamic Workload Console Release Notes at http://www-01.ibm.com/support/docview.wss?rs=672&uid=swg27045183.

Who should read this publication

Learn the audience of this publication.

This publication is intended for system programmers, security administrators, and other personnel who install, customize, or tune IBM Workload Scheduler for z/OS.

To use this publication effectively, you need a working knowledge of z/OS and JES concepts and facilities. You should be familiar with the Interactive System Productivity Facility (ISPF), the Interactive System Productivity Facility/Program Development Facility (ISPF/PDF), and the Time-Sharing Option (TSO). A good working knowledge of Virtual Storage Access Method (VSAM) is desirable but not essential.

To implement security, you must know the Resource Access Control Facility (RACF®) or a similar product. To implement IBM Workload Scheduler for z/OS exits or subroutines, you must know job control language (JCL) and have a good working knowledge of a programming language, for example, assembler or PL/I. You can use programming languages that support OS/390® linkage conventions and that can load and delete an assembler program.

Accessibility

Accessibility features help users with a physical disability, such as restricted mobility or limited vision, to use software products successfully.

With this product, you can use assistive technologies to hear and navigate the interface. You can also use the keyboard instead of the mouse to operate all features of the graphical user interface.

For full information, see the Accessibility Appendix in the *IBM Workload Scheduler User's Guide and Reference*.

Technical training

Cloud & Smarter Infrastructure provides technical training.

For Cloud & Smarter Infrastructure technical training information, see: http://www.ibm.com/software/tivoli/education

Support information

IBM provides several ways for you to obtain support when you encounter a problem.

If you have a problem with your IBM software, you want to resolve it quickly. IBM provides the following ways for you to obtain the support you need:

- Searching knowledge bases: You can search across a large collection of known problems and workarounds, Technotes, and other information.
- Obtaining fixes: You can locate the latest fixes that are already available for your product.
- Contacting IBM Software Support: If you still cannot solve your problem, and you need to work with someone from IBM, you can use a variety of ways to contact IBM Software Support.

For more information about these three ways of resolving problems, see the appendix about support information in *IBM Workload Scheduler: Troubleshooting Guide*.

Conventions used in this publication

Conventions used in this publication.

The publication uses several typeface conventions for special terms and actions. Technical changes to the text are indicated by a vertical line to the left of the change. These conventions have the following meanings:

Information type	**Style convention**	**Example**
Commands	All capital letters	CREATE
References in the text to fields on panels	All capital letters	QUANTITY
File and directory names, input you should type in panel fields	Monospace	`MYAPPLICATION`
First time new term introduced, publication titles	Italics	*Application*

How to read syntax diagrams

Syntax diagrams help to show syntax in a graphical way.

Throughout this publication, syntax is described in diagrams like the one shown here, which describes the SRSTAT TSO command:

```
►►──SRSTAT──'──resource name──'──┬──────────────────────────────────────┬──────────►
                                 │               ┌─OPCA───────────┐     │
                                 └─SUBSYS──(─────┼─subsystem name─┼──)──┘
                                                 └─MSTR───────────┘

►──┬───────────────────────────┬─┬──────────────────────────────┬──────────────────►
   │         ┌─KEEP──┐         │ │             ┌─KEEP───┐       │
   └─AVAIL──(┼─RESET─┼──)──────┘ └─DEVIATION──(┼─amount─┼──)────┘
             ├─NO────┤                         └─RESET──┘
             └─YES───┘

►──┬────────────────────────────────┬─┬─────────────────────────┬──────────────────►
   │            ┌─KEEP───┐          │ │            ┌─YES─┐      │
   └─QUANTITY──(┼─amount─┼──)───────┘ └─CREATE──(──┴─NO──┴──)───┘
                └─RESET──┘

►──┬──────────────────────────────────┬────────────────────────────────────────────►◄
   │          ┌─0───────────┐         │
   └─TRACE──(─┴─trace level─┴──)──────┘
```

The symbols have these meanings:

```
►►───────
```
The statement begins here.

```
────────►
```
The statement is continued on the next line.

```
►────────
```
The statement is continued from a previous line.

```
────────►◄
```
The statement ends here.

Read the syntax diagrams from left to right and from top to bottom, following the path of the line.

These are the conventions used in the diagrams:
- Required items appear on the horizontal line (main path):

```
►►──STATEMENT──required item────────────────────────────────────────────►◄
```

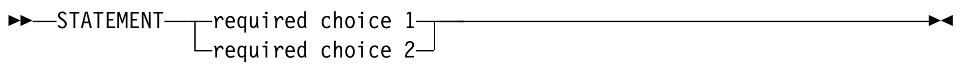
- Optional items appear below the main path:

```
►►──STATEMENT──┬───────────────┬────────────────────────────────────────►◄
               └─optional item─┘
```

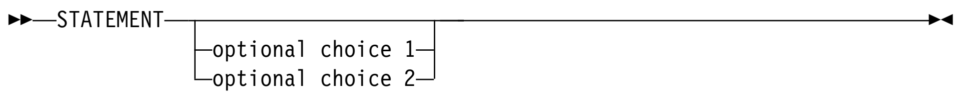
- An arrow returning to the left above the item indicates an item that you can repeat. If a separator is required between items, it is shown on the repeat arrow.

```
                 ┌─,─────────────┐
►►──STATEMENT────▼─repeatable item─┴────────────────────────────────────►◄
```

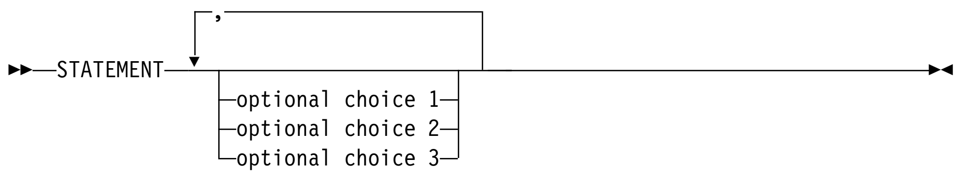
- If you can choose from two or more items, they appear vertically in a stack.
  - If you must choose one of the items, one item of the stack appears on the main path:

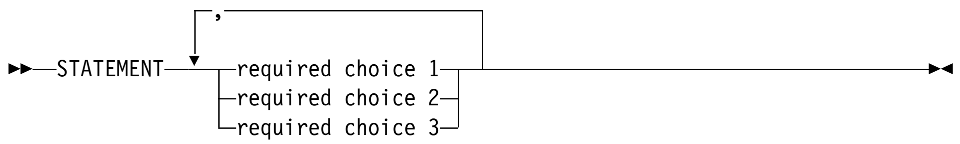


  - If choosing one of the items is optional, the entire stack appears below the main path:

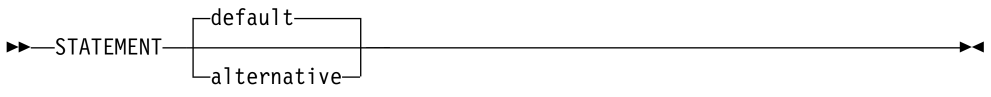


  - A repeat arrow above a stack indicates that you can make more than one choice from the stacked items:

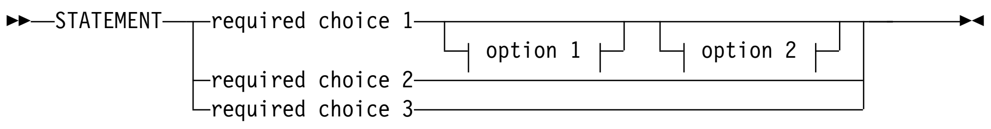




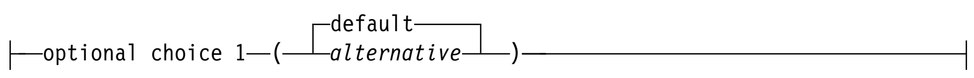
- Parameters that are above the main line are default parameters:



- Keywords appear in uppercase (for example, STATEMENT).
- Parentheses and commas must be entered as part of the command syntax, as shown.
- For complex commands, the item attributes might not fit on one horizontal line. If that line cannot be split, the attributes appear at the bottom of the syntax diagram:



option 1



option 2

```
                          ┌─default─────┐
├──optional choice 2──(───┴─alternative─┴───)──────────────────────────────┤
```

Part 1. Customizing IBM Workload Scheduler for z/OS

How to customize IBM Workload Scheduler for z/OS.

Chapter 1. Defining initialization statements

This chapter describes the IBM Workload Scheduler for z/OS initialization statements that you can define to customize your work. It provides information about specifying the statements and a detailed description for each of them. The statements are listed in alphabetical order.

Diagrams describe the syntax of each statement. For a description of syntax diagrams, see “How to read syntax diagrams” on page xii.

Specifying the statements

The initialization statements determine the tasks that IBM Workload Scheduler for z/OS starts, how IBM Workload Scheduler for z/OS processes work, and the functions that you can use. This chapter describes the syntax rules for creating statements, where you store the statements, and the statements you can select for a tracker, controller, server, data store, batch, or PIF application.

Creating the statements

To customize the IBM Workload Scheduler for z/OS functions, you must set the initialization statements of the PARMLIB library data set. Each statement consists of a statement name, keywords, and keyword values, and follows TSO command syntax rules:

- Statement data must be in columns 1 through 72. Information in columns 73 through 80 is ignored.
- A statement cannot exceed 455 records.
- A statement can be specified only once, unless otherwise indicated. If more than one instance of a statement is specified, only the last one is taken into account. No warning message is reported in EQQMLOG.
- A keyword can be specified in a statement only once, unless otherwise indicated. If more than one instance of a keyword is specified in the same statement, only the last one is taken into account. No warning message is reported in EQQMLOG.
- A blank or comma serves as the delimiter between two keywords; if you supply more than one delimiter, the extra delimiters are ignored.
- Values for keywords are contained within parentheses. If a keyword can have multiple values, the list of values must be separated by valid delimiters. Delimiters are not allowed between a keyword and the left parenthesis of the specified value.
- Some keywords have multiple values. If you want to use the default for a value that occurs before the one you are specifying explicitly, use the delimiter shown in the keyword syntax. For example, if you specify TRACK(READYFIRST) on the JTOPTS statement, the default ALL is used for the first value.
- Type /* to start a comment and */ to end a comment. A comment can span record images in the parameter member and can appear anywhere *except* in the middle of a keyword or a specified value.
- Only comment statements can appear before the statement name on a record.

- A statement continues until the next statement or until the end of records in the member. Continuation characters are not used to define a statement that spans parameter records.
- You can abbreviate keywords to the smallest unambiguous length in the current statement. Statement names cannot be abbreviated.

Note:

1. If an abbreviation matches more than one keyword on a statement, IBM Workload Scheduler for z/OS writes to the message log the TSO-parser message `IKJ56704I` *parameter* `IS AMBIGUOUS`. This can cause a subtask to end or IBM Workload Scheduler for z/OS itself to end.
2. The names of RODM classes, objects, and fields are case-sensitive. Ensure you preserve the case if you specify RODMOPTS statements in the parameter library. Also, if a name contains anything other than alphanumeric or national characters, you must enclose the name in double quotation marks.

Storing the statements

IBM Workload Scheduler for z/OS initialization statements are kept in a parameter library that is identified by the EQQPARM DD statement in the JCL procedure. This library is a partitioned data set with a logical record length of 80 bytes. When you start IBM Workload Scheduler for z/OS, a member in the library containing initialization statements is read. This member is identified by the PARM parameter on the `EXEC PGM=EQQMAJOR` statement in the JCL procedure.

Overriding the EQQPARM statements

When you are using the IBM Workload Scheduler for z/OS program interface (PIF), you can specify a second parameter file in the EQQYPARM DD statement in the JCL of the PIF application. This can be a member of a partitioned data set or a sequential file. EQQYPARM contains an initialization statement, INIT, that overrides values set by the INTFOPTS statement in EQQPARM.

Choosing the appropriate statements

The statements that you define determine the IBM Workload Scheduler for z/OS functions that you can use. Table 1 shows which statements you can define for a tracker, a controller, a server, a data store, a batch, or a PIF application. For a standby controller, you specify the controller statements.

To use the end-to-end scheduling with fault tolerance capabilities, you must also define the job definition statements for fault-tolerant workstations in the SCRPTLIB library as described in Table 2 on page 7.

Table 1. Defining the appropriate initialization statements

Name	Tracker	Controller	Server	Data Store	Batch	PIF	Specifies options for
"ALERTS" on page 7	✓	✓					Generating NetView®, IBM® Tivoli® Monitoring, message log, and WTO alerts.
"AROPTS" on page 11		✓					Automatic job recovery.
"AUDIT" on page 14		✓					Creating audit information for changes to IBM Workload Scheduler for z/OS data.

Table 1. Defining the appropriate initialization statements (continued)

Name	Tracker	Controller	Server	Data Store	Batch	PIF	Specifies options for
"AUDITCP" on page 16		✓					Creating audit information for automatic status changes of an operation condition in the current plan.
"AUTHDEF" on page 17	✓	✓					Security checking.
"BATCHOPT" on page 21					✓		Specifying options for all batch jobs.
"CPUREC" on page 37					✓		Specifying the configuration options for a fault-tolerant workstation.
"DBCSOPTS" on page 38		✓					Japanese language feature.
"DBOPT" on page 39			✓		✓		Dynamic Workload Console reporting.
"DOMREC" on page 42					✓		Defining a domain for distributed agents network.
"DSTOPTS" on page 43				✓			Specifying options for the data store.
"DSTUTIL" on page 48				✓			Specifying options for data store batch utilities and the CleanUp subtask.
"ERDROPTS" on page 51	✓	✓					Event reader task.
"EWTROPTS" on page 52	✓						Event writer task.
"EXITS" on page 57	✓	✓					Calling IBM Workload Scheduler for z/OS exits.
"FLOPTS" on page 58		✓					Communicating with data store (allowing job log retrieval and restart and cleanup functions).
"HTTPOPTS" on page 61		✓					Tracking jobs running on z-centric agents and for retrieving their job execution logs.
"INCLUDE" on page 66		✓					NOERROR table definition members.
"Purpose" on page 66			✓			✓	Run-time options for processing requests from a PIF application and a server.
"INTFOPTS" on page 72		✓					Requests from programming interfaces (required).
"JCCOPTS" on page 73	✓						Job completion checker task.
"JTOPTS" on page 77		✓					Determining how operations behave at workstations and how they are submitted and tracked.

Table 1. Defining the appropriate initialization statements (continued)

Name	Tracker	Controller	Server	Data Store	Batch	PIF	Specifies options for
"MONOPTS" on page 103		✓					Enabling monitoring by an external agent. Used by IBM Tivoli Monitoring.
"MONPOL" on page 104		✓					Defining the monitoring policy to be used by external monitors.
"NOERROR" on page 106		✓					Treating job-tracking error codes as normal completion codes.
"OPCOPTS" on page 110	✓	✓					Starting IBM Workload Scheduler for z/OS subtasks.
"RCLDDP" on page 130		✓					Listing protected DDnames
"RCLDSNP" on page 130		✓					Listing protected data set name.
"RCLOPTS" on page 130		✓					Defining the options used during the restart and cleanup functions.
"RCLSKIP" on page 137		✓					Listing the INCLUDEs to keep at the beginning of a JCL when it is tailored by the Restart and Cleanup function.
"RESOPTS" on page 138		✓					Controlling special resources
"RESOURCE" on page 143					✓		Defining for which special resources daily planning should produce reports.
"RODMOPTS" on page 143		✓					Monitoring special resources through RODM.
"ROUTOPTS" on page 147		✓					Communication routes to tracker, z-centric agent and remote engine destinations.
"SERVOPTS" on page 152			✓				Defining options for a server.
"TCPOPTS" on page 158	✓	✓	✓	✓		✓	Local TCP/IP communication task. You can define it also in an EQQYPARM file referenced in the user logon procedure.
"TOPOLOGY" on page 162			✓		✓		Specifying configuration options for end-to-end scheduling with fault tolerance capabilities.
"TRGOPT" on page 162		✓					Event-driven workload automation support. Used by the Java™ program that creates configuration files for data set triggering.
"TRROPTS" on page 164	✓						The communication route to the controller.
"USRREC" on page 166					✓		Defining the user id and password to be used for the operations running on fault-tolerant workstations.

Table 1. Defining the appropriate initialization statements (continued)

Name	Tracker	Controller	Server	Data Store	Batch	PIF	Specifies options for
"XCFOPTS" on page 167	✓	✓					XCF communications.

Table 2 shows the statements you must define for the jobs running on fault-tolerant workstations. For detailed information, see *IBM Workload Scheduler for z/OS: Scheduling End-to-end with Fault Tolerance Capabilities*.

Table 2. Defining the initialization statements for end-to-end scheduling with fault tolerance capabilities

Name	Description
JOBREC	Fault-tolerant workstation job properties
VARSUB	Variable substitution options
RECOVERY	Recovery for a job whose status is in error and the error code is not FAIL

Refer to the following sections for a detailed syntax and description of each initialization statement, listed in alphabetical order.

ALERTS

Purpose

The ALERTS statement specifies the conditions under which IBM Workload Scheduler for z/OS generates an alert. You can specify this statement for a tracker, controller, or standby controller. You can use these alert actions when an alert condition occurs:
- Send a generic alert to NetView.
- Write a message to the IBM Workload Scheduler for z/OS message log.
- Generate a WTO (write-to-operator) message.
- Send an alert to IBM Tivoli Monitoring (Tivoli Enterprise Portal).

ALERTS is defined in the member of the EQQPARM library as specified by the PARM parameter on the JCL EXEC statement.

Format

```
►►──ALERTS──┬──────────────────────────────────────────────┬──────────────►
            │                   ┌─,──────────────┐         │
            └─GENALERT──(──────▼─┬─DURATION───┬─┴──────)──┘
                                 ├─ERROROPER──┤
                                 ├─LATEOPER───┤
                                 ├─OPCERROR───┤
                                 └─QLIMEXCEED─┘
```

```
►─┬──────────────────────────────────────────┬──────────────────────────────────►
  │            ┌─OPCERROR─────────────┐      │
  │            │  ┌─,──────────────┐  │      │
  │            │  ▼                │  │      │
  └─MLOG──(────┴────┬─DURATION───┬─┴──┴──)──┘
                    ├─ERROROPER──┤
                    ├─LATEOPER───┤
                    ├─QLIMEXCEED─┤
                    └─RESCONT────┘

►─┬───────────────────────────────────────────┬─┬──────────────────────────┬────►
  │              ┌─,──────────────┐           │ │             ┌─YES─┐      │
  │              ▼                │           │ └─MONOPER──(──┴─NO──┴──)──┘
  └─MONALERT──(────┬─DURATION───┬─┴────────)──┘
                   ├─ERROROPER──┤
                   ├─LATEOPER───┤
                   ├─OPCERROR───┤
                   ├─QLIMEXCEED─┤
                   ├─SPECRES────┤
                   └─WLMOPER────┘

►─┬────────────────────────────────────────────────┬────────────────────────────►
  │                  ┌─NETVALRT──────────┐         │
  └─RECEIVERID──(────┴─Alert Receiver ID─┴──)─────┘

►─┬─────────────────────────────────────┬───────────────────────────────────────►◄
  │          ┌─,──────────────┐         │
  │          ▼                │         │
  └─WTO──(─────┬─DURATION───┬─┴──────)──┘
               ├─ERROROPER──┤
               ├─LATEOPER───┤
               ├─OPCERROR───┤
               ├─QLIMEXCEED─┤
               └─RESCONT────┘
```

Parameters

GENALERT(*error condition,...,error condition***)**
: Defines the conditions under which IBM Workload Scheduler for z/OS sends a generic alert to NetView.

MLOG(*error condition,...,error condition***|OPCERROR)**
: Defines the conditions under which IBM Workload Scheduler for z/OS writes a message to the message log.

MONALERT(*error condition,...,error condition***)**
: Defines the conditions under which IBM Workload Scheduler for z/OS sends a generic alert to the IBM Tivoli Monitoring agent. This parameter is significant only if the MONOPTS statement is provided.

MONOPER(YES|NO)
: Defines if the following error conditions specified in the MONALERT parameter are in effect for monitored jobs only (default) or for all jobs. The error conditions are ERROROPER, LATEOPER, DURATION, and WLMOPER.

RECEIVERID(*Alert Receiver ID***|NETVALRT)**
: Defines the NetView alert receiver that generic alerts are sent to. Specify this keyword if the alert receiver in NetView address space that handles IBM Workload Scheduler for z/OS alert automation does not have the NetView default ID, NETVALRT.

WTO(*error condition,...,error condition*)
Defines the conditions under which IBM Workload Scheduler for z/OS generates a write-to-operator (WTO) message.

Error conditions

You can specify one or more of the following error conditions for each alert type. Note that only OPCERROR and QLIMEXCEED are applicable to a tracker; other error conditions are ignored if you specify them.

DURATION
The alert action is taken when an operation in the current plan is active for an unexpectedly long time. This means that an operation that has started (extended status S) must be active longer than its estimated duration multiplied by either the following values:

- The alert action limit that you set for ALEACTION (for details, see "JTOPTS" on page 77).

  -OR-
- The duration feedback limit that you set for LIMFDBK (for details, see "JTOPTS" on page 77). This value is used if ALEACTION is not set.

then divided by 100.

For example, if an operation has an estimated duration of 10 minutes and the limit for the alert action is 200, the alert action is taken if the operation is active for longer than 20 minutes. The alert action is also taken if the operation has been started but the associated job or started task has not yet started to run after 10 minutes (no A2/B2 event has been received), that is, the operation has had status/extended status SU and SQ totally for more than 10 minutes. The alert action is taken only for operations that have started status.

For MLOG and WTO alert actions, message EQQE028I is issued for an operation at a general workstation and EQQE038I for an operation at a computer or printer workstation for long running operations. Message EQQE039I is issued for computer operations that have been submitted but have not started.

Notes:

1. The value used to select the operations for which a long duration alert must be issued is set with the ALEACTION keyword in the "JTOPTS" on page 77 statement. If ALEACTION is not specified, the value set for LIMFDBK is used instead. In this case, the value for the feedback limit that you can optionally enter in the application description is ignored.
2. If the alert action limit is 0 or the alert action limit is not specified and the feedback limit is 0, the alert action is taken as soon as the operation is active longer than its estimated duration.
3. If the estimated duration of an operation is 99 hours, 59 minutes and 01 seconds, no duration alert is sent for this operation.

ERROROPER
The alert action is taken when an operation in the current plan is set to ended-in-error status. For MLOG and WTO alert actions, message EQQE026I is issued for an operation at a general workstation, and EQQE036I for an operation at a computer or printer workstation.

LATEOPER
The alert action is taken when an operation in the current plan becomes

late. An operation is considered late if it reaches its latest start time and does not have the status started, complete, or deleted. For MLOG and WTO alert actions, message EQQE027I is issued for an operation at a general workstation and EQQE037I for an operation at a computer or printer workstation.

Note: Use LATEOPER only when deadlines are accurate because it can affect the performance of IBM Workload Scheduler for z/OS.

OPCERROR
The alert action is taken when an IBM Workload Scheduler for z/OS subtask or IBM Workload Scheduler for z/OS subsystem ends unexpectedly. For MLOG and WTO alert actions, message EQQZ045W and EQN019E are issued. If GENALERT action is specified and the EQQN019E alert condition occurs, then the subtask-failed alert is sent to NetView.

Note: OPCERROR is always in effect for MLOG.

QLIMEXCEED
The alert action is taken each time an IBM Workload Scheduler for z/OS subtask queue exceeds a threshold value. Except for the event-writer queue, the threshold values are multiples of 10 between 10% and 90%, and then 95% and 99%. IBM Workload Scheduler for z/OS checks the size of a queue when an event is added to it. Except for the event-writer queue, IBM Workload Scheduler for z/OS subtask queues can contain up to 32,000 elements.

The size of the event-writer queue is determined by the ECSA you allocate. The queue is checked each time the event writer is about to read events; the alert action is taken if the queue exceeds 50%. If the event-writer queue becomes full, a message is issued indicating how many events have been lost.

The value in the alert shows the actual percentage used, which will be more than the threshold value. For MLOG and WTO alert actions, message EQQZ106W is issued.

RESCONT
You can specify RESCONT (resource contention) only for MLOG and WTO alert types. The alert action is taken when an operation has been waiting on a resource queue for the time specified on the CONTENTIONTIME keyword of RESOPTS. Message EQQQ515W is issued.

SPECRES
The alert action is taken when the time that an operation in the current plan is waiting to allocate a given resource exceeds the time specified by the RESOPTS CONTENTIONTIME parameter. This alert takes effect when it is defined in the MONALERT parameter.

WLMOPER
The alert action is taken when an operation in the current plan is promoted by WLM. The alert is sent only if specified in the MONALERT parameter.

Examples

```
ALERTS MLOG(ERROROPER,LATEOPER,DURATION)    1
       WTO(DURATION)                        2
       GENALERT(ERROROPER)                  3
       MONALERT(DURATION,OPCERROR,WLMOPER)  4
       MONOPER(YES)                         5
```

In this example of an ALERTS statement:

1 IBM Workload Scheduler for z/OS writes a message to the message log for operations that are set to ended-in-error status, are late, or are active an unexpectedly long time. Although it is not specified, the OPCERROR condition also applies for the MLOG alert action.

2 A write-to-operator message is generated for operations that are active an unexpectedly long time.

3 IBM Workload Scheduler for z/OS sends a generic alert to NetView for operations that are set to ended-in-error status. By default, generic alerts are sent to the alert receiver NETVALRT.

4 IBM Workload Scheduler for z/OS sends a generic alert to IBM Tivoli Monitoring for operations that have long durations, for IBM Workload Scheduler for z/OS substaks or subsystems that end unexpectedly, and for operations promoted by WLM.

5 IBM Workload Scheduler for z/OS sends a generic alert to IBM Tivoli Monitoring only for monitored operations (operations with EXTERNAL MONITOR = YES) satisfying the conditions specified in the MONALERT parameter and not for all jobs.

AROPTS

Purpose

The AROPTS statement defines run-time options for automatic job and started-task recovery. It is used by a controller or standby controller where OPCOPTS RECOVERY(YES) is specified.

AROPTS is defined in the member of the EQQPARM library as specified by the ARPARM parameter on the OPCOPTS statement.

Format

```
►►──AROPTS──┬────────────────────────────────────┬──┬─────────────────────────────┬──────►
            │                ┌─JCLUSER───┐       │  │                ┌─NO──┐       │
            └─AUTHUSER──(────┼─GROUP─────┼──)────┘  └─CHKRESTART──(──┴─YES─┴──)────┘
                             ├─JCLEDITOR─┤
                             └─OWNER─────┘

►──┬───────────────────────────┬──┬────────────────────────────────┬──────────────────────►
   │               ┌─2359─┐    │  │                 ┌─NOAR───┐     │
   └─ENDTIME──(────┴─hhmm─┴─)──┘  └─EXCLUDECC──(────┴─code───┴─)───┘

►──┬─────────────────────────────────────────────────────────┬────────────────────────────►
   │                 ┌─6──────────────────────────────┐      │
   └─EXCLUDERC──(────┴─highest no-recovery return code─┴──)──┘
```

```
►─┬──────────────────────────────────────────────────┬────────────────────────►
  └─PREDWS──(────predecessor workstation name────)───┘

►─┬───────────────────────────────┬─┬──────────────────────────┬─────────────►◄
  │                ┌─0000─┐       │ │              ┌─NO──┐     │
  └─STARTTIME──(───┴─hhmm─┴───)───┘ └─USERREQ──(───┴─YES─┴──)──┘
```

Parameters

AUTHUSER(GROUP|JCLEDITOR|OWNER|JCLUSER)
Defines where IBM Workload Scheduler for z/OS retrieves the name that is used for authority checking in automatic recovery:

GROUP
The authority group ID of the failing occurrence.

JCLEDITOR
The name in the JCL repository (EQQJSnDS) file. If the JCL has not been updated through the dialogs or PIF, IBM Workload Scheduler for z/OS does not perform authority checking. The name in the ISPF statistics of the job library (EQQJBLIB) is not used.

OWNER
The owner ID of the failing occurrence. Owner ID is truncated if it is more than 8 characters.

JCLUSER
The name of the user who created or last updated the JCL. JCLUSER is the default value. If the JCL has not been updated through the dialogs or PIF, IBM Workload Scheduler for z/OS uses the name in the ISPF statistics of the job library (EQQJBLIB). If no statistics exist, IBM Workload Scheduler for z/OS does not perform authority checking.

CHKRESTART(YES/NO)
Usually the Automatic Recovery function postpones the recovery actions, whenever cleanup type is Immediate, until any cleanup actions have been successfully completed. This is done even if the recovery actions do not require the operation to be restarted, for example when ADDAPPL is the required recovery action. This is the default behavior and corresponds to AROPTS CHKRESTART(NO).

If you want the postpone mechanism (wait for cleanup actions to complete) to occur only when recovery actions require a job or step restart, you must specify AROPTS CHKRESTART(YES). When AROPTS CHKRESTART(YES) is specified, and the recovery actions do not require a restart, the recovery actions are performed immediately and then the immediate cleanup actions start, but there is no waiting for their completion.

Cleanup types None and Immediate are compatible with Automatic Recovery in all scenarios. Cleanup types Manual and Automatic are only compatible with Automatic Recovery if AROPTS CHKRESTART(YES) is specified and the recovery actions do not require a job or step restart (recovery proceeds, in this case). If AROPTS CHKRESTART(NO) is specified, or the recovery actions require a job or step restart and the failing operation cleanup type is Manual or Automatic, the Automatic Recovery function issues an appropriate error message and stops processing the related operation.

ENDTIME(*hhmm*|2359)
Defines the end of the time range for which automatic recovery is performed for jobs and started tasks that contain a RECOVER statement without a TIME

specification. This default end time-of-day is specified in the format *hhmm*, where *hh* is the hour in the range 00–23, and *mm* is the minute in the range 00–59.

EXCLUDECC(*code***|NOAR)**
Defines an individual error code or a case code for which no automatic recovery is performed unless explicitly requested by a RECOVER statement in the failing job or started task. A case code is a group of abend and return codes. Case codes are defined in the EQQCASEM module using the EQQCASEC macro. The default case code is NOAR, which contains S122, S222, CAN, JCLI, JCL, and JCCE. For more information about case codes, see "Creating case-code-definition modules" on page 323.

EXCLUDERC(*highest no-recovery return code***|6)**
Defines the maximum step-completion-code value for which no automatic recovery is performed unless explicitly requested by a RECOVER statement in the job or started task.

PREDWS(*predecessor workstation name***)**
Defines a default predecessor workstation name that is used by automatic recovery to create an external dependency when a predecessor occurrence to a failing operation is added to the current plan. The default is used if no specific predecessor operation is found. This might occur, for example, if an operation has been deleted or a workstation name has been changed. The highest operation number in the predecessor occurrence with a workstation name that matches the PREDWS definition is set up as the predecessor.

If PREDWS is not specified or if there is no match on the workstation name, the end point in the predecessor occurrence is used to establish the dependency. If the predecessor occurrence contains multiple end points, the end point with the highest operation number is used.

The workstation name can be specified generically.

STARTTIME(*hhmm***|0000)**
Defines the start of the time range for which automatic recovery is performed for jobs and started tasks that contain a RECOVER statement without a TIME specification. The default start time-of-day is specified in the format *hhmm*, where *hh* is the hour in the range 00–23, and *mm* is the minute in the range 00–59.

USERREQ(YES|NO)
Defines if automatic recovery is permitted to update the current plan when no user ID can be established or when the user ID is not known to the security product. The value that is specified for AUTHUSER determines where IBM Workload Scheduler for z/OS attempts to retrieve a name for authority checking.

Specify YES if a valid user ID is required. Specify NO if automatic recovery is permitted to update the current plan, even though no user ID is available or the user ID is not known to the security product. NO is the default value.

Examples

```
AROPTS STARTTIME(0800)   1
       ENDTIME(1700)     2
```

In this example of an AROPTS statement, jobs and started tasks that contain a RECOVER statement without a TIME specification are recovered only if they fail between 0800 and 1700.

AUDIT

Purpose

The AUDIT statement defines how changes to IBM Workload Scheduler for z/OS data is logged. You can specify this statement for a controller or standby controller. You can also use more than one AUDIT statement to define IBM Workload Scheduler for z/OS auditing actions.

When an access is logged, a record containing auditing information is written to the current log data set, which depends on the value you set for the AMOUNT parameter. The mapping of the records on the log data set is described in *Diagnosis Guide and Reference*.

When you request logging of accesses to JCL data, logging is performed if the JCL is present in, or is inserted into, the JCL repository.

Note:

1. Changes to current plan, current-plan-extension, and event-triggering records are always logged for IBM Workload Scheduler for z/OS restart purposes. Read accesses to these resources are not logged, and you cannot prevent the logging of update accesses.
2. If you set AMOUNT(DATA) or AMOUNT(KEY), auditing records are written to the job-tracking-log data set and this affects how often IBM Workload Scheduler for z/OS performs a current-plan-backup process. Keep this in mind when specifying a value for the BACKUP keyword of JTOPTS. Also consider auditing records when allocating the job-tracking-log data sets, especially if you specify AMOUNT(DATA).
3. The AUDIT records contain a field to indicate whether or not they are related to a real data change. Based on that field, EQQAUDIT program reports only those records that actually changed from an external point of view. For example, when a dialog user selects a data item in a list through the M (Modify) row command and exits using the PF3 key, without changing the data item, the process rewrites related data records to keep any change that the scheduler performed for internal purposes. In this case, the scheduler creates an AUDIT record for a data item that the external user did not really change, however EQQAUDIT does not report that record. In particular, using the J row command always causes the JCL to be written to the JS file, even if the dialog user did not change the JCL.

The IBM Workload Scheduler for z/OS sample library contains a program that can format reports of job-tracking, track-log, and extended-auditing records. See “IBM Workload Scheduler for z/OS auditing package” on page 401 for more information.

AUDIT is defined in the member of the EQQPARM library as specified by the PARM parameter on the JCL EXEC statement.

Format

```
►►─AUDIT─┬──────────────────────────────┬─┬──────────────────────────────────┬──────────────►
         │               ┌─UPDATE─┐     │ │                ┌─KEY──────┐       │
         └─ACCESS──(─────┴─READ───┴──)──┘ └─AMOUNT──(─────┼─DATA─────┼──)────┘
                                                           └─EXTENDED─┘

►─FILE──(──┬─ALL─────┬──)─────────────────────────────────────────────────────────────────►◄
           └─┬─AD───┬┘
             ├─CAL──┤
             ├─JLIB─┤
             ├─JS───┤
             ├─JV───┤
             ├─LT───┤
             ├─OI───┤
             ├─PER──┤
             ├─RD───┤
             ├─RG───┤
             ├─VAR──┤
             ├─WS───┤
             └─WSCL─┘
```

Parameters

ACCESS(READ|UPDATE)
Defines when IBM Workload Scheduler for z/OS tracks accesses to the file. Specify UPDATE to track all changes to the file. Specify READ to track all accesses.

AMOUNT(DATA|EXTENDED|KEY)
Defines how much data is to be logged by IBM Workload Scheduler for z/OS for accesses to the file.

Specify KEY to log only the VSAM key (this is enough to identify the resource). Specify DATA to log the VSAM key and record.

Specify EXTENDED to use the extended auditing feature. In this case, all data about the database changes is logged in the EQQDB*nn* data sets (instead of EQQJT*nn*). The EQQDB*nn* data sets are created by the EQQPCS14 sample and provide the input for the extended auditing feature. The EXTENDED argument applies only to the following files:
- AD
- CAL
- JV
- OI
- PER
- RD
- RG
- WS

FILE(*name of IBM Workload Scheduler for z/OS file*)
Defines the name of the file for which auditing is being defined. A separate AUDIT statement is required to identify each file for which you want auditing information recorded. Specify one of these files:

AD	Application description
CAL	Calendar definition
JLIB	JCL in a job library
JS	JCL in the JS VSAM file
JV	JCL variable table

LT	Long-term-plan occurrence
OI	Operator instruction
PER	Period definition
RD	Special resource descriptions
RG	Run cycle group definition
VAR	JCL variable setting
WS	Workstation description
WSCL	All-workstations-closed definition
ALL	All files

Examples

```
AUDIT FILE(ALL) ACCESS(UPDATE) AMOUNT(KEY)   1
AUDIT FILE(JS) ACCESS(UPDATE) AMOUNT(DATA)   2
```

In this example of AUDIT statements:

1 IBM Workload Scheduler for z/OS tracks all changes to all files by logging the VSAM key.

2 IBM Workload Scheduler for z/OS tracks all changes to JCL for jobs and started tasks by logging the VSAM key and record.

AUDITCP

Purpose

Use the AUDITCP statement to activate or deactivate the tracking process automatically performed by the scheduler, related to the status change of an operation condition in the current plan.

The following statements are optional because the related TRL records are used only for monitoring and not for JT reapply processing.

Format

```
►►──AUDITCP──┬──────────────────────────────┬──┬──────────────────────────────┬──────────►
             │                ┌─NO──┐       │  │                ┌─NO──┐       │
             └─CONDSTATUS──(──┴─YES─┴──)────┘  └─CDEPSTATUS──(──┴─YES─┴──)────┘

►──┬───────────────────────────────┬──┬───────────────────────────────┬───────────────────►◄
   │                 ┌─NO──┐       │  │                  ┌─NO──┐      │
   └─CDEPSTEPEND──(──┴─YES─┴──)────┘  └─UNEXPECTEDRC──(──┴─YES─┴──)───┘
```

Parameters

CONDSTATUS(YES|NO)
: If you set this parameter to YES, every time a condition automatically changes its status, the scheduler creates a data record TRLBDY45 in the JT log.

  If you use the default, the scheduler performs the standard process for TRLBDY45 records, that is creates them only to track condition status change after a MCP request.

CDEPSTATUS(YES|NO)
If you set this parameter to YES, every time a condition dependency changes its status, the scheduler creates a data record TRLBDY44 in the JT log.

If you use the default, the scheduler performs the standard process for TRLBDY44 records, that is creates them only to track condition dependency status change after a MCP request.

CDEPSTEPEND(YES|NO)
If you set this parameter to YES, every time a step-end event is received and a condition dependency referring it is found in the plan, the scheduler creates a data record TRLBDY49 in the JT log.

If you use the default, TRLBDY49 records are not created.

UNEXPECTEDRC(YES|NO)
If you set this parameter to YES, every time an Unexpected RC situation occurs (messages EQQE141W, EQQE142W or EQQM215W are issued on controller MLOG) the scheduler creates a data record TRLBDY50 in the JT log.

If you use the default TRLBDY50, records are not created.

AUTHDEF

Purpose

The AUTHDEF statement specifies the IBM Workload Scheduler for z/OS resources that are defined to a security product. For a description about how you use IBM Workload Scheduler for z/OS security features to protect IBM Workload Scheduler for z/OS functions and data, see Chapter 3, "Implementing security," on page 185.

You can specify this statement for a controller, a standby controller, or a tracker.

AUTHDEF is defined in the member of the EQQPARM library as specified by the PARM parameter on the JCL EXEC statement.

Format

```
►►─AUTHDEF──┬──────────────────────────────────────────┬──────────────►
            │          ┌─OPCCLASS───────────────┐      │
            └─CLASS──(─┴─name of resource class─┴──)───┘

►─┬────────────────────────────────────────────────────┬─────────────►
  └─COMMAND1, ...., COMMAND9──(───list of commands───)─┘

►─┬──────────────────────────────┬───────────────────────────────────►
  │                 ┌─ALL───┐    │
  └─LISTLOGGING──(──┼─FIRST─┼──)─┘
                    └─NONE──┘
```

```
|  ►──┬────────────────────────────────────────────────┬──────────►
      │                    ┌─,──────────────────────┐  │
      │                    ▼                        │  │
      └─SUBRESOURCES──(────┬─AD.ADNAME───────────────┬─┴──)─┘
                           ├─AD.ADGDDEF──────────────┤
                           ├─AD.GROUP────────────────┤
                           ├─AD.JOBNAME──────────────┤
                           ├─AD.NAME─────────────────┤
                           ├─AD.OWNER────────────────┤
                           ├─AD.RESNAME──────────────┤
                           ├─AD.SECELEM──────────────┤
                           ├─AD.UFVAL────────────────┤
                           ├─CL.CALNAME──────────────┤
                           ├─CP.ADD──────────────────┤
                           ├─CP.ADDOPER──────────────┤
                           ├─CP.ADNAME───────────────┤
                           ├─CP.COMMAND1, ...., CP.COMMAND9─┤
                           ├─CP.CPGDDEF──────────────┤
                           ├─CP.DELETE───────────────┤
                           ├─CP.DELOPER──────────────┤
                           ├─CP.GROUP────────────────┤
                           ├─CP.JOBNAME──────────────┤
                           ├─CP.MODDEP───────────────┤
                           ├─CP.MODIFY───────────────┤
                           ├─CP.MODOPER──────────────┤
                           ├─CP.MODOPSTAT────────────┤
                           ├─CP.NAME─────────────────┤
                           ├─CP.OWNER────────────────┤
                           ├─CP.SECELEM──────────────┤
                           ├─CP.UFVAL────────────────┤
                           ├─CP.WSNAME───────────────┤
                           ├─CP.ZWSOPER──────────────┤
                           ├─ET.ADNAME───────────────┤
                           ├─ET.ETNAME───────────────┤
                           ├─JL.DSNAME───────────────┤
                           ├─JL.MEMBER───────────────┤
                           ├─JS.ADNAME───────────────┤
                           ├─JS.GROUP────────────────┤
                           ├─JS.JOBNAME──────────────┤
                           ├─JS.OWNER────────────────┤
                           ├─JS.WSNAME───────────────┤
                           ├─JV.OWNER────────────────┤
                           ├─JV.TABNAME──────────────┤
                           ├─LT.ADNAME───────────────┤
                           ├─LT.LTGDDEF──────────────┤
                           ├─LT.OWNER────────────────┤
                           ├─OI.ADNAME───────────────┤
                           ├─PR.PERNAME──────────────┤
                           ├─RD.RDNAME───────────────┤
                           ├─RG.RGNAME───────────────┤
                           ├─RG.OWNER────────────────┤
                           ├─RL.ADNAME───────────────┤
                           ├─RL.GROUP────────────────┤
                           ├─RL.OWNER────────────────┤
                           ├─RL.WSNAME───────────────┤
                           ├─RL.WSSTAT───────────────┤
                           ├─SR.SRNAME───────────────┤
                           └─WS.WSNAME───────────────┘
```

```
►─┬──────────────────────┬──────────────────────────────────────────►◄
  │             ┌─0─┐    │
  └─TRACE──(────┼─4─┼──)─┘
                └─8─┘
```

Parameters

CLASS(*name of resource class***|OPCCLASS)**
Defines the name of the security resource class that protects IBM Workload Scheduler for z/OS resources. The value is valid until you specify a different value and restart the IBM Workload Scheduler for z/OS address space.

Consider the following checklist when using this parameter:

- The resource class must be defined in the RACF class descriptor and routing tables.
- New definitions in the RACF class descriptor and routing tables require an IPL.
- If multiple controller subsystems require separate policies, they require separate classes.
- IBMOPC is a predefined class that you can use with no need for an IPL if only one class is required.
- After a RACF migration, consider redefining any class you defined in a previous version of RACF.
- The default class OPCCLASS is not already defined in RACF. Before using this class, make sure there are the necessary entries in the RACF class descriptor and routing tables.

COMMAND1, ..., COMMAND9(*list of commands***)**
Defines the list of commands to which you want to authorize a user. If the same command is listed in more than one COMMAND*n* parameter and different levels of authorization are assigned, the authorization with the higher level of privileges is always applied to the command.

You can specify any combinations of the following occurrence and operation commands:

Table 3. Occurrence commands that you can specify in the **Commandn** *parameter*

Command	Description
C	Complete an occurrence
CG	Complete group
DG	Delete group
R	Rerun
RG	Remove from group
W	Set waiting

Table 4. Operation commands that you can specify in the **Commandn** *parameter*

Command	Description
ARC	Attempt Automatic Recovery
BND	Bind Operation
DJ	Delete JCL
EX	Execute

Table 4. Operation commands that you can specify in the **Commandn** *parameter (continued)*

Command	Description
J	Edit JCL
JR	JR, Fast Path JR
K	Kill (K and KR)
MH	Manual Hold
MR	Manual Release
NP	NOP
RI	Recovery Info (PY and PN)
SC	SC, Fast Path SC
SJR	Simple Job Restart
SR	SR, Fast Path SR
UN	UN NOP

LISTLOGGING(FIRST|NONE|ALL)
In the resource profile, you define how data is logged for accesses to a resource. If you restrict access to IBM Workload Scheduler for z/OS data on the record level by specifying subresources, a request to list IBM Workload Scheduler for z/OS data can result in several access violations being recorded for those records that satisfy the filter criteria but to which the user is not permitted access. LISTLOGGING lets you alter the amount of data that is logged for list requests.

Specify FIRST when logging is performed only for the first read attempt to a resource. Logging occurs only for the first entry that has a profile, which specifies that logging should occur. Specify NONE if no logging is performed. Specify ALL if logging is performed as specified in the profile for the resource. ALL is the default value.

SUBRESOURCES(*resource*,...,*resource*)
Defines whether IBM Workload Scheduler for z/OS checks on the record level if a user is authorized to access information in an IBM Workload Scheduler for z/OS VSAM file.

In the list of resources you can specify one or more of the items shown in the syntax diagram. For a description of all the fixed resources and subresources, see Table 29 on page 200.

Whenever a user accesses a record, for example in the AD file, IBM Workload Scheduler for z/OS checks if the user is authorized to access the record in the manner intended. To do this, a resource name is generated, and a request is sent through SAF (system authorization facility) to the security system to test the user authority. For example, if you specify AD.ADNAME, the application name is retrieved from the record, and the prefix **ADA.** is added to create the resource name. The security system is then called to test if this resource exists in the resource class defined by the CLASS keyword and if the user is authorized to access it. The default resource list for the SUBRESOURCES keyword is the empty list. This means that the default is to use already established authority and not to check the user authority to access individual VSAM records.

Note: If you have specified OPCHOST(NO) in the "OPCOPTS" on page 110 statement, only the RL.WSNAME, RL.WSSTAT, and SR.SRNAME subresources are relevant. AD.SECELEM and CP.SECELEM are relevant only if you run System Automation V3.1 (with the appropriate maintenance level installed), or later. When set, they protect the whole System Automation information in the AD segment and CP33 record, respectively.

TRACE(4|8|0)
Defines if IBM Workload Scheduler for z/OS writes trace information to the message log (EQQMLOG) each time the RACROUTE macro is invoked. Specify 0, which is the default value, if you do not want trace information. Specify 4 if you want partial trace information. Specify 8 if you want full trace information.

Examples

```
AUTHDEF CLASS(OPCCLASS)                    1
        SUBRESOURCES(AD.ADNAME,WS.WSNAME)  2
```

In this example of an AUTHDEF statement:

1 The default resource class is used.

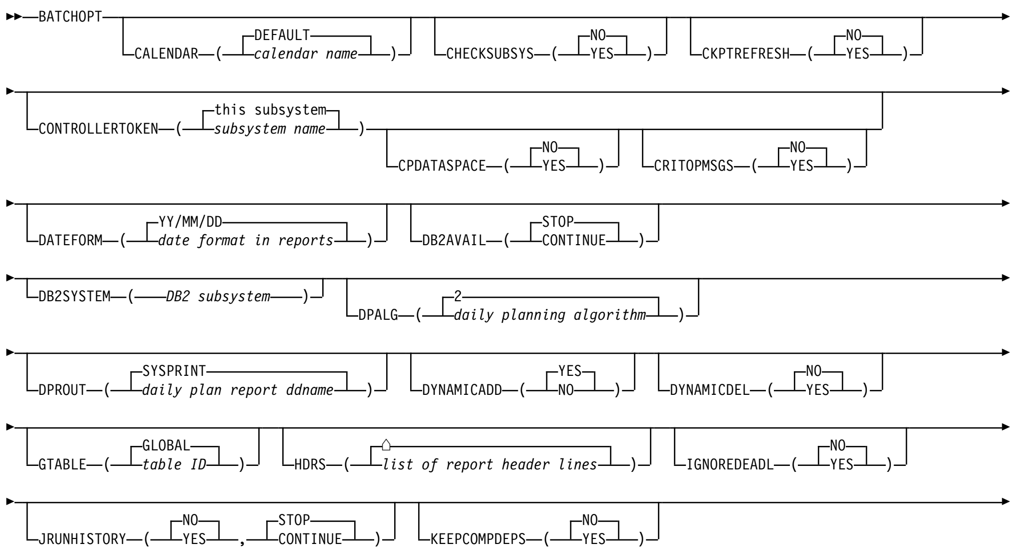
2 IBM Workload Scheduler for z/OS will verify authorization for application descriptions (by checking the application name) and workstations (by checking the workstation name).

BATCHOPT

Purpose

The BATCHOPT statement defines run-time options for IBM Workload Scheduler for z/OS batch jobs. BATCHOPT is defined in its own member of the EQQPARM library. The member is referenced by IBM Workload Scheduler for z/OS batch jobs at run time.

Format


BATCHOPT
CALENDAR (DEFAULT | calendar name)
CHECKSUBSYS (NO | YES)
CKPTREFRESH (NO | YES)
CONTROLLERTOKEN (this subsystem | subsystem name)
CPDATASPACE (NO | YES)
CRITOPMSGS (NO | YES)
DATEFORM (YY/MM/DD | date format in reports)
DB2AVAIL (STOP | CONTINUE)
DB2SYSTEM (DB2 subsystem)
DPALG (2 | daily planning algorithm)
DPROUT (SYSPRINT | daily plan report ddname)
DYNAMICADD (YES | NO)
DYNAMICDEL (NO | YES)
GTABLE (GLOBAL | table ID)
HDRS (list of report header lines)
IGNOREDEADL (NO | YES)
JRUNHISTORY (NO | YES , STOP | CONTINUE)
KEEPCOMPDEPS (NO | YES)

```
►──┬──────────────────────────────────────────┬─┬────────────────────────────┬──────────►
   │            ┌─01───────────────────────┐  │ │               ┌─YES─┐      │
   └─LOGID──(───┴─ID on track log data set─┴─)┘ └─LTPDEPRES──(──┴─NO──┴──)──┘

►──┬─────────────────────────────────────────┬─┬─────────────────────────────────────────┬──►
   │                 ┌─0──────────────┐      │ │                    ┌─5000───────────┐   │
   └─LTPREMSHIFT──(──┴─number of days─┴──)───┘ └─MAXHISTORYROWS──(──┴─number of rows─┴─)─┘

►──┬──────────────────────────────┬─┬───────────────────────┬─┬───────────────────────┬─┬──────────────────────┬──►
   │               ┌─32767───┐    │ │              ┌─YES─┐  │ │              ┌─NO──┐  │ │              ┌─N─┐   │
   └─MAXOCCNUM──(──┴─nnnnnnn─┴─)──┘ └─NCPTROUT──(──┴─NO──┴─)┘ └─OCPTROUT──(──┼─YES─┼─)┘ └─OPERDALL──(──┴─Y─┴─)─┘
                                                                             └─CMP─┘

►──┬──────────────────────────────┬─┬───────────────────────┬─┬─────────────────────────────────────────┬──►
   │                 ┌─NO──┐      │ │              ┌─N─┐    │ │              ┌─55─────────────────────┐  │
   └─OPERHISTORY──(──┴─YES─┴──)───┘ └─OPERIALL──(──┴─Y─┴──)─┘ └─PAGESIZE──(──┴─page size for reports─┴─)┘

►──┬────────────────────────────────────────────┬─┬─────────────────────────────────────────────────────────┬──►
   │               ┌─6──────────────────────┐   │ └─PREDWS──(───default predecessor workstation name───)──┘
   └─PLANHOUR──(───┴─planning period start──┴─)─┘

►──┬─────────────────────────┬─┬─────────────────────────┬─┬───────────────────────┬─┬────────────────────────┬──►
   │             ┌─YES─┐     │ │              ┌─NO──┐    │ │             ┌─NO──┐   │ │                ┌─0────┐ │
   └─PREVRES──(──┴─NO──┴──)──┘ └─RCLEANUP──(──┴─YES─┴──)─┘ └─REMDSREC──(─┴─YES─┴─)─┘ └─RETAINOPER──(──┴─days─┴─)┘

►──┬─────────────────────────────────────┬─┬─────────────────────────────────────────────────────┬──►
   │            ┌─OPCA───────────┐       │ └─SUCCWS──(───default successor workstation name───)──┘
   └─SUBSYS──(──┴─subsystem name─┴──)────┘

►──┬──────────────────────────────┬─┬────────────────────────────────────┬─┬────────────────────────────┬──►◄
   │                ┌─NO──┐       │ │               ┌─TPLGPARM────┐      │ │                ┌─ABEND─┐   │
   └─TIMEDEPCHK──(──┴─YES─┴──)────┘ └─TPLGYPRM──(───┴─member name─┴──)───┘ └─VALEACTION──(──┼─END───┼─)─┘
                                                                                            └─WARN──┘
```

Parameters

CALENDAR(*calendar name*|DEFAULT)

Defines the IBM Workload Scheduler for z/OS default calendar. You can specify a name, from 1 to 16 characters, referencing a calendar in the calendar database.

The IBM Workload Scheduler for z/OS planning functions use this calendar:

- During long-term plan batch processing, to determine run days for an application if no calendar is specified in the application description.
- When extending the current plan, if the specified input parameters indicate that only work days are considered in the extension period.
- During long-term plan processing and Mass Update, to validate that the EVERY options specified for an application run cycle is consistent with the application calendar work day end time.

If no default calendar is specified, or if the specified calendar does not exist in the calendar database, a calendar with the name DEFAULT is used as the default calendar. If no calendar with the name DEFAULT exists, all days are considered work days and the work day end time is set to 00.00.

If the specified calendar does not exist in the calendar database, Mass Update does not check for the EVERY options.

CHECKSUBSYS(YES|NO)

Controls whether the daily planning program checks that it can synchronize with the processing in the controller address space. The synchronization is performed using ENQ locking with scope SYSTEMS. The synchronization can fail because:

- The controller is not started.
- The controller is not executing within the same GRS complex as the daily planning program.
- The SUBSYS keyword does not identify the correct controller.
- The SUBSYS keyword does not identify the correct controller.

If NO is specified or defaulted, the last two cases will result in a corrupt status record in the checkpoint file, and the new current plan will not be taken over by the controller. Manual recovery is then required, and the current plan files might be corrupted.

Specify YES for the daily planning program to end if the IBM Workload Scheduler for z/OS controller specified by the SUBSYS keyword cannot be reached when daily planning requests the subsystem to prepare for the planning process. When NO is specified or defaulted, the daily planning program continues to process if the controller cannot be reached. Ensure this only occurs because the subsystem is not started.

CKPTREFRESH(YES|NO)
This parameter can be used to refresh the checkpoint data set during a daily plan extend or replan from the old entries that are not "in use" any more.

Note: The default value is NO and the checkpoint data set keeps a record for each remote destination defined in the ROUTOPTS and connected at least once with the Controller. Thus, if you are going to add some new destinations or change some names already defined in the ROUTOPTS statements, it would be advisable to set this parameter to YES at least once, after completing the planned changes. Message EQQN036I can help to determine when the 1000 limit is getting close and suggest that it is time to refresh the checkpoint data set.

CONTROLLERTOKEN(*subsystem name*|this subsystem)
This parameter is used for the history function. The subsystem name is a key for the history information. When the current plan is extended, the BATCHOPT CONTROLLERTOKEN value is used to write the information. When the information is retrieved from the database, the OPCOPTS CONTROLLERTOKEN value is used, so you can rerun jobs initiated from another subsystem current plan (for example, when a standby system takes over) by specifying the correct token after takeover.

If you specify OPERHISTORY(NO) or omit the OPERHISTORY keyword, the CONTROLLERTOKEN keyword is ignored.

| **CPDATASPACE(YES|NO)**
| Specify YES for the daily planning program to load portions of the in-storage operations and occurrences into the data space, when building the network of data to simulate the scheduling.

| Setting this parameter to YES is particularly helpful when you generate a current plan including more than one million operations, because it optimizes the use of storage. With a CP of this size, you must also:

- Allocate the following data sets as extended VSAM:
  - EQQACPDS
  - EQQCP1DS
  - EQQCP2DS
  - EQQNCPDS
  - EQQSCPDS
- In the batch jobs of the daily plan, allocate the EQQDIN data set with DSTYPE LARGE.

| For more details about managing a current plan with one million jobs, see *Managing the Workload*.

This setting does not affect the final content of the NCP. When the daily planning program terminates, the data space is deleted.

CRITOPMSGS(NO|YES)
This parameter is used to filter the messages that are issued for operations that are missing the deadline. Specify YES to issue missing deadline messages that refer only to operations with the CRITICAL field set to P or W. Specify NO to issue missing deadline messages that refer to any types of operation. The default is NO.

DATEFORM(*date format in reports*|YY/MM/DD)
Specifies the date format used in reports. You can specify any combination of century (*CC*), year (*YY*), month (*MM*), and day (*DD*). *CC* is optional. You can also use the day number (*DDD*) from the Julian calendar rather than month and day. The delimiter can be any character other than C, Y, M, or D. However, if you specify *CCYYMMDD*, in any order, you cannot use delimiters.

DB2AVAIL(CONTINUE|STOP)
This keyword is used for the history function. It specifies the action that IBM Workload Scheduler for z/OS takes if the DB2 subsystem is not available. If you specify STOP, IBM Workload Scheduler for z/OS writes a DB2 error message to the EQQMLOG data set and, if the current plan is being extended, stops with return code 8.

If you specify OPERHISTORY(NO) or omit the OPERHISTORY keyword, the DB2AVAIL keyword is ignored.

DB2SYSTEM(*DB2 subsystem*)
This keyword is required for the history function. It specifies the DB2 subsystem, as in the IEFSSNxx member of SYS1.PARMLIB.

If you specify OPERHISTORY(YES) but omit the DB2SYSTEM keyword, an error message is issued.

If you specify OPERHISTORY(NO) or omit the OPERHISTORY keyword, the DB2SYSTEM keyword is ignored.

DPALG(*daily planning algorithm*|2)
Determines how much an application's priority will influence the way it is planned. Acceptable values are 0 to 32 767.

A value of 0 means priority is disregarded; 2 gives a reasonable balance between priorities and operation slack times; and 32 767 means operations are ordered only on priority.

DPROUT(*daily plan report ddname*|SYSPRINT)
Specifies the ddname that IBM Workload Scheduler for z/OS uses when producing daily-planning reports.

DYNAMICADD(NO|YES)
DYNAMICADD determines if IBM Workload Scheduler for z/OS creates a special resource during planning, if an operation needs a resource that is not defined in the special resource database.

Specify YES if you want IBM Workload Scheduler for z/OS to create a resource in the current plan. The special resource database is not updated.

Specify NO if IBM Workload Scheduler for z/OS is not to dynamically create a resource. IBM Workload Scheduler for z/OS plans the operation as if it does not use the resource.

A dynamically created resource has these values:

Special resource
The name specified by the allocating operation.

Text Blank.

Specres group ID
Blank.

Hiperbatch
No.

Used for
Both planning and control.

On error
Blank. If an error occurs, IBM Workload Scheduler for z/OS uses the value specified in the operation details or, if this field is blank, the value of the ONERROR keyword of RESOPTS.

Default values
The resource has these default values for quantity and availability:

Quantity
The amount specified in the first allocating operation. The quantity is increased if more operations plan to allocate the special resource at the same time. IBM Workload Scheduler for z/OS increases the quantity only for dynamically created resources to avoid contention.

Available
Yes.

Intervals
No intervals are created. The default values specify the quantity and availability.

Workstations
The resource has default value *, which means all workstations. Operations on all workstation can allocate the resource.

Also see the DYNAMICADD keyword of RESOPTS in the list of RESOPTS “Parameters” on page 139, which controls the dynamic creation of undefined special resources in the current plan.

DYNAMICDEL(YES|NO)
DYNAMICDEL determines if a special resource that has been dynamically added to the current plan can be deleted if the current plan is changed, without checking the normal conditions listed in the section "setting the global values" of the *Managing the Workload* manual.

Specify NO if dynamically added changes can be deleted when the current plan is changed.

Specify YES if dynamically added changes can be deleted when the current plan is changed, without further checking.

Note: It is strongly recommended that DYNAMICDEL(YES) be specified by all users who use significantly the dynamic addition of special resources. If the records dynamically added are not deleted using this feature, the size of the dataspace continues to increase in time with an initial performance degradation that worsens until the space available is exhausted and the batch job terminates with Abend 01D.

GTABLE(*table ID***|**__**GLOBAL**__**)**

Defines the name of the global variable table for the IBM Workload Scheduler for z/OS complex when you schedule in an end-to-end with fault tolerance capabilities environment. This table contains variable definitions that can be used for any operation within the IBM Workload Scheduler for z/OS complex. The global variable table is searched when a table (or variable within a table) that is referenced by an operation cannot be found. You must set this keyword to the same value as the GTABLE keyword in the OPCOPTS statement (for additional details, see GTABLE in the list of OPCOPTS "Parameters" on page 112. The value of this keyword is specified in the TABLES parameter of the SCRPTLIB VARSUB statement (for details, see the description about this statement in the *IBM Workload Scheduler for z/OS: Scheduling End-to-end with Fault Tolerance Capabilities* manual) and is processed during the execution of the daily planning batch jobs. It is ignored if you do not use the end-to-end scheduling with fault tolerance capabilities feature.

You can specify only one table ID for the IBM Workload Scheduler for z/OS complex. IBM Workload Scheduler for z/OS uses the default name GLOBAL if you do not specify a table ID.

HDRS(*list of report header lines***|**__△__**)**

Defines a list of character strings containing report headers. The maximum length of each report header is 120 bytes. The batch job does not use more than three report headers. __△__ represents a blank line and is the default report header.

If you use the IBM Workload Scheduler for z/OS Japanese language feature, you can specify report headers in DBCS format.

IGNOREDEADL(YES|__**NO**__**)**

When setting this keyword to YES, any operation in the current plan will have the deadline forced to the CP tail end date and time, (that is, after the planning end) except operations that are set as critical or suppress-if-late jobs. This implies that their deadlines are completely ignored in the calculation of latest start times. The immediate consequence is that, in a critical network, operations not set as suppress-if-late, will result being late based on the deadline of the critical job. This parameter does not affect occurrence deadlines because automatic setting occurs at operation level.

If you set this parameter to YES, the same behaviour also applies to any dynamic change using modify current plan (that is, dynamic occurrence addition, occurrence modification and so on). To return to the previous behaviour, a new current plan must be created specifying NO or using the default value.

If the deadline is manually anticipated, this causes a recalculation of the latest start time for eligible operations within the occurrence (that is, affected operations and internal predecessors). Eligible operations are ALL the operations, if IGNOREDEADL value is set to NO, and critical jobs and suppress-if-late jobs when IGNOREDEADL is set to YES. For performance reasons, external predecessors are not affected by the deadline change until DP batch runs.

Any attempt to manually change deadlines for non eligible operations is ignored by MCP engine. For operations that have their deadline forced to the CP tail end using the parameter IGNOREDEADL, message EQQM225W will not be issued.

JRUNHISTORY(YES|__**NO**__**, CONTINUE|**__**STOP**__**)**

The JRUNHISTORY parameter contains two keyword values:

- The first keyword value defines whether the daily planning batch process backs up the old current plan on Generation Data Group (GDG) data sets. Specify YES to request the backup, that is necessary to populate the history DB2 tables used by the Tivoli Dynamic Workload Console reporting feature. Use the default value NO to skip the backup.
- The second keyword value is ignored if you specify NO as first keyword value. It defines whether the daily planning job continues when an error occurs during the backup process:

  CONTINUE
  The daily plan batch job continues and a warning message is issued. The occurrences that were removed from the current plan will not be archived.

  STOP The daily plan batch job fails. This is the default.

KEEPCOMPDEPS(NO|YES)

This keyword determines the permanence of external dependencies on a completed operation that belongs to occurrences that are still active when a daily plan job is submitted (either extended or replanned). When a plan is extended or a replan is performed, using this parameter you can maintain dependencies between two operations that belong to different occurrences when the predecessor operation has completed. Normally, the dependency is removed after the run of a plan job, even though the completed operation is still in the plan because it belongs to an active occurrence.

Example: Assume you have Occurrence A with Operation1 and Operation2, and Occurrence B with Operation1 and Operation2, where Operation1, in both occurrences, is the predecessor of Operation2, and where Operation1 (Occurrence A) is also a predecessor of Operation2 (Occurrence B), thereby creating an external dependency. If Operation1 (Occurrence A) completes and Operation2 (Occurrence A) is in error, then Occurrence A is in error status. Assume also that Operation1 (Occurrence B) is waiting for a time dependency. If you now run a daily plan job, both occurrences remain in the new plan, but the external dependency is removed because the predecessor (Operation1 - Occurrence A) is complete. Using the **KEEPCOMPDEPS** keyword set to YES, the dependency is maintained in the new plan.

Valid values are:

NO The default. When the plan is extended, external dependencies on a completed operation are removed, even if the occurrence is still active.

YES When the plan is extended, external dependencies remain defined on the operation. This facilitates a rerun operation because the dependency does not need to be redefined.

LOGID(*ID on track log data set***|01)**
An identification placed in all records on the track log (EQQTROUT). The length is 2 characters. Valid values are in the numeric range from 01 to 99.

If you have specified the EQQTROUT data set in the daily planning JCL, the job-tracking archive data set (EQQJTARC) is copied to EQQTROUT. NCPTROUT and OCPTROUT keywords specify if current plan records are also copied. If more than one controller is active and the same EQQTROUT data set is used to collect job-tracking information, you can use LOGID to differentiate between the logs from the different controllers.

LTPDEPRES(NO|YES)
This keyword is used by the batch job that extends the long-term plan.

Specify YES if changes to IBM Workload Scheduler for z/OS databases are reflected in both the extended part of the LTP and the existing part of the plan. The existing part of the LTP is from the time of the end of the current plan to the end of the LTP. Specify NO if only the extended part of the LTP reflects changes to the databases. The existing part of the LTP is not changed when the batch job is run to extend the LTP.

| An LTP occurrence that has been modified through the dialogs or PIF is not
| affected by changes to the databases, even if you specify LTPDEPRES(YES).
| During extend processing, an old occurrence can be copied from the old LTP to
| the new LTP for any of the following reasons:

- | A predecessor or successor dependency was added manually to the LTP occurrence.
- | A dependency was deleted or a dependency was manually changed.
- | The occurrence was manually changed in the LTP.
- | The occurrence has a successor that is in the current plan.

LTPDEPRES does not affect the removal of completed occurrences from the LTP. Whenever you run a modify or extend batch job, all completed occurrences, whose input arrival precedes the day on which the earliest uncompleted occurrence exists, are removed from the LTP.

For more information about the long-term plan, see *Managing the Workload*.

LTPREMSHIFT(*number of days***|0)**

Used in the extension of the long-term plan for dependency management. Normally, at each extension the LTP process removes all the occurrences that precede an FNONC (first not completed occurrence). You can use LTPREMSHIFT to keep the occurrences that completed within a certain number of days before the FNONC.

Specify a number from 0 to 7 to define that the occurrences that completed within that number of days before the FNONC be kept through the LTP extension (that is, copied to the renewed LTP and to the new current plan). The parameter becomes effective at the next DP extension.

MAXHISTORYROWS(*number of rows***|5000**

This keyword, if specified with a nonzero value, tells IBM Workload Scheduler for z/OS the maximum number of rows that can be retrieved from DB2 by using the HIST command. The default value is 5000. the maximum value is 999999. If you specify 0, all the rows are retrieved without limitations.

For example, by specifying 1000 you set 1000 as the maximum number of rows that can be retrieved by the HIST command processing. If less than 1000 rows are retrieved, the rows are shown in the requesting panel. If more that 1000 rows are retrieved, the HIST command processing is interrupted, an error message is issued, and no data is shown in the requesting panel. To reduce the amount of data to retrieve, set the appropriate filtering criteria.

MAXOCCNUM(*nnnnnnn***|32767)**

IBM Workload Scheduler for z/OS maintains an upper limit on the number of occurrences in the current plan. When this limit is exceeded during the extension of the plan or during its creation, the daily planning batch program fails and no new plan is created. If the keyword is omitted, the limit is 32767 occurrences. It is advisable not to set the parameter to a larger number than required by actual workload needs, due to the increased overhead incurred. Occurrences coming from the old plan remain unaffected.

NCPTROUT(NO|YES)
Defines if IBM Workload Scheduler for z/OS copies records from the new current plan to the data set referenced by the EQQTROUT ddname in the daily planning process. If you specify NO, tracklog records are not created for the workstation, occurrence, and operation records updated by the daily planning process when a new current plan is created.

The default value, YES, results in the generation of tracklog records for the corresponding new current plan records.

OCPTROUT(YES|CMP|NO)
Defines if IBM Workload Scheduler for z/OS copies records from the old current plan to the data set referenced by the EQQTROUT ddname in the daily planning process. If you specify YES, tracklog records are created for old current plan record types 01, 02, 03, and 04. Type 03 records carried through to the new current plan for reporting purposes and for pending predecessor occurrences are not copied to the EQQTROUT file, because these records are copied in a subsequent daily plan.

If you specify CMP, the current-plan record type 03 for completed and deleted occurrences is copied, like type 01 and type 04 records.

The default value, NO, defines that IBM Workload Scheduler for z/OS does not copy records from the old current plan to EQQTROUT during the daily planning process.

OPERDALL(Y|N)
This parameter is used by the long-term-planning batch jobs when determining the deadline date for an operation that has a deadline day offset greater than zero, relative to the application input-arrival date.

Y means that IBM Workload Scheduler for z/OS considers all days (work and free) when determining the deadline date for the operation. N means that only work days are considered.

The calendar you specified in the application description determines if a day is a work day or a free day. If you did not specify a calendar, the default calendar is used.

Note: OPERDALL does not affect occurrence deadlines.

OPERHISTORY(YES|NO)
Specifies that IBM Workload Scheduler for z/OS stores completed operation information in the DB2 database. This parameter is required for the history function.However, any operation records longer than 32,000 bytes are skipped by the archiving process.

If you specify YES, you must also set the DB2SYSTEM parameter.

If you specify NO or omit this parameter, but specify related parameters (like DB2SYSTEM, CONTROLLERTOKEN, DB2AVAIL, or RETAINOPER) a warning message is issued to inform you that the history function is not active (and, therefore, that completed occurrences were not stored in DB2).

OPERIALL(Y|N)
This parameter is used by the long-term-planning batch jobs when determining the input-arrival date for an operation that has an input-arrival-day offset greater than zero, relative to the application input-arrival date.

Y means that IBM Workload Scheduler for z/OS considers all days (work and free) when determining the input-arrival date for the operation. N means that only work days are considered.

Whether a day is a work day or a free day is determined from the calendar specified in the application description. If no calendar is specified, the default calendar is used.

Note: OPERIALL does not affect occurrence input arrival.

PAGESIZE(*page size for reports***|55)**
Defines the number of lines per page in reports generated by IBM Workload Scheduler for z/OS. You can specify a numeric value from 30 to 500.

PLANHOUR(*planning period start***|6)**
Defines the time-of-day in hours when the daily planning period starts. This value must be the same as the value you specified for PLANSTART on the JTOPTS statement.

PREDWS(*default predecessor workstation name***)**
Defines a default predecessor workstation name that is used by daily planning to create an external dependency when a specific predecessor operation cannot be found. This might occur, for example, if an operation has been deleted or a workstation name has been changed. The highest operation number in the predecessor occurrence with a workstation name that matches the PREDWS definition is set up as the predecessor.

If PREDWS is not specified, or if there is no match on the workstation name, the end point in the predecessor occurrence is used to establish the dependency. If the predecessor occurrence contains multiple end points, the end point with the highest operation number is used. The daily planning generates a warning message to identify the occurrences involved in the dependency.

The workstation name can be specified generically.

PREVRES(NO|YES)
Defines if IBM Workload Scheduler for z/OS maintains data and produce reports for the previous 24 hours when a daily planning job is run. The default is to produce reports for the previous planning period.

RCLEANUP (YES NO)
The RCLEANUP value must match the RCLEANUP value in the controller OPCOPTS. It specifies if DP batch must create the CP records containing the list of the operinfo associated to the completed occurrence. The controller later deletes this operinfo when the turnover has completed, if RCLEANUP(YES) is specified in the controller OPCOPTS.

REMDSREC (YES NO)
Controls if the daily planning program ignores and does not carry forward any CP16 data store records that accumulated. Typically, use the value NO (this is also the default). If you need to set this parameter to YES, change it back to NO as soon as possible.

CP16 data store records accumulate in several situations, for example when the RCLEANUP parameter of BATCHOPT and the RCLEANUP parameter of OPCOPTS are set to different values. Usually, these two RCLEANUP parameters are either both set to YES or both set to NO. However, if RCLEANUP(YES) is set in BATCHOPT and RCLEANUP(NO) is set in OPCOPTS, the CP16 records accumulate over the period that RCLEANUP(NO) is specified for the controller task. This occurs because the CP16 records are created by the batch job, but they are never processed or deleted by the controller. Later, when you set RCLEANUP(YES) also for the controller, the

| controller performance might significantly decrease and return to normal after
| a long time, or the controller might run out of storage and end with ABEND
| S878.

| When you set REMDSREC=YES, accumulated CP16 records are deleted and
| message EQQ0548I is issued in the message log.

RETAINBIND(*days***|7)**
After an operation is bound in the current plan, the controller sends notifications with the bind result and with the status of the job bound to the engine that requested the bind. On the controller the request for the bind is stored in a *bind request record* until the last status change is notified. If a notification cannot be sent, the **RETAINBIND** keyword specifies how many days the bind request record is to be kept after the instance of the job bound is not included anymore in the current plan. The default value is 7.

RETAINOPER(*days***|0)**
This keyword is used for the history function. It specifies how many days an operation is to be kept in the history database.

Zero is the default. It means that, before IBM Workload Scheduler for z/OS inserts a new occurrence into a DB2 table, it deletes all occurrences that have the same application identifier and that belong to previous versions stored after a current plan extension or replan operation has completed.

If you specify a nonzero value, IBM Workload Scheduler for z/OS purges data that is older than that number of days. You can thus keep more than one occurrence of each operation.

If you specify OPERHISTORY(NO) or omit the OPERHISTORY keyword, the RETAINOPER keyword is ignored.

SUBSYS(*subsystem name***|OPCA)**
Identifies the name of the controller subsystem for which the batch job is run.

Batch jobs of IBM Workload Scheduler for z/OS, for example daily planning or long-term planning, must run on the same system as the controller if global resource serialization (GRS) is not used. If your installation uses GRS, the batch jobs and the controller must run on systems in the same GRS complex.

SUCCWS(*default successor workstation name***)**
Defines a default successor workstation name that is used by daily planning to create an external dependency when a specific successor operation cannot be found. This might occur, for example, if an operation has been deleted or a workstation name has been changed. The lowest operation number in the successor occurrence with a workstation name that matches the SUCCWS definition is set up as the successor.

If SUCCWS is not specified, or if there is no match on the workstation name, the starting point in the successor occurrence is used to establish the dependency. If the successor occurrence contains multiple start points, the start point with the lowest operation number is used. The daily planning program generates a warning message to identify the occurrences involved in the dependency.

The workstation name can be specified generically.

TIMEDEPCHK(YES|NO)

When setting this keyword to YES, planned start time calculation for operations that are not time-dependent starts from the beginning of the extend or re-plan interval instead of starting from the Input Arrival Time. This affects the Dynamic Critical Path feature (also known as, Workload Service Assurance), as planned start and end times are used to initiate the estimated start and end times that are responsible for the choice of the critical path among all the possible predecessor paths that start from a critical job.

Moreover, if you specify YES, any new entry added to the critical job table coming from a dynamic addition to the current plan (no planned start time) will have the estimated start time set based on the current date and time if the operation is not time dependent, or on Input Arrival Time if the operation is time dependent and the Input Arrival Time is later than the current date. For example, if the Input Arrival Time is 10:00 AM and the current date is 9:00 AM, the current date is used for operations that are not time-dependent, while for time-dependent operations, the Input Arrival Time is used.

The negative or default value (NO) corresponds to the previous behaviour that calculates estimated start times from Input Arrival Times in any case. To return to the previous behaviour, a new current plan must be created specifying NO for this parameter or using the default. When YES is specified and current date and time is used as estimated start time, this value is saved as planned start time in the operation record. The main reason of this choice is given by the need of using it in the recovery scenarios (controller restart).

Consider also that for any value of this parameter, in case of dynamically added operations, estimated start and end times setting does not cause any update in the estimated times of its predecessors and successors in the critical job table. As soon as a new current plan is created, all these times are recalculated without any additional processing effort. This behavior reflects the original design of the feature with the objective of keeping data updated without heavily affecting performances.

TPLGYPRM (*member name*|TPLGPARM)
Specify this keyword if you want to activate the end-to-end scheduling with fault tolerance capabilities feature in the daily plan.

The specified member name is a member of the PARMLIB in which the fault-tolerant end-to-end options are defined by the TOPOLOGY statement.

VALEACTION(END|WARN|ABEND)
Defines what action a daily planning batch program takes when its validation code detects an inconsistency in the data.

Specify ABEND if an abnormal end must occur (this is the default value). The daily planning job ends with error code S0C1, and a new current plan is not created. Error information is written to the diagnostic data set, EQQDUMP. The characters VALE*xxxx* follow the invalid operation code (X'0000') and identify the module where the error occurred. The system dump produced at the time of the abend will be needed if the error is to be reported as a program defect.

Specify END if the daily planning job must end. Error information is written to the diagnostic data set, EQQDUMP. A new current plan is not created.

Specify WARN if the daily planning job, where possible, must bypass detected errors. Errors are described in messages that are written to the message log and diagnostic information that is written to EQQDUMP. If errors can be bypassed, a new current plan is created. Check the current plan as soon as possible because it might contain errors. For example, a dependency might not have been resolved.

Examples

```
BATCHOPT SUBSYS(OPCB)                          1
         CALENDAR(NSW)                         2
         SUCCWS(CPU*)                          3
         PREDWS(CPU*)                          4
         DATEFORM('YY-MM-DD')                  5
         HDRS('HEADER NUMBER 1',               6
              'HEADER 2       ',
              'THIS IS THE THIRD HEADER LINE')
```

In this example of a BATCHOPT statement:

1 The batch job is run against the OPCB subsystem.

2 The default calendar NSW is used.

3 Any operation that is executing on a workstation whose name starts with the characters CPU is eligible as a default successor.

4 Any operation that is executing on a workstation whose name starts with the characters CPU is eligible as a default predecessor.

5 The date format is year, month, and day, separated by a hyphen (-).

6 The report headers have this text:

```
HEADER NUMBER 1
HEADER 2
THIS IS THE THIRD HEADER LINE
```

BKPTOPTS

Purpose

The optional BKPTOPTS statement defines the local attributes for the TCP/IP communication between the primary controller (that is, a subsystem with OPCOPTS OPCHOST set to YES, PLEX, or STANDBY) and the backup controller. Define this statement on both controllers.

Note: If in ROUTOPTS you set that the primary controller uses HTTP or HTTPS communications to reach a destination, ensure that you set the same protocol for the backup controller to communicate.

Format

```
| ►►──BKPTOPTS──┬────────────────────────────┬─┬──────────────────────────────────────────────┬──────────────►
                │             ┌─NO──┐        │ │                ┌─60───────────────────┐      │
                └─CHECKROLE──(┴─YES─┴──)─────┘ └─CONNTIMEOUT──(─┴─TCPIP timeout interval┴──)──┘

►──┬─────────────────────────────────────────────┬─┬──────────────────────────────────────────────────┬────────►
   │                ┌─EQQSECP1────────────┐       │ │                ┌─EQQRECP1───────────────┐        │
   └─CP1DUMPPROC──(─┴─dump procedure name─┴──)───┘ └─CP1RESTPROC──(─┴─restore procedure name─┴──)────┘

►──┬─────────────────────────────────────────────┬─┬──────────────────────────────────────────────────┬────────►
   │                ┌─EQQSECP2────────────┐       │ │                ┌─EQQRECP2───────────────┐        │
   └─CP2DUMPPROC──(─┴─dump procedure name─┴──)───┘ └─CP2RESTPROC──(─┴─restore procedure name─┴──)────┘
```

```
►──┬──────────────────────────────┬──┬────────────────────────────────────────┬──────────────►
   │                ┌─NO──┐       │  │                ┌─local hostname─┐      │
   └─ENABLEFIPS──(──┴─YES─┴──)────┘  └─HOSTNAME──(──┼─hostname───────┼──)──┘
                                                      └─IP address─────┘

►──┬─────────────────────────────────┬──┬─────────────────────────────────────────┬──────────►
   │               ┌─60──────┐       │  │                    ┌─426────────┐       │
   └─KEEPALIVE──(──┴─seconds─┴──)────┘  └─LOCPORTNUMBER──(──┴─TCPIP port─┴──)───┘

►──┬─────────────────────────────────────────────────┬──┬─────────────────────────────────────────────────┬──►
   │                  ┌─EQQSENLT────────────┐        │  │                 ┌─EQQRESLT───────────────┐      │
   └─LTPDUMPPROC──(──┴─dump procedure name─┴──)─────┘  └─LTPRESTPROC──(──┴─restore procedure name─┴──)──┘

►──┬─────────────────────────────────────────────────┬──┬─────────────────────────────────────────────────┬──►
   │                  ┌─EQQSENCP────────────┐        │  │                 ┌─EQQRENCP───────────────┐      │
   └─NCPDUMPPROC──(──┴─dump procedure name─┴──)─────┘  └─NCPRESTPROC──(──┴─restore procedure name─┴──)──┘

►──┬──────────────────────────────────────────────┬──┬───────────────────────────────────────────┬──────►
   │                  ┌─511──────────────┐        │  │                 ┌─512─────────────┐       │
   └─PEERHTPPORT──(──┴─HTTP port number─┴──)─────┘  └─PEERHTSPORT──(──┴─SSL port number─┴──)───┘

►──┬──────────────────────────────────────────┬──┬──────────────────────────────────────────┬──────────►
   └─PEERHOSTNAME──(──┬─hostname───┬──)──────┘  │                     ┌─426────────┐       │
                      └─IP address─┘             └─PEERPORTNUMBER──(──┴─TCPIP port─┴──)───┘

►──┬───────────────────────────────────────┬──┬──────────────────────────────────────────┬─────────────►
   │                  ┌─CAONLY─┐           │  │                     ┌─tws────────┐       │
   └─SSLAUTHMODE──(──┴─STRING─┴──)────────┘  └─SSLAUTHSTRING──(──┴─SSL string─┴──)───┘

►──┬─────────────────────────────────────────────────────┬─────────────────────────────────────────────►
   └─SSLKEYSTORE──(────SSL keystore db file name────)───┘

►──┬──────────────────────────────────────────────────────────┬──┬─────────────────────────────┬───────►
   └─SSLKEYSTOREPSW──(────SSL keystore pw file name────)─────┘  │              ┌─OFF───┐      │
                                                                 └─SSLLEVEL──(──┴─FORCE─┴──)──┘

►──┬────────────────────────────────────────────────┬──────────────────────────────────────────────────►◄
   │                   ┌─TCPIP─────────────────┐     │
   └─TCPIPJOBNAME──(──┴─TCPIP started task─┴──)────┘
```

Parameters

CHECKROLE(NO|YES)

Determines if the controller configured as primary (with **OPCHOST(YES)** set in the OPCOPTS statement) performs a check about the backup controller activity, before starting.

The primary controller tries to connect to the backup controller host name and port number as a TCP/IP client. If the connection is established, it means that the backup controller is running as the primary controller after a takeover occurred. In this case, the primary controller starts as the backup controller.

If the connection is not established, it means that the backup controller is not running as the primary controller, therefore the primary controller continues its normal startup processing.

CONNTIMEOUT(*TCPIP timeout interval*|60)

It defines how many seconds a TCP/IP connection attempt waits before a timeout occurs. It is expressed in seconds. Valid values are from 1 to 10000. The default is 60.

CP1DUMPPROC(*dump procedure name***|EQQSECP1)**
: The name of the dump procedure that you set with EQQJOBS for the CX, CP1, and XD1 files. The default is **EQQSECP1**.

CP1RESTPROC(*restore procedure name***|EQQRECP1)**
: The name of the restore procedure that you set with EQQJOBS for the CX, CP1, and XD1 files. The default is **EQQRECP1**.

CP2DUMPPROC(*dump procedure name***|EQQSECP2)**
: The name of the dump procedure that you set with EQQJOBS for the CX, CP2, and XD2 files. The default is **EQQSECP2**.

CP2RESTPROC(*restore procedure name***|EQQRECP2)**
: The name of the restore procedure that you set with EQQJOBS for the CX, CP2, and XD2 files. The default is **EQQRECP2**.

ENABLEFIPS(NO|YES)
: Indicates whether the SSL communication must comply with FIPS standards. Specify YES to have a FIPS compliant SSL communication. This keyword is ignored if the SSL communication is not enabled.

For more information about how you activate the support for FIPS standard, see *IBM Workload Scheduler for z/OS: Planning and Installation*.

HOSTNAME('*hostname***|***IP address***|** ***local hostname***')**
: The local host name or IP address used by the TCP/IP communication subtasks. The default is the IP address returned by TCP/IP. It can be up to 52 alphanumeric characters and specifies a host name or IP address in IPv4 or IPv6 format. Enclose this value in single quotation marks.

Omitting this parameter might affect how long the server initialization process takes. TCP/IP must free resources used by previously opened connections. Before doing this, it waits the time specified in the TCP/IP profile, through the FINWait2time parameter of the TCPCONFIG statement. When this time limit is reached, the system waits a further 75 seconds before dropping the connection. The default is 600 seconds, but you can specify a lower value. For details about the TCPCONFIG statement, see *z/OS Communication Server IP Configuration Reference*.

KEEPALIVE(seconds|60)

Sets the TCP_KeepAlive socket option for the current connection between the primary controller and backup controller. This allows the backup controller to be notified in a timely manner when an unexpected system failure occurs on the primary controller site.

It is expressed in seconds. Valid values are from 1 to 2147460. The default is 60.

For detailed information about the TCP_KeepAlive socket option, see *z/OS Communications Server: IP Programmer's Guide and Reference*.

LOCPORTNUMBER(TCPIP port|426)
: The local TCP/IP port number used by the TCP/IP communication subtask of the controller or backup controller to communicate with each other. Valid values are from 0 to 65535. The default is 426.

LTPDUMPPROC(dump procedure name|EQQSENLT)
: The name of the dump procedure that you set with EQQJOBS for the LTP file. The default is **EQQSENLT**.

LTPRESTPROC(restore procedure name|EQQRESLT)
: The name of the restore procedure that you set with EQQJOBS for the LTP file. The default is **EQQRESLT**.

NCPDUMPPROC(dump procedure name|EQQSENCP)
The name of the dump procedure that you set with EQQJOBS for the NCP, NCX, and NXD files. The default is **EQQSENCP**.

NCPRESTPROC(restore procedure name|EQQRENCP)
The name of the restore procedure that you set with EQQJOBS for the NCP, NCX, and NXD files. The default is **EQQRENP**.

PEERHOSTNAME('*hostname*|*IP address*')
The host name or IP address of the partner controller. It can be up to 52 alphanumeric characters and specifies a host name or IP address in IPv4 or IPv6 format. Enclose this value in single quotation marks. This parameter is required.

PEERHTPPORT(HTTP port number|511)
The port number used by the HTTP server to listen for non-SSL connections. This parameter is used by the backup controller after you issue the modify command `/f subsys,BKTAKEOVER`. Valid values are from 0 to 65535.

PEERHTSPORT(SSL port number|512)
The SSL port number used by the HTTP server to listen for SSL-connections. This parameter is used by the backup controller after you issue the modify command `/f subsys,BKTAKEOVER`. Valid values are from 0 to 65535.

PEERPORTNUMBER(TCPIP port|426)
The TCP/IP port number of the partner controller. Valid values are from 0 to 65535. The default is 426.

SSLAUTHMODE(STRING|CAONLY)
The SSL authentication type. Specify one of the following values:

CAONLY
The controller checks the certificate validity by verifying that a recognized Certification Authority has issued the peer certificate. Information contained in the certificate is not checked. This is the default value.

STRING
The controller checks the certificate validity as described in the CAONLY option. It also verifies that the Common Name (CN) of the Certificate Subject matches the string specified in the SSLAUTHSTRING parameter.

To avoid any communication error, specify the same SSLLEVEL value for the scheduler started tasks that are to communicate with each other.

SSLAUTHSTRING(SSL string|tws)
The string used to verify the certificate validity when you set SSLAUTHMODE to STRING. The string is up to 64 characters. The default is tws.

SSLKEYSTORE(*SSL keystore db filename*)
Identifies the database containing keys and certificates. It consists of an SSL working directory name and file name, in the format SSLworkdir/TWS.kbd. It is case sensitive. This field is required if the SSLLEVEL parameter is set to FORCE.

SSLKEYSTOREPSW(*SSL keystore pw filename*)
Identifies the file containing the key password. It consists of an SSL working directory name and file name, in the format SSLworkdir/TWS.sth. It is case sensitive. This field is required if the SSLLEVEL parameter is set to FORCE.

SSLLEVEL(FORCE|OFF)
The SSL authentication type. Specify one of the following values:

OFF The scheduler component does not support SSL authentication for its connections. This is the default value.

FORCE
The scheduler component uses SSL authentication for all its connections. It refuses any incoming connection that is not SSL.

To avoid communication errors, specify the same SSLLEVEL value for the scheduler started tasks that are to communicate with each other.

TCPIPJOBNAME(*TCPIP started task*|TCPIP)
The name of the TCP/IP started task running on the z/OS system where you run the controller. Set this parameter when you have multiple TCP/IP stacks or a TCP/IP started task with a name different from TCPIP.

Examples

```
BKPTOPTS TCPIPJOBNAME('TCPIP')          1
         HOSTNAME('1.11.111.111')       2
         LOCPORTNUMBER(33014)           3
         PEERHOSTNAME('2.22.222.222')   4
         PEERPORTNUMBER(33014)          5
```

In this example, the BKPTOPTS statement is set on the backup controller, with the following meanings:

1	The TCP/IP started task name set to the default value.
2	The IP address of the backup controller.
3	The number of the local port to communicate with the primary controller.
4	The IP address of the primary controller.
5	The number of the port of the primary controller.

CPUREC

Purpose

The CPUREC statement begins an IBM Workload Scheduler workstation (CPU) definition. You must specify one CPUREC for each IBM Workload Scheduler workstation you intend to add to the end-to-end with fault tolerance capabilities network, with the exception of the controller that acts as master domain manager of the network.

CPUREC is defined in the member of the EQQPARM library, as specified by the TPLGYMEM keyword in the TOPOLOGY statement.

For a detailed description of this statement, see *IBM Workload Scheduler for z/OS: Scheduling End-to-end with Fault Tolerance Capabilities*.

DBCSOPTS

Purpose

The DBCSOPTS statement defines the format options for the IBM Workload Scheduler for z/OS Japanese language feature. You can specify this statement for a controller or standby controller.

For the online environment, DBCSOPTS appears in the same member of the EQQPARM library as the statement. For the batch environment, it appears in the same parameter library member as the BATCHOPT statement.

Important: Use the same DBCSOPT statement definition for both online and batch initialization.

Format

```
►►─DBCSOPTS─┬──────────────────────────────┬─┬───────────────────────────────┬──────────►
            │               ┌─EBCDIC─┐      │ │                ┌─EBCDIC─┐      │
            └─APPLID──(──┴─DBCS───┴──)─┘ └─OWNERID──(──┴─DBCS───┴──)─┘

►─┬───────────────────────────────┬──────────────────────────────────────────────────►◄
  │                ┌─KS─┐         │
  └─SORTORDER──(──┴─KR─┴──)─┘
```

Parameters

APPLID(DBCS|EBCDIC)
: Defines the format for storing the application ID field and group definition ID field in the IBM Workload Scheduler for z/OS database. Specify DBCS if these fields are stored in DBCS bracketed format. Specify EBCDIC if these fields are stored in EBCDIC format.

  Note: You cannot use the Job Description dialog if you specify DBCS, because the job name cannot be specified in DBCS format.

OWNERID(DBCS|EBCDIC)
: Defines the format for storing the owner ID field in the IBM Workload Scheduler for z/OS database. Specify DBCS if the owner ID is stored in DBCS bracketed format. Specify EBCDIC if it is stored in EBCDIC format.

SORTORDER(KR|KS)
: Defines the DBCS order to be used when sorting fields containing DBCS data. You can specify one of the following types:

KR	Kanji basic radical stroke-count
KS	Kanji basic total stroke-count.

Examples

```
DBCSOPTS APPLID(DBCS)    1
         SORTORDER(KR)   2
```

In this example of a DBCSOPTS statement:

1 Application IDs and group definition IDs are stored in DBCS bracketed format.

2 DBCS fields are sorted according to Kanji basic radical stroke-count in IBM Workload Scheduler for z/OS reports.

DBOPT

Purpose

Use this statement to support Tivoli Dynamic Workload Console reporting. It provides the information required to connect to the database and store data. The information is used by the Java program that populates the history tables and by the server that provides the URL information to Tivoli Dynamic Workload Console process.

Format

```
►►─DBOPT──┬──────────────────────────────────┬──────────────────────────────────────►
          │                   ┌─10───┐       │
          └─CLEANUPPOLICY──(──┴─days─┴──)────┘

►─┬───────────────────────────────────────────┬─┬───────────────────────┬───────────►
  │               ┌─IBM - 037────────────┐    │ └─DBPSW──(──password──)─┘
  └─CODEPAGE──(───┴─host system codepage─┴──)─┘

►─┬────────────────────┬─┬────────────────────────┬─────────────────────────────────►
  └─DBURL──(──url──)───┘ └─DBUSER──(──userid──)───┘

►─┬───────────────────────────────────────────────────────────────────┬─────────────►
  │                 ┌─NO──┐                                            │
  └─EXTENDEDARC──(──┴─YES─┴──┬───────────────────────────────────┬──)──┘
                             │                    ┌─150─┐        │
                             └─LONGDURPOLICY──(───┴─nnn─┴───)────┘

►─┬─────────────────────────────────────┬─┬───────────────────────────────────┬────►
  └─SMOOTHPOLICY──(──percentage──)──────┘ └─TIMEZONE──(──timezone_string──)───┘

►─┬──────────────────────────────────┬─┬──────────────────────────────────────┬───►◄
  │                 ┌─0─────┐        │ └─WRKDIR──(──working directory──)──────┘
  └─TRACELEVEL──(───┴─level─┴──)─────┘
```

Parameters

CLEANUPPOLICY (*days*|10)
The number of days for which the historical run data is kept in the database. Data older than the number of days specified is deleted. The default value is 10 days.

CODEPAGE (*host system codepage*|IBM–037)
The name of the host code page, used by the archiving process. You can provide the IBM–*nnn* value, where *nnn* is the EBCDIC code page. The default value, IBM–037, defines the EBCDIC code page for US English, Portuguese, and Canadian French. The following list shows the EBCDIC code pages:

IBM–939
Japan Extended

IBM–937
Taiwan

IBM–935
China

IBM–933	Korea
IBM–975	Greece
IBM–971	Iceland
IBM–970	Latin 2
IBM–838	Thai
IBM–500	International
IBM–424	Israel
IBM–297	France
IBM–285	UK
IBM–284	Spain - Latin America
IBM–280	Italy
IBM–278	Sweden - Finland
IBM–277	Denmark - Norway
IBM–274	Belgium
IBM–273	Germany
IBM–1388	China
IBM–1122	Estonia
IBM–1112	Baltic
IBM–1047	Open Systems
IBM–1026	Latin 5 (Turkey)
IBM–1025	Cyrillic

The following list shows the EBCDIC code pages for EURO support:

IBM–1140	Finland, Sweden
IBM–1141	Austria, Germany
IBM–1142	Denmark, Norway
IBM–1143	USA
IBM–1144	Italy
IBM–1145	Spain, spanish-speaking Latin America

IBM–1146
UK

IBM–1147
France

IBM–1148
Belgium, Switzerland

DBPSW (*password*)

The password associated with the user set in DBUSER. To avoid leaving the password in plaintext, you can have it encrypted. For details about how to encrypt the password, see *Managing the Workload*.

The password can be long up to 15 characters. This limit ensures that the password will continue to fit on one line after it is encrypted.

This parameter is required and does not have a default value.

DBURL (*url*)

Information about the URL, in the following format:

```
jdbc:db2://server:port/database
```

Ensure that `jdbc:db2` is written in lowercase format, and replace the variables as follows:

server TCP/IP address or host name of the system where the database resides.

port SQL port number used by the DB2 server.

database
Name of the target database.

This parameter is required and does not have a default value.

DBUSER (*userid*)
The user ID to access the database. This parameter is required and does not have a default value.

| **EXTENDEDARC (YES|NO)**
| Specifies the type of occurrences that are saved by the archiving process.

| If you specify YES, the archiving process stores the occurrences belonging to
| the old current plans and the completed operations, either successful or in
| error, that still belong to the current plan. For example, the completed
| operations related to started occurrences are processed to be saved in the DB2
| tables.

| If you specify NO, the archiving routines save only the operations belonging to
| the occurrences that are no longer in the current plan. All the other entries are
| ignored.

| **Note:** When the archiving routines process a job, they do not process it again
| at the next archiving run. This means that if you set EXTENDEDARC(NO), the
| occurrences that were not stored during the current run will not be stored
| during the next run, even if you change the setting to EXTENDEARC(YES).

LONGDURPOLICY (*nnn*|150)
The policy used to decide if a job is a long duration job, based on the formula: `AD >= (ED * LDP / 100)` where:

AD Actual duration.

ED	Estimated duration.
LDP	Long duration policy.

Specify a value in the range 100 to 999. The default value is 150.

SMOOTHPOLICY (*percentage*)

The policy used to calculate the average duration of a job. It sets the weighting factor that favors the most recent job run, when calculating the average duration for a job. This is expressed as a percentage. For example, a value of 40 applies a weighting factor of 40% to the most recent job run, and 60% to the existing average. If you do not set this parameter, the smoothing is not used and the average duration is calculated as the total elapsed time divided by the number of successful runs.

TIMEZONE (*timezone_string*)

The local time zone of the z/OS system where the controller runs, for example `TIMEZONE('Europe/Rome')`. It is the format used in standard `zoneinfo` files at distributed side. Do not specify the time zone in a three-letter format, because this results in an incorrect time zone from Java APIs during daylight saving time.

This parameter is required and does not have a default value.

TRACELEVEL (*level*|0)

Trace level for internal logging and traces. Possible values are:

0	To receive error messages only.
1	To receive error and warning messages.
2	To receive error, warning, and informational messages.
3	Indicates the *fine* level, to receive the most important messages with the lowest volume.
4	Indicates the *finer* level, to activate entry and exit traces.
5	Indicates the *finest* level, to receive the most detailed tracing output.

The default value is 0.

WRKDIR (*working directory*)

The complete path of the working directory for the archiving process. Each subsystem must have its own working directory. You can use the same working directory used for end-to-end scheduling with fault tolerance capabilities. This parameter is required and does not have a default value.

Run EQQPCS08 to customize the content of the working directory.

This parameter is required and does not have a default value.

DOMREC

Purpose

The DOMREC statement begins a domain definition. You must specify one `DOMREC` for each IBM Workload Scheduler domain you intend to add to the end-to-end with fault tolerance capabilities network, with the exception of the master domain. The domain name used for the master domain is `MASTERDM`. The master domain consists of the controller, which acts as the master domain manager, and of any fault-tolerant and standard agents that are directly attached to it.

DOMREC is defined in the member of the EQQPARM library, as specified by the TPLGYMEM keyword in the TOPOLOGY statement.

For a detailed description of this statement, see *IBM Workload Scheduler for z/OS: Scheduling End-to-end with Fault Tolerance Capabilities*.

DSTOPTS

Purpose

The DSTOPTS statement defines the run time options for the data store.

DSTOPTS is defined in the member of the EQQPARM library as specified by the DD statement in the JCL.

Format

```
►►──DSTOPTS──┬───────────────────────────────────────────┬──────────────────────────►
             │                ┌─0─────────────┐          │
             └─CINTERVAL──(───┴─interval time─┴───)──────┘

►──┬─────────────────────────────────┬──┬────────────────────────────┬──────────────►
   │            ┌─EQQCLNPA────┐      │  │               ┌─1──┐       │
   └─CLNPARM──(─┴─member name─┴──)───┘  └─DELAYTIME──(──┴─nn─┴──)────┘

►──┬─────────────────────────────────────────────────────────────────┬─────────────►
   ├─DSTLUNAM(LU Name) CTLLUNAM(LU Name)─────────────────────────────┤
   ├─DSTGROUP(XCFGroupName) CTLMEM(XCFMemName) DSTMEM(XCFMemName)─┤
   └─CTLHOSTNAME──(──┬─hostname───┬──)──CTLPORTNUMBER(TCPIP port)─┘
                     └─IP address─┘

►──┬────────────────────────────────────┬──┬───────────────────────────────┬───────►
   │            ┌─FAILDEST────┐         │  │               ┌─JOBNAME─┐     │
   └─FAILDEST──(┴─destination─┴───)─────┘  └─HDRJOBNAME──(─┴─value───┴──)──┘

►──┬────────────────────────────────────┬──┬──────────────────────────────────┬────►
   │               ┌─STEPNAME─┐         │  │                ┌─PROCSTEP─┐      │
   └─HDRSTEPNAME──(┴─value────┴──)──────┘  └─HDRPROCNAME──(─┴─value────┴──)───┘

►──┬───────────────────────────────────┬──┬────────────────────────────────────┬───►
   │                ┌─13,21──┐         │  │                   ┌─22,30──┐       │
   └─HDRJOBLENGTH──(┴─length─┴──)──────┘  └─HDRSTEPLENGTH──(──┴─length─┴──)────┘

►──┬────────────────────────────────────┬──┬─────────────────────────────────────┬─►
   │                 ┌─31,39──┐         │  │                      ┌─120────┐     │
   └─HDRPROCLENGTH──(┴─length─┴──)──────┘  └─HDRSTEPNOLENGTH──(───┴─length─┴──)──┘

►──┬─────────────────────────┬──┬───────────────────────────┬──────────────────────►
   └─HOSTCON──(──┬─SNA─┬──)──┘  │             ┌─0─────┐     │
                 ├─XCF─┤        └─MAXSTOL──(──┴─nnnnn─┴──)──┘
                 └─TCP─┘
```

```
►──┬──────────────────────────────────┬──┬────────────────────────────┬──────────────►
   │                ┌─4096─┐          │  │            ┌─0────┐        │
   └─MAXMVSPAGES──(─┴─nnnn─┴──)─────┘  └─MAXSYSL──(─┴─nnnn─┴──)─────┘

►──┬──────────────────────────────────┬──┬────────────────────────────┬──────────────►
   │               ┌─4096──┐          │  │            ┌─1──┐          │
   └─MAXUNPAGES──(─┴─nnnnn─┴──)─────┘  └─NWRITER──(─┴─nn─┴──)───────┘

►──┬────────────────────────────────┬──┬────────────────────────────┬────────────────►
   │             ┌─15────┐          │  │                 ┌─1──┐     │
   └─QTIMEOUT──(─┴─nnnnn─┴──)─────┘  └─RETRYCOUNTER──(─┴─nn─┴──)──┘

►──┬──────────────────────────────────┬──┬──────────────────────────────────────────┬─►
   │                 ┌─NO──┐          │  │                      ┌─IMMEDIATE─┐       │
   └─SMSMODDELETE──(─┴─YES─┴──)─────┘  └─STORESTRUCMETHOD──(─┴─DELAYED───┴──)────┘

►──┬──────────────────────────────┬──┬───────────────────────────────────┬───────────►
   │            ┌─NO──┐           │  │            ┌─OPC─────────┐        │
   └─STOUNSD──(─┴─YES─┴──)──────┘  └─SYSDEST──(─┴─destination─┴──)───┘

►──┬────────────────────────────────┬────────────────────────────────────────────────►◄
   │              ┌─30───┐          │
   └─WINTERVAL──(─┴─nnnn─┴──)─────┘
```

Parameters

CINTERVAL(*interval time*|<u>0</u>)
: Defines the cleanup task wake-up interval time, expressed in minutes. Valid values are from 0 to 1440. 0 is the default and means that the Data Store CleanUp subtask is started at data store initialization, but never runs.

CLNPARM(*member name*|<u>EQQCLNPA</u>)
: Defines the parameter member name that will contain the CleanUp task criteria. The default is EQQCLNPA.

  This parameter is meaningful only if CINTERVAL is greater than 0; nevertheless it is required at data store startup because the data store loads the information contained in the member only at startup time and, even if you initially set CINTERVAL(0), you might later modify this value with an operator command.

CTLLUNAM(LU Name)
: Defines the VTAM® application LU name that identifies the controller in the SNA connection between the controller and data store. It is mandatory if the SNA connection is used between the controller and data store, and must be the same as that specified in the controller FLOPTS CTLLUNAM statement.

CTLMEM(XCFMemName)
: XCF member name, identifying the controller in the XCF connection between controller and data store. It is mandatory if XCF connection is used and must be the same specified in the controller FLOPTS CTLMEM statement.

CTLHOSTNAME(*hostname*|*IP address*)
: The host name or IP address of the remote controller. Valid values are fully-qualified names, up to 52 characters.

CTLPORTNUMBER(*TCPIP port*|<u>423</u>)
: The TCP/IP port number used to communicate with the remote controller. Valid values are from 0 to 65535. The default is 423.

DELAYTIME(nn|1)
Delay time, expressed in minutes, used to decide when to discard/store incomplete sysout. Valid values are from 0 to 1440. 1 is the default.

DSTGROUP(XCFGroupName)
Defines the XCF group name that identifies the data store in the XCF connection with the controller. It is mandatory if XCF connection is used.

The XCF group defined to connect the data store to the controller must be different from the one defined in the XCFOPTS group to connect the controller to the z/OS tracker.

DSTLUNAM(LU Name)
Defines the VTAM application LU name that identifies the data store in the SNA connection between controller and data store. It is mandatory if SNA connection is used between the controller and the data store.

DSTMEM(XCFMEMName)
The XCF member name, that identifies the data store in the XCF connection between the controller and the data store. It is mandatory if XCF connection is used.

FAILDEST(*destination*|FAILDEST)
Defines a destination where the jobs that could not be archived will be requeued so that they can be examined.

If the JES2 DESTDEF statement specifies NODENAME=REQUIRED, the destination specified in the FAILDEST parameter must be defined to JES2. using the following command:

```
$ADD DESTID(xxxxxxxx),DEST=xxxxxxxx
```

where *xxxxxxxx* is the destination specified in the DSTDEST parameter.

If the JES2 DESTDEF statement specifies NODENAME=OPTIONAL, you do not need to specify a destination in the FAILDEST parameter.

HDRJOBNAME(*value*|JOBNAME)
The jobname identifier for the STEPTABLE HEADER in the JESMSGLG sysout. JOBNAME is the default value used by IBM Workload Scheduler for z/OS. You can specify any other value up to eight characters, according to the EQQACTR1 exit customization.

HDRSTEPNAME(*value*|STEPNAME)
The stepname identifier for the STEPTABLE HEADER in the JESMSGLG sysout. STEPNAME is the default value used by IBM Workload Scheduler for z/OS. You can specify any other value up to eight characters in length, according to the EQQACTR1 exit customization.

HDRPROCNAME(*value*|PROCSTEP)
The procstep identifier for the STEPTABLE HEADER into the JESMSGLG sysout. PROCSTEP is the default value used by IBM Workload Scheduler for z/OS. You can specify any other value up to eight characters in length, according to the EQQACTR1 exit customization.

HDRJOBLENGTH(*length*|21)
The start position of the jobname in the STEPTABLE. You can specify any other length according to the EQQACTR1 exit customization. The ASA character is not included.

HDRSTEPLENGHTH(*length*|30)
The start position of the STEPNAME in the STEPTABLE. The default value is

30. You can specify any other length according to the EQQACTR1 exit customization. The ASA character is not included.

HDRPROCLENGTH(*length*|39)
The start position of the PROCNAME in the STEPTABLE. The default value is 39. You can specify any other length according to the EQQACTR1 exit customization. The ASA character is not included.

HDRSTEPNOLENGTH(*length*|120)
The start position of the STEPNO in the STEPTABLE. The default value is 120. You can specify any other length according to the EQQACTR1 exit customization. The ASA character is not included.

HOSTCON(SNA|XCF|TCP)
Defines the type of connection used when transmitting data to the controller. It is mandatory (no default exists).

MAXSTOL(*nnnnn*|0)
Defines the maximum number of user sysout (not lines of sysout) the data store can store for a single job. Valid values are from 0 to 10000. If 0 is specified no user sysout is stored. 0 is the default. User sysout information is *not* required for restart and cleanup processing.

A nonzero value should only be specified if it is necessary to view user sysout data when a job log is retrieved using the "L" row command (for example if a user sysout contains information needed by operations to determine how to rerun a job). Specifying a nonzero value could cause a large amount of data to be written to the data store UDF files, so you might need to increase the size or the number of UDF files.

Note: The User Sysout field in the Restart and Cleanup Operation Details (EQQAMRCL) panel for an operation must be set to Y, otherwise the user sysout will not be written to the data store for that operation.

MAXMVSPAGES(*nnnn*|4096)
Defines the maximum size (in number of pages) allowed for the MVS™ part of a job log (JESJCLIN, JESJCL, JESMSGLG, JESYSMSG) to be managed by the process. If the MVS part of the job log exceeds this value, the job log is not stored, parser processing is not performed, and the job log is queued again to the failure destination, as set by the FAILDEST keyword.

For example, if *Tot_Rec* is the number of records in the job log, the number of pages for the job log is calculated as (*Tot_Rec**135)/4096. Valid values are from 0 to 4096. The default value is 4096.

MAXSYSL(*nnnn*|0)
Defines the maximum number of user sysout (not lines of sysout) the data store can send back to the controller. Valid values are from 0 to 10000. If 0 is specified no user sysout is sent back to the controller. 0 is the default.

User sysout information is NOT required for restart and cleanup processing. A nonzero value should only be specified if it is necessary to view user sysout data when a job log is retrieved using the "L" row command (for example if a user sysout contains information needed by operations to determine how to rerun a job).

Specifying a nonzero value could cause a large amount of data to be written to the data store UDF files, so you might need to increase the size or the number of UDF files.

MAXUNPAGES(*nnnn*|4096)
Defines the maximum size (in terms of number of pages) allowed for a job log to be stored. When the size of a job log is greater than this value, the job log is not stored. The structured data, instead, is stored regardless of the STORESTRUCMETHOD specified.

If Tot_Rec is the number of the records of a job log, the number of pages for a job log is (Tot_Rec*135) / 4096. Valid values are from 0 to 4096. The default is 4096.

If MAXUNPAGES is set too low, more CPU is used by the data store, because a large number of job logs will be parsed. If MAXUNPAGES is set too high, a lot of time might be spent writing large job logs.

NWRITER(*nn*|1)
Defines the number of writer tasks to be activated by the data store. Valid values are from 1 to 16. 1 is the default. The number of the writer tasks must be less than or equal to the number of the data store VSAM files. For details, see Chapter 8, “Using the data store,” on page 307.

RETRYCOUNTER(*nn*|1)
Defines the maximum number of times that data store will retry job archival within the specified delay time, before it removes the job from the reserved queue. This number corresponds to the maximum number of 'archiving in progress' messages that will be issued before the job is removed from the reserved queue and requeued to the failure destination (the one specified with the FAILDEST parameter). As troublemaking jobs can cause performance degradation to the data store, you are recommended to use low values or the default (1).

SMSMODDELTE(YES|NO)
This parameter is valid only for the Cleanup actions that EQQCLEAN performs for SMS data sets allocated with DISP=(MOD, CATLG, CATLG) in the previous run of the job. If the SMS data set existed before the previous run of the job, it is not deleted, regardless of the value you set in this parameter.

Specify NO to prevent EQQCLEAN from deleting the SMS data sets that were created in the previous run of the job. Specify YES to have EQQCLEAN delete the SMS data sets at restart. The default value is NO.

QTIMEOUT(*nnnnn*|15)
Defines the maximum time that the data store can take to satisfy a job log retrieval request issued from the controller when the sysout is not yet stored in the VSAM database, because the archiving process for that job log is in progress. If this QTIMEOUT time expires and the request has not been satisfied, then message EQQFSR5I is issued. After this message, there is no other attempt to get the job log. Therefore, you must issue another request to receive that job log. It is expressed in seconds and the allowed values are from 1 to 10000. 15 is the default.

STORESTRUCMETHOD(DELAYED|IMMEDIATE)
Defines when to archive the structured data.

If you specify IMMEDIATE, data store immediately parses and archives the job logs found at its reserved destination. This means that structured data is always generated for all the job logs handled by data store.

If you specify DELAYED, data store immediately archives the job logs found on its reserved destination, but parses and then archives the created structured information using the unstructured job log previously archived.

When you specify DELAYED, the structured data is stored even for job logs bigger than the value set for MAXUNPAGES or bigger than 4096 pages, if MAXUNPAGES is not specified.

STOUNSD(YES|NO)
The default value is NO. Select YES to store unstructured data in the data store database and to enable the job log retrieval function for the z/OS job log.

Note: If you set STORESTRUCMETHOD to DELAYED you must also set STOUNSD to YES. In this way the method for archiving can extract the structured data from the job log.

SYSDEST(OPC|*destination*)
Defines the destination used to capture the sysout from the job entry subsystem. It must be equal to the value of the RCLOPTS DSTDEST parameter.

If the JES2 DESTDEF statement specifies NODENAME=REQUIRED, the destination specified in the SYSDEST parameter must be defined to JES2. using the following command:

```
$ADD DESTID(xxxxxxxx),DEST=xxxxxxxx
```

where *xxxxxxxx* is the destination specified in the DSTDEST parameter.

If the JES2 DESTDEF statement specifies NODENAME=OPTIONAL, you do not need to specify a destination in the SYSDEST parameter.

WINTERVAL(*nnnn*|30)
Defines the maximum time interval that can occur before the data store scans again the reserved destination. To better explain how this value is used, consider that:

- JESQUEUE is the task that reads the JES spool to get the job logs to be handled by data store using as filter criterion the "reserved destination"
- From this reading, the JESQUEUE builds a linked list of JOBIDs to be handled (the maximum number of JOBIDs read from the JES spool each time is 3N+1 where N is the number of writers tasks defined)
- Once this linked list has been built, the JESQUEUE task checks it until it is emptied by the WRITERS tasks
- At this point, before rebuilding the linked list again, JESQUEUE task waits the time interval specified into WINTERVAL.

It is expressed in seconds and the allowed values are from 0 to 3600. The default is 30.

DSTUTIL

Purpose

The DSTUTIL statement defines the possible maintenance actions that can be done on the data store database, and are usually invoked by a batch program.

The maintenance cleanup actions (DELUNSTR/DELSTRUC/DELBOTH) can be done by means of the cleanup data store task (if specified in the member identified by the CLNPARM statement).

Format

```
►►──┬─DELSTRUC─┬──┬──────────────┬──DELBOTH──┬─────────────────────────────┬──────────►
    └─DELUNSTR─┘  └─DDNAME(ddn)──┘           └─DDNAME──(────ddn1,ddn2────)─┘

►──┬─EXPSTRUC─┬──DDNAME(ddn)──EXPBOTH──┬─────────────────────────────┬──IMPORT───────►
   └─EXPUNSTR─┘                        └─DDNAME──(────ddn1,ddn2────)─┘

►─DDNAME(ddn)──REPLACE ( YES|NO)──RECOVER (primary)──┬──────────────────┬────────────►
                                                     └─SEARCH1(value)───┘

►──┬─────────────────┬──┬─────────────────┬──┬─────────────────┬─────────────────────►
   └─SEARCH2(value)──┘  └─SEARCH3(value)──┘  └─SEARCH4(value)──┘

►──┬─────────────────┬──┬─────────────────┬──┬─────────────────┬─────────────────────►
   └─SEARCH5(value)──┘  └─SEARCH6(value)──┘  └─SEARCH7(value)──┘

►──┬─────────────────┬──┬─────────────────┬──┬──────────────────┬────────────────────►
   └─SEARCH8(value)──┘  └─SEARCH9(value)──┘  └─SEARCH10(value)──┘

►──┬──────────────────┬──────────────────────────────────────────────────────────────►◄
   └─REPLACE(YES|NO)──┘
```

Parameters

DELSTRUC|DELUNSTR|DELBOTH

Defines the criteria for deleting sysout from the data store database. It is possible to specify whether structured data (DELSTRUC) or unstructured data (DELUNSTR) are to be deleted. You can require to delete both types specifying DELBOTH.

The keyword DDNAME, if coded, identifies the DD name of a data set where the records will be copied before they are deleted from the database. When DELBOTH is coded, you must specify two DDnames, for the two different types of data: the first one is for the unstructured data, the second one is for the structured data).

The copy on a data set is allowed only in batch mode.

The SEARCH*nn* keyword (where *nn* can be a value from 1 to 10) identifies the criteria selected. The string *value* has the following format:

CodeNameComparisonOperFieldName

CodeName can assume the following values:

JBNM	Job name
JBDT	Date
JBTM	Time
JBID	Job ID
DSID	data set ID
STNM	Step name
PSNM	Procedure step name
DDNA	DD name
SYCL	SYSOUT class
OLDR	Older than

ComparisonOper can assume the following values:

EQ	Equal
NE	Not equal

GT	Greater than
GE	Greater or equal
LT	Less than
LE	Less or equal
LK	Like
UK	Unlike
MM	Month, used with OLDR codename only
DD	Days, used with OLDR codename only
HR	Hours, used with OLDR codename only
MN	Minutes, used with OLDR codename only

`FieldName` indicates the field where the selection is performed. For example, it can be a specific jobname if the JBNM codename has been used (JBNM EQ myjob causes all the job logs having the jobname equal to myjob to be deleted. For the LK operator, it can contain the wildcard * or ?).

Note: All dates must be specified in the format *yyyymmdd*. For example, the 22nd of July 2002 is coded as 20020722.
All the terms inside the SEARCHn key are logical ANDs, so they must be all true to select the sysout for the operation. The different search criteria (SEARCH1 to SEARCH10) are logical ORs, so at least one must be true to run the command.

`EXPSTRUC|EXPUNSTR|EXPBOTH`

Defines the criteria for exporting sysout records from the data store database. It's possible to specify whether structured data (EXPSTRUC) or unstructured data (EXPUNSTR) are to be exported.

The keyword DDname is always required and identifies the DD name of a data set where the records will be exported. You can require to export both types specifying EXPBOTH: in this case, you must specify two different ddnames as values in the DDNAME clause. The first one is for the unstructured data, the second one is for the structured data).

The keywords used in the same clause have the same meaning specified for the parameters DELSTRUC/DELUNSTR/DELBOTH.

You can also specify selection criteria, coding the SEARCHnn keyword.

`IMPORT`

Defines the criteria for importing sysout records (previously stored in a sequential data set) into the data store database.

The keywords have the same meaning as those specified for the previous parameters. In addition, the REPLACE keyword specifies whether matching sysout records in the data store data base should be overwritten. The default is NO. One data set at a time can be imported (only one DDname can be coded). If both structured and unstructured data is to be imported, you must codify the parameter IMPORT twice, to process separately the structured and the unstructured export files. IMPORT can be used by the batch program only when data store is not operating.

You can also specify selection criteria, coding the SEARCHnn keyword.

`RECOVER(PRIMARY)`

Indicates that the batch program extracts a data set of keys for all sysout records stored in the data store database, to enable the recovery of a corrupted primary key data set. The EQQPKREC DD statement in the batch JCL identifies the name of the extracted data set. It can be used by the batch program only when the data store is not operating.

In this case, you cannot use SEARCHnn keyword to specify selection criteria, because it is not accepted.

ERDROPTS

Purpose

The ERDROPTS statement defines run-time options to an event-reader task. You can specify this statement for a tracker, controller, or standby controller. This statement is required if you do not specify ERDRTASK(0) on the OPCOPTS statement.

ERDROPTS is defined in the member of the EQQPARM library as specified by the ERDRPARM keyword on the OPCOPTS statement.

Format

```
►►──ERDROPTS──ERSEQNO──(────event reader sequence number────)──────────────────►

►─┬─────────────────────────────────────┬────────────────────────────────────────►
  │              ┌─10──────────┐        │
  └─ERWAIT──(────┴─wait limit──┴───)────┘

►─┬────────────────────────────────────────────────────────────┬─────────────────►◄
  └─RELDDNAME──(────ddname of submit/release data set────)─────┘
```

Parameters

ERSEQNO(*event reader sequence number***)**
Defines which event reader is being started. This number must be in the range 1 to 16 and defines the ddname of the input event data set.

The ddname of the corresponding input event data set has the format EQQEVD*nn*, where *nn* is the event-reader sequence number specified as a 2-digit number. If the sequence number is less than 10, you must add a leading zero. For example, if you specify ERSEQNO(1), the ddname should be EQQEVD01. Ensure that this ddname is in your JCL procedure.

Note: You can start only *one* event reader for each input event data set. You must not specify the same number for ERSEQNO and the event-writer sequence number (EWSEQNO) if both tasks are active on the same system.

ERWAIT(*wait limit***|10)**
Defines how long the event reader waits (in seconds) before rechecking the input event data set after all event records have been read. It will influence the rate with which the ready list is updated.

RELDDNAME(*ddname of submit/release data set***)**
RELDDNAME is used in a shared DASD configuration, where the target system uses the hold/release function. It identifies the submit/release data set to which the controller writes release commands. RELDDNAME is required if events are not created on the same system that the event reader is started on. The ddname is the same name that you specify in the workstation destination field and the DASD keyword of ROUTOPTS.

Examples

```
ERDROPTS ERSEQNO(1)   1
         ERWAIT(5)    2
```

In this example of an ERDROPTS statement:

1 Reader number 1 is started.

2 The reader will wait 5 seconds before rechecking its input event data set.

EWTROPTS

Purpose

The EWTROPTS statement defines run-time options for the event-writer task. EWTROPTS is required when you specify OPCOPTS EWTRTASK(YES). This statement is used by a tracker.

EWTROPTS is defined in the member of the EQQPARM library as specified by the EWTRPARM keyword of the OPCOPTS statement.

Format

```
        (1)
►►────────EWTROPTS──┬───────────────────────────────────────────────────┬────────►
                    └─EWSEQNO──(───event data set sequence number───)─┘

►─┬───────────────────────────────────┬─┬─────────────────────────────┬──────────►
  │        (2)           ┌─10─────────┐ │ │             ┌─NO───┐        │
  └─EWWAIT───────(───┴─wait limit─┴──)─┘ └─HOLDJOB──(───┼─USER─┼───)─┘
                                                       └─YES──┘

►─┬──────────────────────────────────┬─┬──────────────────────────────────┬──────►
  │                  ┌─END─┐           │ │               ┌─LAST────┐          │
  └─PRINTEVENTS──(───┼─NO──┼───)─┘ └─RETCODE──(───┴─HIGHEST─┴───)─┘
                     └─ALL─┘

►─┬──────────────────────────────────────────┬───────────────────────────────────►
  └─SDEPFILTER──(───startpos,stringvalue───)─┘

►─┬─────────────────────────────────────┬─┬──────────────────────────────────┬───►
  │          (2)          ┌─860101─┐      │ │         (2)          ┌─0000─┐      │
  └─SKIPDATE───────(───┴─yymmdd─┴───)─┘ └─SKIPTIME───────(───┴─hhmm─┴───)─┘

►─┬─────────────────────────────────┬─┬───────────────────────────────┬──────────►
  │                 ┌─ABEND─┐          │ │              ┌─NO──┐          │
  └─STEPEVENTS──(───┼─ALL───┼───)─┘ └─STEPINFO──(───┴─YES─┴───)─┘
                    └─NZERO─┘

►─┬──────────────────────────┬──────────────────────────────────────────────────►◄
  │          ┌─NO──┐          │
  └─SUREL──(───┴─YES─┴───)─┘
```

Notes:

1 If the IBM Workload Scheduler for z/OS controller has to perform NOERROR processing, in all the trackers connected to that controller RETCODE must be set to HIGHEST and STEPEVENTS must be set to either ALL or NZERO.

2 IBM Workload Scheduler for z/OS uses EWWAIT, SKIPDATE, and SKIPTIME values only if SUREL(YES) is specified.

Parameters

EWSEQNO(*event data set sequence number***)**

Starts an event writer with an event-reader function, where the tracker has a non-DASD connection with the controller. That is, where the tracker is connected by a VTAM, XCF, or TCP/IP link or where the event writer is started in the same address space as the controller. You cannot specify EWSEQNO where HOSTCON(DASD) is specified on the TRROPTS statement.

EWSEQNO eliminates the need for a separate event-reader task. The event writer writes job-tracking events to the event data set, and also forwards the events to the task handling communication with the controller. This function can enhance performance because IBM Workload Scheduler for z/OS need not write events to the event data set and then read them back again.

If the tracker cannot communicate with the controller, the event writer continues to collect events and writes them to the event data set. When the connection is restored, these events are forwarded to the controller for processing.

Valid values for EWSEQNO are from 1 to 16. If any EVENT READERS is configured to this same started task, then the value specified for EWTROPTS EWSEQNO() must be different from any ERDROPTS ERSEQNO() value for this task. If this task has no ERDROPTS statements, then the EWSEQNO entry serves only as a flag and its actual value is not important.

Note: EWSEQNO is not used to build the event-writer ddname in the tracker JCL procedure, as happens with the ERSEQNO keyword of the ERDROPTS statement for a controller. The ddname is always EQQEVDS.

EWWAIT(*wait limit***|10)**

Defines how long the event writer waits (in seconds) before rechecking the submit/release data set after all records are read. The wait-limit value is used only when SUREL(YES) has been specified.

HOLDJOB(USER|YES|NO)

The HOLDJOB keyword tells the IBM Workload Scheduler for z/OS event writer whether it should place jobs in hold status as they enter this system so that they can be released at a later time. This might be necessary if some of your IBM Workload Scheduler for z/OS-planned jobs are submitted by a non-IBM Workload Scheduler for z/OS process. To prevent such jobs from running before their predecessors are complete, they should be held as they enter the system and then released at the correct time.

Use HOLDJOB(NO) when your IBM Workload Scheduler for z/OS-planned jobs are always submitted automatically by IBM Workload Scheduler for z/OS. This is the preferred method of running jobs because it is the simplest and involves the least overhead. When you specify HOLDJOB(NO), no jobs are held and released.

Use HOLDJOB(USER) if you have IBM Workload Scheduler for z/OS-planned jobs that must be submitted by a non-IBM Workload Scheduler for z/OS process. When you specify HOLDJOB(USER), you must place all jobs submitted by non-IBM Workload Scheduler for z/OS methods in hold status. This means that all such jobs must have the TYPRUN=HOLD parameter on their job cards. These jobs are then released automatically by IBM Workload Scheduler for z/OS, as follows:

- Immediately, if the operation options specify HOLD/RELEASE=NO.
- When normal submit criteria are met, if the options of the operation specify HOLD/RELEASE=YES.

Use HOLDJOB(YES), the least preferred of the three options, only if you have IBM Workload Scheduler for z/OS-planned jobs that both:

- Cannot be submitted by IBM Workload Scheduler for z/OS.
- Do not have TYPRUN=HOLD specified on their job cards.

When you specify HOLDJOB(YES), the IBM Workload Scheduler for z/OS event-writer subtask places *all* jobs that enter the system in hold status. Some jobs are IBM Workload Scheduler for z/OS-planned and cannot run immediately; for example, if predecessors are not complete. The event writer must hold *all* jobs because it cannot determine whether a particular job is IBM Workload Scheduler for z/OS-planned or not. (It does not have access to the IBM Workload Scheduler for z/OS current plan.)

For each job that enters the system and is held, the event writer creates an event record. The IBM Workload Scheduler for z/OS event manager, which processes all event records, checks the current plan to determine which held jobs are IBM Workload Scheduler for z/OS-planned and which are not. Non-IBM Workload Scheduler for z/OS jobs are immediately released. IBM Workload Scheduler for z/OS-planned jobs remain on hold until their predecessors are complete, when they are automatically released.

The result is that:

- IBM Workload Scheduler for z/OS-planned jobs are run in the correct sequence according to the current plan.
- Non-IBM Workload Scheduler for z/OS jobs are held by the event writer and then released by the event manager when they are found to be non-IBM Workload Scheduler for z/OS jobs.

Note:

| 1. Starting from z/OS V2.2, you can hold jobs until a specific time by setting HOLDUNTL in the new JCL statement SCHEDULE. If you set both the HOLDUNTL and HOLDJOB(YES) parameters, HOLDUNTL is ignored.
2. IBM Workload Scheduler for z/OS does not use VTAM, XCF, or TCP/IP connections to transmit release commands for non-IBM Workload Scheduler for z/OS jobs. If you specify HOLDJOB(YES), release commands for held jobs that are not in the current plan are transmitted through NJE to a tracker system. The tracker must be part of the same NJE network as the controller.
3. If you specify HOLDJOB(USER), this value is valid even if the event writer subtask is not active. For HOLDJOB(YES), the default value NO is used when the event writer subtask is not active.

PRINTEVENTS(NO|ALL|END)

Defines how IBM Workload Scheduler for z/OS should create print events (type 4).

Specify NO if you do not intend to track print operations. No print events are created.

Specify ALL if print events should reflect the actual print time of operations. Time that an operation is in status I (interrupted) is not included in the total print time. IBM Workload Scheduler for z/OS creates print events when an operation is interrupted and when it has completed.

Specify END if print events should be created when SYSOUT printing has completed. END is the initial default value. Use END if you want to track print operations, but the total time need not reflect the actual print time. That is, the total time will include time that the printer was interrupted.

The initial value of PRINTEVENTS remains in effect until an IPL is performed or until you specify the keyword with a different value.

RETCODE(HIGHEST|LAST)

Defines how the event writer creates a return code for a job or started task for the job-end (3J) event record. If you specify HIGHEST, the event writer creates an event record with the highest return code of all the performed steps.

If you specify LAST, the event writer creates an event record with the return code of the last performed step. In detail:

- If all the steps fail, the operation error code is created from the abend code of the last step that failed.
- If all the steps fail, except for the last one, which flushed, the operation error code is created from the abend code of the first step that failed.

The keyword value is valid until you specify a different value and restart IBM Workload Scheduler for z/OS.

Note: With z/OS 1.13 and later the JOBRC keyword can be added in the JCLs. If JOBRC is defined in a JCL with the MAXRC or LASTRC values, the value entered for RETCODE is overwritten by the JOBREC value. If JOBRC is defined with the STEP value, it is ignored, and the value entered for RETCODE is used instead.

SDEPFILTER(*startpos*,*stringvalue*)

Defines if and how the scheduler must check the JOB card programmer name, to decide if a step-end event has to be produced. It is useful in particular if you use step-level dependency, to avoid specifying STEPEVENTS(ALL), that might affect the scheduler performance.

It consists of two elements:

startpos
: An integer from 1 to 20, which is the maximum length of JOB card programmer name.

stringvalue
: A character string with a maximum length of 20 characters.

If the programmer name contains special characters, such as an imbedded apostrophe, when specifying *startpos* consider the final form of the name and exclude control characters. For example, if the programmer name contains `'T.O''NEILL'`, specify SDEPFILTER(5,NEILL) to filter by `NEILL`.

If this keyword is omitted, the scheduler does not check the JOB card programmer name; otherwise the scheduler checks it, by using *startpos* as the start position from which to look for a match with *stringvalue*.

If a match is found, the step-end event is always produced, independently from other specified criteria, such as the STEPEVENTS value or EQQUX004 exit parameters.

SKIPDATE(*yymmdd*|860101)
Defines an upper limit on the age of records in the submit/release data set. The event writer does not use records created before the skip-limit date. The skip-limit date is specified in the format *yymmdd*, where *yy* is the year, *mm* the month, and *dd* the day. The skip-limit value is used only if you specify SUREL(YES).

SKIPTIME(*hhmm*|0000)
Defines an upper limit on the age of records in the submit/release data set. The event writer does not use records created before the skip-limit time on the skip-limit date. The skip-limit time-of-day is specified in the format *hhmm*, where *hh* is the hour in the range 00 to 23, and *mm* is the minute in the range 00 to 59. The skip-limit value is used only if you specify SUREL(YES).

STEPEVENTS(ALL|NZERO|ABEND)
The step event option defines how ending job steps are processed by IBM Workload Scheduler for z/OS.

If you specify ABEND, a step-end event is created and written to the event data set only for abending job steps.

If you specify ALL, a step-end event is created for all ending job steps. Specify this value only if you use one of the following:

- Automatic job recovery, to detect a flushed step or to restrict the recovery action to a step within a JCL procedure.
- Step-level dependency.

This value might affect the scheduler performance, therefore consider using the SDEPFILTER parameter as an alternative.

If you specify NZERO, a step-end event is created for all job steps that end with a nonzero completion code.

Note: The keyword value is valid until you specify a different value and restart IBM Workload Scheduler for z/OS.

STEPINFO(YES|NO)
Specify YES to create all the step events related to the jobs you run, and send them to the primary controller. The default is NO.

SUREL(YES|NO)
Specify YES if a controller submits jobs to this system through a submit/release data set. YES can be specified only if OPCHOST(NO) is also specified in the OPCOPTS statement (for details, see “OPCOPTS” on page 110). The submit/release data set is defined by the EQQSUDS ddname.

Examples

```
EWTROPTS EWSEQNO(1)           1
         STEPEVENTS(NZERO)    2
         RETCODE(HIGHEST)     3
         SDEPFILTER(5,NEILL)  4
```

In this example of an EWTROPTS statement:

1 The event writer is started with an event-reader task. The event writer collects event information and writes it to the event data set. It also forwards the event information to the controller. (The tracker is connected to the controller through XCF or NCF, or the tracker and controller are started in the same address space.)

2 IBM Workload Scheduler for z/OS creates a step-end event (type 3S) only for job steps that end with a nonzero completion code.

3 The event writer creates a job-end event record (3J) with the highest return code of all the performed steps.

4 The scheduler creates a step-end event only for jobs that specify `'T.O''NEILL'` in the JOB card programmer name.

EXITS

Purpose

The EXITS statement defines exit options to IBM Workload Scheduler for z/OS. It applies to IBM Workload Scheduler for z/OS exits whose names are prefixed with EQQUX0. You can use the EXITS statement to stop IBM Workload Scheduler for z/OS from attempting to load a particular exit. You can also use EXITS to change the default name of the load module. By default, the tracker attempts to load the EQQUX000, EQQUX004, EQQUX005, and EQQUX006 exits. By default, the controller attempts to load the EQQUX000, EQQUX001, EQQUX002, EQQUX003, EQQUX007, EQQUX009, EQQUX011, EQQUX013, EQQUX014 and EQQUXSAZ exits.

EXITS is defined in the member of the EQQPARM library as specified by the PARM parameter on the JCL EXEC statement.

Format

```
          ┌────────────────────────────┐
►►──EXITS─▼────────────────────────────┴──────────────────────────►
           └─CALL──nn──(─┬─YES─┬─)─┘
                         ├─NO──┤
                         └─X───┘

   ┌──────────────────────────────────┐
►──▼──────────────────────────────────┴───────────────────────────►◄
    └─LOAD──nn──(─┬─EQQUX0──nn───┬─)─┘
                  └─module name──┘
```

Parameters

CALLnn(NO|X|YES)
Specifies whether an exit should be loaded. The exit is either the IBM Workload Scheduler for z/OS exit with suffix *nn*, or its alternative as specified by the LOAD *nn* keyword. The value 'X' allows EQQUX007 to see the extended status 'X' (waiting for special resource). If you specify this value with the CALL07 statement, exit EQQUX007 is called and every time an operation changes its status, IBM Workload Scheduler for z/OS scans the chain of special resources to see if an operation is waiting for any of them. Use this feature only when strictly necessary so as not to decrease performance.

Use the CALL12 statement to disable the loading of System Automation exit EQQUXSAZ. Consider, however, that by setting CALL12(NO) you will not be able to use any System Automation functions.

LOAD*nn***(***module name***|**<u>**EQQUX0**</u>*nn***)**
Specifies an alternative load module, which is called instead of the default exit named EQQUX0*nn*.

Examples

```
EXITS CALL04(NO)        1
      LOAD00(TWS00)  2
```

In this example of an EXITS statement, which applies to a tracker:

1 The EQQUX004 exit is not loaded.

2 A module called TWS00 is loaded instead of exit EQQUX000.

The subsystem attempts to load exits EQQUX005 and EQQUX006 by default.

For more information about IBM Workload Scheduler for z/OS exits, see Chapter 4, "IBM Workload Scheduler for z/OS exits," on page 215.

FLOPTS

Purpose

The FLOPTS statement defines the options for Fetch Job Log (FL) task.

A controller uses this statement when OPCOPTS RCLEANUP (YES) is specified.

Note: If you changed the destination name in SNADEST, XCFDEST, or TCPDEST parameters at migration phase, specify both the old and new destination names, according to the tracker and data store considerations in the **Migrating** section of the *Planning and Installation Guide*.

Format

```
►►──FLOPTS──┬──────────────────────────────┬─┬──────────────────────────────┬──────►
            └─CTLLUNAM──(───LU Name───)─┘ └─CTLMEM──(───XCFMemName───)─┘

►──┬──────────────────────────────────┬────────────────────────────────────────────►
   └─DSTGROUP──(───XCFGroupName───)─┘

►──┬──────────────────────────────────────────┬────────────────────────────────────►
   │               ┌─,──────────────────────┐ │
   └─SNADEST──(──┴─TrackerDest.DSTdest─┴──)─┘

►──┬──────────────────────────────────────────┬────────────────────────────────────►
   │               ┌─,──────────────────────┐ │
   └─TCPDEST──(──┴─TrackerDest.DSTdest─┴──)─┘
```

```
►──┬───────────────────────────────────────────┬──────────────────────────────►◄
   │               ┌─,──────────────────┐       │
   └─XCFDEST──(────┴─TrackerDest.DSTdest─┴───)───┘
```

Parameters

CTLLUNAM(LU Name)
: Defines the VTAM application LU name identifying the controller in the SNA connection between the controller and data store. It is required if the SNA connection is used between the controller and data store, and must be the same specified in the data store DSTOPTS CTLLUNAM statement.

  You must specify at least the keyword CTLLUNAM, or the keywords DSTGROUP and CTLMEM.

CTLMEM(XCFMemName)
: Defines the XCF member name identifying the controller in the XCF connection between the controller and data store. It must be specified together with the DSTGROUP keyword. It is required if the XCF connection is used between the controller and data store and must be the same specified in the data store DSTOPTS CTLMEM statement.

DSTGROUP(XCFGroupName)
: Defines the XCF group name identifying the controller in the XCF connection between the controller and data store. It must be specified together with the CTLMEM keyword. It is required if the XCF connection is used between the controller and data store.

  The XCF group defined to connect the controller to the data store must be different from the one defined in the XCFOPTS group to connect the controller to the z/OS tracker.

SNADEST(TrackerDest.DSTdest)
: Defines a table of pairs of tracker destinations and data store destinations, each pair separated by a period, when the data store destinations are LU names. SNADEST is used by the FL (Fetch Job Log) task to decide from which data store the job log will be retrieved. Several tracker destinations can be associated to the same data store only in a system where the spool is shared (for example, JES2 MAS). The tracker destination of 8 asterisks (********) identifies the data store associated with a tracker running in the same address space as the controller. You can connect the controller to SNA and XCF data stores at the same time, that is, you can specify SNADEST and XCFDEST together, but you cannot specify the same tracker or data store destinations in SNADEST and XCFDEST. You must specify at least SNADEST or XCFDEST.

  When using a shared DASD connection, you must specify the DDname of the submit/release dataset. When using a SNA connection, you must specify the tracker LU name.

TCPDEST(TrackerDest.DSTdest)
: Defines a table of pairs of tracker destinations and data store destinations, each pair separated by a period, when the destinations are TCP/IP destinations. The following rules apply to the destination sub-values:

  - The TrackerDest name can be up to 8 alphanumeric characters. In association with the host name or IP address, it is used to identify the communication partners. This sub-value is required.
  - The DSTdest consists of a host name or IP address and optionally a port number:

– The host name or IP address can be up to 52 alphanumeric characters. It can contain a host name or IP address in IPv4 or IPv6 format. Enclose this value in single quotation marks. This sub-value is required.
– The port number can be up to 5 numeric characters. Valid values are from 0 to 65535. This sub-value is optional. The default is 0, meaning that any port number is accepted.

- The TrackerDest name and the DSTdest name must be separate by a period.
- The required values and the port number must be separated by a slash.

TCPDEST is used by the Fetch Job Log (FL) task to decide from which data store the job log is to be retrieved. Several tracker destinations can be associated with the same data store only in a system where the spool is shared (for example, JES2 MAS). In this case, you must use the same address format for the different destinations. The tracker destination marked with 8 asterisks (********) identifies the data store associated with a tracker running in the same address space as the controller. You can connect the controller to SNA, XCF, and TCPIP data stores at the same time, that is, you can specify SNADEST, XCFDEST, and TCPDEST together, but you cannot specify the same tracker or data store destinations in SNADEST, TCPDEST, or XCFDEST. You must specify at least one of the following destinations: SNADEST, XCFDEST, or TCPDEST.

XCFDEST(TrackerDest.DSTdest)
Defines a table of pairs of tracker destinations and data store destinations, each pair separated by a period, when the data store destinations are XCF member names. XCFDEST is used by the FL task to decide from which data store the job log will be retrieved. Several tracker destinations can be associated to the same data store only in a system where the spool is shared (for example, JES2 MAS). Because the restart and cleanup function adds a job card to the procedures for scheduled STC workstation operations at the same time it adds the //TIVDSTxx output JCL statements, there are some exceptions to the previous instructions if you want to use the restart and Cleanup function. The JCL for a started task can contain a job card *only* if the JCL is in a data set in the IEFPDSI or IEFJOBS concatenations of MSTJCLxx when the start command is issued. The XCFDEST value (********.DSTdest) identifies the data store associated with a tracker running in the same address space or in the same LPAR as the controller. You can connect the controller to SNA and XCF data stores at the same time, that is, you can specify XCFDEST and SNADEST together, but you cannot specify the same tracker or data store destinations in XCFDEST and SNADEST. You must specify at least XCFDEST or SNADEST.

Examples

```
FLOPTS SNADEST (SNATRK1.SNAD001)    1
XCFDEST (XCFTRK1.XCFD001)           2
CTLLUNAM (LU0001)                   3
DSTGROUP (XCFGRP1)                  4
CTLMEM (GRP1001)                    5
TCPDEST(TRK1.'9.12.134.9'/5555)     6
```

In this example of an FLOPTS statement:

1 SNATRK1 is the tracker destination and SNAD001 is the data store destination separated by a period when the destination is an LU name.

2 XCFTRK1 is the tracker destination and XCFD001 is the data store destination separated by a period when the destination is an XCF member name.

3 LU0001 is the LU node name of the server communicating with the controller system.

4 XCFGRP1 is the name of the XCF group that the IBM Workload Scheduler for z/OS system should join.

5 GRP1001 is a member of the XCF group XCFGRP1.

6 TRK1 identifies a TCP/IP link with a data store and the details of it are defined in the TCPDEST keyword.

HTTPOPTS

Purpose

The HTTPOPTS statement defines the options for tracking jobs and for retrieving job execution logs that are issued on IBM Workload Scheduler for z/OS Agents.

Format

```
►►──HTTPOPTS──┬───────────────────────────────────────────────┬──────────────►
              │                   ┌─10────────────────┐       │
              └─CLNTHREADNUM──(───┴─number of threads─┴───)───┘

►──┬────────────────────────────────────────────────────┬────────────────────►
   │                  ┌─15───────────────────┐          │
   └─CONNTIMEOUT──(───┴─HTTP timeout interval─┴───)─────┘

►──┬──────────────────────────────┬──┬──────────────────────────────────────┬─►
   │                ┌─NO──┐       │  │               ┌─local hostname─┐     │
   └─ENABLEFIPS──(──┴─YES─┴──)────┘  └─HOSTNAME──(───┼─hostname───────┼──)──┘
                                                     └─IP address─────┘

►──┬──────────────────────────────────────────────┬──────────────────────────►
   │                     ┌─511───────┐            │
   └─HTTPPORTNUMBER──(───┴─HTTP port─┴───)────────┘

►──┬──────────────────────────────────────────────┬──────────────────────────►
   └─JLOGHDRTEMPL──(────member name────)──────────┘

►──┬────────────────────────────────────────────────────┬────────────────────►
   │                   ┌─1─────────────────┐            │
   └─JLOGTHREADNUM──(──┴─number of threads─┴───)────────┘

►──┬────────────────────────────────────────────────────┬────────────────────►
   │                     ┌─100─────────────┐            │
   └─JOBLOGMAXLINES──(───┴─number of lines─┴───)────────┘

►──┬──────────────────────────────────────────────┬──────────────────────────►
   │                       ┌─ONDEMAND─┐           │
   └─JOBLOGRETRIEVAL──(────┴─ONERROR──┴───)───────┘
```

```
►──┬──────────────────────────────────────────┬──┬──────────────────────────────────────────┬──►
   │                 ┌─FIRSTLAST─┐             │  │                           ┌─NO──┐         │
   └─JOBLOGSECTION──(─┼─FIRST─────┼─)─┘          └─OUTPUTCOLLECTOR──(─┴─YES─┴─)─┘
                      └─LAST──────┘

►──┬───────────────────────────────────────────────────────┬──────────────────────────────────►
   │               ┌─0──────────────────────────┐         │
   └─PULSEIVL──(─┴─heartbeat checking interval─┴─)─┘

►──┬─────────────────────────────────────────────────┬────────────────────────────────────────►
   │                    ┌─10────────────────┐         │
   └─SRVTHREADNUM──(─┴─number of threads─┴─)─┘

►──┬─────────────────────────────────────┬──┬──────────────────────────────────────────┬──────►
   │                   ┌─CAONLY─┐         │  │                     ┌─tws────────┐       │
   └─SSLAUTHMODE──(─┴─STRING─┴─)─┘       └─SSLAUTHSTRING──(─┴─SSL string─┴─)─┘

►──┬─────────────────────────────────────────────────────┬────────────────────────────────────►
   └─SSLKEYRING──(───SSL key ring db filename───)─┘

►──┬─────────────────────────────────────────────────────────┬────────────────────────────────►
   └─SSLKEYRINGPSW──(───SSL key ring psw filename───)─┘

►──┬──────────────────────────────────────┬──┬─────────────────────────────────────────┬──────►
   │                       ┌─SAF─┐         │  │               ┌─512─────────────┐       │
   └─SSLKEYRINGTYPE──(─┴─USS─┴─)─┘        └─SSLPORT──(─┴─SSL port number─┴─)─┘

►──┬─────────────────────────────────────────────────────┬────────────────────────────────────►
   │                     ┌─TCPIP──────────────┐          │
   └─TCPIPJOBNAME──(─┴─TCPIP started task─┴─)─┘

►──┬──────────────────────────────────────────────────────┬───────────────────────────────────►
   │                     ┌─300───────────────────┐        │
   └─TCPIPTIMEOUT──(─┴─TCPIP timeout interval─┴─)─┘

►──┬──────────────────────────────────────┬───────────────────────────────────────────────────►
   │              ┌─USRINFO─────┐           │
   └─USRMEM──(─┴─member name─┴─)─┘

►──┬────────────────────────────────────────────┬──┬───────────────────────────────┬──────────►
   │                  ┌─GLOBAL──────────────┐    │  │                 ┌─NO──┐        │
   └─VARTABLES──(─┼─APPL────────────────┼─)─┘  └─VARFAIL──(─┴─YES─┴─)─┘
                      └─table1,table2,...─┘

►──┬──────────────────────────────┬───────────────────────────────────────────────────────────►◄
   │              ┌─NO──┐          │
   └─VARSUB──(─┴─YES─┴─)─┘
```

Parameters

CLNTHREADNUM(*number of threads***|10)**
: The number of threads that are used by the HTTP client task to send more than one request at the same time. Valid values are from 5 to 100.

CONNTIMEOUT(*HTTP timeout interval***|15)**
: The time (in seconds) that an HTTP connection waits before a timeout occurs. Valid values are from 1 to 10000.

ENABLEFIPS(NO|YES)
: Indicates whether the SSL communication must comply with FIPS standards.

Specify YES to have a FIPS compliant SSL communication. This keyword is ignored if the SSL communication is not enabled.

For more information about how you activate the support for FIPS standard, see*IBM Workload Scheduler for z/OS: Planning and Installation*.

For more information about the FIPS compliance, see Step 23. Activating support for FIPS standard over SSL secured connections.

HOSTNAME(*hostname***|***IP address***|** *local hostname***)**
| The local host name or IP address used by the HTTP server component. To
| connect to z-centric agents, this parameter is required.

It can be up to 52 alphanumeric characters. It specifies a host name or IP address in IPv4 or IPv6 format. Enclose this value in single quotation marks. If you do not specify this parameter here, the system looks for the name specified for the same parameter in the TCPOPTS statement. If it finds no name there, it uses the default, which is the IP address returned by TCP/IP.

HTTPPORTNUMBER(*HTTP port***|511)**
The port number used by the HTTP server to listen for non-SSL connections. Valid values are from 0 to 65535.

JLOGHDRTEMPL(member name)
Specifies the name of the member in the EQQPARM data set that contains the template for the job log header. Use this option if you want to have a header placed on top of all the job logs produced in the z-centric environment. The header contains information necessary to classify the output (application name, job name, extended job name, workstation name, operation number, planned start, late start) and information about the run (job ID, return code, duration, actual start and end times). The header can be customized. The EQQPARM member is used by both the controller and the output collector. See *IBM Workload Scheduler for z/OS: Scheduling End-to-end with z-centric Capabilities* to learn how to customize the template for the job log header.

JLOGTHREADNUM(*number of threads***|1)**
The number of threads that are used by the HTTP client task to manage the requests concerning the job log. This parameter is valid only for IBM Workload Scheduler for z/OS agents.

Valid values are from 0 to 100. 0 means that the job log requests are managed as all the other requests submitted by the HTTP client task.

JOBLOGMAXLINES(*number of lines***|100)**
The maximum number of lines that the job log retriever can return for a single job log. This parameter is valid only for IBM Workload Scheduler for z/OS agents.

| Valid values are from 0 to 65000 lines. To specify which section of the job log
| must be retrieved, set the **JOBLOGSECTION** keyword.

JOBLOGRETRIEVAL(ONERROR|ONDEMAND)
This parameter defines the job log retrieval policy for operations running on z-centric and dynamic workstations. The valid values are:

ONDEMAND
The default value. The user must explicitly make a request to retrieve the joblog.

ONERROR
: After a job runs and ends in error, the joblog is automatically retrieved and sent. Manual changes to the error do not trigger the retrieval process to start.

`JOBLOGSECTION(FIRST|LAST|FIRSTLAST)`
The section of the job log retrieved from an IBM Workload Scheduler for z/OS agent that must be shown. The valid values are:

FIRSTLAST
The initial and last sections of the job log are retrieved. If the total number of lines to display exceeds the `JOBLOGMAXLINES` value, the central part is discarded and replaced with a lines of dashes.

FIRST Only the initial section of the job log is retrieved. If the total number of lines to display exceeds the `JOBLOGMAXLINES` value, the last part is discarded and replaced with a lines of dashes.

LAST Only the last section of the job log is retrieved. If the total number of lines to display exceeds the `JOBLOGMAXLINES` value, the first part is discarded and replaced with a lines of dashes.

`OUTPUTCOLLECTOR(YES|NO)`
Specifies if the logs of jobs and dynamic jobs run in a z-centric environment are to be retrieved by the `output collector` started task and copied to a SYSOUT in JES to be used by an output management product.

Specify YES to activate this feature since by default it is not active. If you activate it, you need to define the EQQOUCEV DD name in the controller JCL. The EQQOUCEV data set is used in the communication between the controller and the `output collector` (when you create the sample job JCL with EQQJOBS, a sample is generated to allocate this and other data sets).

The job log retrieval mechanism implemented by the Output collector is independent of the normal retrieval mechanism defined by the JOBLOGRETRIEVAL keyword. While Output collector retrieves all logs automatically and sends them to JES, the normal mechanism is based on user requests to the controller through one of the user interfaces.

In a sysplex configuration the Output collector started task must reside in the same image where the controller is.

See *IBM Workload Scheduler for z/OS: Scheduling End-to-end with z-centric Capabilities* for details on collecting job logs from IBM Workload Scheduler for z/OS Agents.

`PULSEIVL(`*`heartbeat checking interval`*`|0)`
Defines the frequency in minutes with which the controller checks that all attached z-centric agents and dynamic domain managers are running. This keyword defines the heartbeat interval globally for all devices. To set up heartbeat intervals for specific agents or dynamic domain managers, use the `pulseivl` parameter in the HTTP/HTTPS keyword of the ROUTOPTS statement.

The zero default value implies that no heartbeat checking is run. The maximum allowed is 1440 minutes.

`SRVTHREADNUM(`*`number of threads`*`|10)`
The number of threads that are used by the HTTP Server task to process more than one request at the same time. Valid values are from 2 to 100.

`SSLAUTHMODE(STRING|CAONLY)`
The SSL authentication type. Valid values are:

CAONLY
The scheduler checks the certificate validity by verifying that a recognized Certification Authority has issued the peer certificate. The information contained in the certificate is not checked. This is the default value.

STRING
The scheduler checks the certificate validity as described in the CAONLY option. It also verifies that the Common Name (CN) of the Certificate Subject matches the string specified in the SSLAUTHSTRING parameter.

`SSLAUTHSTRING(`*`SSL string`*`|tws)`
The SSL string used to verify the certificate validity when you set SSLAUTHMODE to STRING. The string can be up to 64 characters.

`SSLKEYRING(`*`SSL key ring db filename`*`)`
If SSLKEYRINGTYPE is SAF, this parameter specifies the SAF key ring used to connect the security certificates.

If SSLKEYRINGTYPE is USS, this parameter specifies the database containing keys and certificates. It consists of an SSL working directory name and file name, in the format *`SSLworkdir`*`/TWS.kbd`.

This parameter is case-sensitive.

`SSLKEYRINGTYPE(USS|SAF)`
Specifies if the key ring file is a key database USS file or a SAF key ring. If this value is set to SAF, you can use the RACF command to manage SSL connections.

`SSLKEYRINGPSW(`*`SSL key ring psw filename`*`)`
If SSLKEYRINGTYPE is USS, specifies the file containing the key password. It consists of an SSL working directory name and file name, in the format *`SSLworkdir`*`/TWS.sth`. This parameter is case-sensitive.

`SSLPORT(`*`SSL port number`*`|512)`
The SSL port number used by the HTTP server to listen for SSL-connections. Valid values are from 0 to 65535.

If you want to disable the SSL connection, you can either:
- Specify neither SSLKEYRING nor SSLPORT keywords.
- Specify `SSLPORT(0)`.

`TCPIPJOBNAME(`*`TCPIP started task`*`|TCPIP)`
The name of the TCP/IP started task running on the z/OS system where you have the scheduler installed. If you do not specify this parameter here, the system looks for the name specified for the same parameter in the TCPOPTS statement. If it finds no name there, it uses the default started task named TCPIP.

`TCPIPTIMEOUT(`*`TCPIP timeout interval`*`|300)`
The time (in seconds) that an HTTP request waits for the response before a timeout occurs. Valid values are from 1 to 10000.

`USRMEM(`*`member name`*`|USRINFO)`
The PARMLIB member where the user definitions are stored. If you are working with fault-tolerant and z-centric end-to-end solutions at the same

time, ensure that the member you set here is different from the member set in the USRMEM keyword of the fault-tolerant end-to-end environment.

This keyword is required only if you run jobs on Windows workstations where the user password is requested.

VARTABLES(GLOBAL|APPL|*table1*,*table2*,...)
Applies to job types with advanced options defined from the Dynamic Workload Console and identifies a list of variable tables that must be searched, and the search order. **APPL** indicates the application variable table. **GLOBAL** indicates the table defined in the `GTABLE` keyword of the `OPCOPTS` controller. The maximum number of characters that can be specified for the table name is 16 and you cannot specify more than 16 tables in the list.

VARFAIL(NO|YES)
Applies to job types with advanced options defined from the Dynamic Workload Console. Set this keyword to YES so that if a variable substitution error occurs, the job results in a JCL error (OJCV). Specify NO so that the variable string remains unchanged, without substituting it with a value and does not report a JCL error (OJCV).

VARSUB(NO|YES)
Applies to job types with advanced options defined from the Dynamic Workload Console and indicates whether variable substitution should be enabled. Specify YES so that variable scanning is performed for job types with advanced options. Specify NO so that the variable string is left unchanged and is not substituted with a value.

INCLUDE

Purpose

The INCLUDE statement reduces the size of the parameter library member that contains the OPCOPTS and JTOPTS statements. Also, this statement reduces the maintenance activities for this member.

By using the INCLUDE statement, the definition of the NOERROR table can be provided by several members of the EQQPARM library. Each of these members could reside in a different data set. Each data set could be protected by a different RACF profile.

The net effect is that it becomes possible to divide the NOERROR information into several parts, each requiring a different access authority.

Note: If you specify multiple members, you cannot use the NOERRMEM command to update the NOERROR list. Changes to the NOERROR list comes into effect when the controller is restarted.

Format

```
                           ┌,─────────────┐
►►──INCLUDE──NOERROR──(────▼──DEPT1, DEPT2──┴──)──────────────────────────►◄
```

Parameters

NOERROR(*list of members***)**
This keyword is used to request IBM Workload Scheduler for z/OS to read NOERROR information from other members of the EQQPARM library. The parameter library members specified by this keyword can only contain NOERROR statements.

Examples

```
INCLUDE NOERROR(DEPT1, DEPT2)
```

In this example, NOERROR information is retrieved from two members, DEPT1 and DEPT2, of the EQQPARM library.

INIT

Purpose

The INIT statement defines the run-time options for processing requests that are sent to IBM Workload Scheduler for z/OS from a PIF application. The parameters specified in the INIT statement override the settings specified in the INIT request of the PIF application. INIT is defined in the parameter file identified by the EQQYPARM DD statement in the JCL of the PIF application and in the parameter file identified by the EQQPARM DD statement in the server procedure.

The EQQYPARM DD statement is valid also for batch loader jobs.

Note: If you plan to run PIF applications many times per day from a long-running non-TSO address space (for example, NetView), to prevent a storage shortage do not specify the EQQYPARM ddname. Instead, specify the parameters either in the PIF application or in the controller INTFOPTS initialization statement. When you run a PIF application by specifying the EQQYPARM ddname, a TSO environment must be established each time and some of the resources remain allocated until the task ends. This might lead to a storage shortage, if the commands are issued many times.

Format

```
►►──INIT──┬──────────────────────────┬──┬────────────────────────┬──────────────►
          │               ┌─N─┐      │  └─ADOPSEG──(───VERS0───)─┘
          └─ADOICHK──(────┴─Y─┴───)──┘

►──┬───────────────────────────────────┬─────────────────────────────────────────►
   └─CALENDAR──(───calendar name───)───┘

►──┬──────────────────────────────────────────────────────┬──────────────────────►
   └─CWBASE──(───base year for PIF century window───)─────┘

►──┬────────────────────────┬──┬───────────────────────────────┬─────────────────►
   │           ┌─N─┐        │  │              ┌─711231─┐       │
   └─DATINT──(─┴─Y─┴───)────┘  └─HIGHDATE──(──┼─991231─┼──)────┘
                                              └─cccccc─┘
```

```
►──┬───────────────────────┬──┬────────────────────────────────────┬──────────────────►
   └─LUNAME──(───LU name───)─┘  │                     ┌─ERROR──┐     │
                                └─OIWSNAME──(───┴─IGNORE─┴──)─┘

►──┬────────────────────────────────────────┬─────────────────────────────────────────►
   └─REMHOSTNAME──(──┬─hostname───┬──)─┘
                     └─IP address─┘

►──┬────────────────────────────────────────────┬─────────────────────────────────────►
   │                       ┌─425─────────┐       │
   └─REMPORTNUMBER──(───┴─TCPIP port─┴──)─┘

►──┬─────────────────────────────────────┬──┬───────────────────────┬─────────────────►
   └─SUBSYS──(───subsystem name───)─┘  │            ┌─0─┐        │
                                          └─TRACE──(──┼─4─┼──)─┘
                                                       └─8─┘

►──┬───────────────────────┬──┬──────────────────────────────┬────────────────────────►
   └─USRLEV──(──┬─9──┬──)─┘  │                  ┌─NO───┐      │
                └─10─┘       └─VERADGRD──(──┼─FULL─┼──)─┘
                                                 └─YES──┘

►──┬─────────────────────────────────┬────────────────────────────────────────────────►◄
   │                 ┌─NO───┐          │
   └─VERSRWSN──(──┼─FULL─┼──)─┘
                     └─YES──┘
```

Parameters

ADOICHK(Y|N)

Use this option to specify whether or not you want AD/OI consistency checks to be made every time an application is deleted or modified.

Consistency checks involve looking in the application description database for matches for all the operator instructions in the application. Any operator instructions without a match are deleted.

The checks are made immediately after the application description PIF action has completed with a zero return code.

Y Consistency checks are performed whenever an application description record is deleted or replaced using the PIF.

N (default)
Consistency checks are not performed.

ADOPSEG (VERS0)

PIF applications can use this option to keep the values for certain fields in the operation part of application descriptions at a Replace.

The ADOPSEG can be used for programs that do not define the fields following the ADOPWSINFO fields in the ADOP segment, and that do not keep the value in the empty field at the segment end.

When updating an application description, Replace, the fields following the ADOPWSINFO in the ADOP segment are not updated, their current values are kept. When creating an application description, or adding an operation to an application description, the fields following the ADOPWSINFO fields in the ADOP segment are given default values.

CALENDAR(*calendar name***)**
Specifies the IBM Workload Scheduler for z/OS default calendar. You can specify a name from 1 to 16 characters, referencing a calendar in the calendar database.

This parameter is used by the programming interfaces only if no calendar is specified in the application description. It is used by programming interfaces to validate that the EVERY options specified for an application run cycle are consistent with the application calendar work day end time.

If DEFAULT is specified and no calendar with the name DEFAULT exists, all days are considered work days and the work day end time is set to 00.00.

If this parameter is not set or the specified calendar does not exist in the calendar database, the validation on the EVERY options is not performed. Any inconsistencies in the definition will be highlighted by a warning message during long-term plan creation or modification.

CWBASE(*base year for PIF century window***)**
Specifies the origin for the century window used by the PIF application. Valid values are 00 through 99. If you specify 00, IBM Workload Scheduler for z/OS uses the same date representation in communication with the PIF application as in the dialog. If you specify 72, the IBM Workload Scheduler for z/OS internal date representation will be used. This parameter affects the date representation for all dates except the default valid-to and out-of-effect dates. The default dates are determined by the value of the HIGHDATE parameter or the PIFHD parameter of the INTFOPTS statement.

For more information about the base year, see the PIFCWB parameter of the INTFOPTS statement.

The CWBASE parameter overrides the global setting of the PIFCWB parameter of the INTFOPTS statement.

DATINT(Y|N)
The keyword allows updates of the same record by different users to be serialized as soon as the request is handled by the PIF code. The default value, N, serializes the update of the same record only immediately before the VSAM access and this could lead to unpredictable results for a PIF user. Specify Y if you want to obtain the serialization at PIF level. This means that if a PIF program containing an update request starts, and in the meantime, another PIF program runs with the same record update request, the first one will be rejected and its update will not be handled. Note that the user IDs of the two programs must be different. If the same user ID is used, for example, by a PIF program and an OCL program, no serialization is performed.

HIGHDATE(991231|711231|*cccccc***)**
Specifies the high date presented to the PIF application in the default valid-to fields of applications and run cycles. For more information about the high date, see the PIFHD parameter of the INTFOPTS statement.

You can select one of these values:

991231 Use this if you have specified the CWBASE parameter as 72.

711231 As displayed in the IBM Workload Scheduler for z/OS dialog. This date represents 31 December 2071.

cccccc A 6-character string to symbolize the default valid-to date, for example *DEFHID*. The string cannot include numeric characters. The character string will be displayed and will always be processed by IBM Workload Scheduler for z/OS as the default valid-to date.

The HIGHDATE parameter overrides the global setting of the PIFHD parameter of the INTFOPTS statement.

LUNAME(*LU name***)**
Identifies the LU node name of the server communicating with the controller system. This keyword overrides the setting of the LU name given in the EQQYCOM INIT request. This is valid only if the server uses the APPC protocol.

OIWSNAME(IGNORE|ERROR)
Specifies whether to ignore the workstation name argument when it is used in a PIF program as the argument of an OI/OICOM resource code.

By default, an error message appears, for instance, when an OICOM list or an OI deletion is requested with the use of the work station name argument. The message states the erroneous use of the no longer significant argument and no list or deletion will be performed. The workstation name argument, when present, is simply ignored if the OIWSNAME keyword is set to IGNORE.

REMHOSTNAME (*hostname***|***IP address***)**
The server host name or IP address used by the PIF program to communicate with the server through a TCP/IP network. This parameter overrides the value of REMHOST specified in the EQQYCOM INIT request. If you specify the REMHOSTNAME parameter in the INIT statement of the server procedure, it is ignored. REMHOSTNAME and LUNAME are mutually exclusive.

REMPORTNUMBER (*value***|425)**
The server TCP/IP port number used by the PIF program to communicate with the server through a TCP/IP network. This parameter overrides the value of REMPORT specified in the EQQYCOM INIT request. If you specify the REMPORTNUMBER parameter in the INIT statement of the server procedure, it is ignored. Valid values are from 0 to 65535. The default is 425. REMPORTNUMBER and LUNAME are mutually exclusive.

SUBSYS(*subsystem name***)**
Identifies the name of the subsystem controller to which the request is directed. This keyword overrides the setting of the RESOURCE parameter in the EQQYCOM INIT request.

TRACE(4 |8|0)
Defines the level of trace information IBM Workload Scheduler for z/OS writes to the diagnostic file (EQQDUMP). Specify 0, which is the default value, if you do not want trace information. Specify 4 if you want partial trace information. Specify 8 if you want full trace information.

| **USRLEV (9|10)**
| Introduced by APAR PI57310 for compatibility reasons, communicates to EQQYCOM the level of user program. Use the parameter USRLEV to enable the dialog between the old user-written PIF programs and the new PIF delivered through the Small Programming Enhancements (SPE), which has caused changes in the segment layouts.

| Valid values are:

| **9** Identifies the Small Programming Enhancements (SPE) delivered through the APAR PI57310.

| **10** Identifies the Small Programming Enhancements (SPE) delivered through the APARs PI62520 and PI62521.

VERADGRD(FULL|YES|NO)
Application descriptions that are members of an application group have the

name of the group definition in field ADGROUPID of segment ADCOM. VERADGRD controls the verification of this field when a new application description is created or an existing one is replaced. The verification is done for active application descriptions.

Specify FULL to check that the group definition is active and valid for at least a part of the validity period of the application description being created or updated.

Specifying YES is the same as specifying FULL, except that the application group id is accepted if the application description already has this application group id. It could be an update without any change to the application group id or an insert of a new version when there already are active versions with the same application group id.

Specifying NO means that no check is made to verify that the application group exists.

VERSRWSN(FULL|YES|NO)
The special resource description, SR, has fields representing workstations, the full workstation names or generic names; field SRDWSNAME of segment SRDWS for default connected workstations, field SRIWSNAME of segment SRIWS for workstations connected to an interval. VERSRWSN controls the verification of these fields when a new special resource is created or an existing one is replaced.

Specify FULL to verify the workstation fields against the workstation description file. Each workstation field in the resource description must match at least one of the workstation descriptions.

Specifying YES is the same as specifying FULL except that the workstation value is accepted if the resource description already has this workstation name. It could be an update without any change to the workstation names.

Specifying NO means that no check is made to verify that the workstation description exists.

Examples

```
INIT SUBSYS(OPCB)              1
     HIGHDATE(991231)          2
     CWBASE(72)                3
     LUNAME(SEIBM200.IS4MSV3B) 4
```

In this example of an INIT statement:

1 A PIF application sends a request to the OPCB subsystem. The statement will override global settings of the INTFOPTS statement.

2 The default valid-to date will be presented to the to the PIF application as 991231.

3 72 is the base year for the PIF application.

4 The request is sent to the LU node SEIBM200.IS4MSV3B.

INTFOPTS

Purpose

The INTFOPTS statement defines the global run-time options for handling requests from programming interfaces, for example PIF application requests. Specify this statement for a controller or standby controller. This is a required statement.

INTFOPTS is defined in the member of the EQQPARM library as specified by the PARM parameter on the JCL EXEC statement.

Format

```
►►──INTFOPTS──┬────────────────────────────────────────────────────┬──────────►
              │                 ┌─00─────────────────────────────┐ │
              └─PIFCWB──(───────┴─base year for PIF century window─┴───)─┘


          ┌─711231─┐
►─PIFHD──(─┼─991231─┼──)──────────────────────────────────────────────────────►◄
          └─cccccc─┘
```

Parameters

PIFCWB(*base year for PIF century window*|00)
Defines the origin for the PIF century window. Valid values are 00-99.

The origin is the year that you want to be represented as 00 in the 00 to 99 year span (century window). For example, if the PIFCWB is 72, then 1972 is treated as 00, 1992 as 20, and 2002 as 30.

Internally IBM Workload Scheduler for z/OS works with a two-digit year format, so dates are represented as 00 to 99. In order to handle dates before and after 2000, IBM Workload Scheduler for z/OS has chosen 72 as its base year. This means that internally, 1972 is represented as 00, 1995 as 23, and 2071 as 99.

When you use PIF applications, the internal dates are presented to PIF applications. The PIFCWB defines whether the dates are presented as they are stored, or if they are translated into some other base.

If you choose 72 as the PIFCWB, the dates are presented as stored, 1995 will be presented as 22 and 2071 as 99. Choosing 72 makes it possible to sort in the right order dates that are before and after 2000.

If you choose 00 as the PIFCWB, the dates are presented as displayed in the ISPF dialogs; 1995 as 95, and 2071 as 71. This is the default.

Note: The global PIFCWB setting is overridden if a PIF program specify the CWBASE parameter of the INIT statement.

PIFHD(991231|711231|*cccccc*)
Specifies the high date value presented in the default valid-to fields of applications and run cycles. This date only affects *default* valid-to dates. You can select one of these values:

991231 Use this value together with PIFCWB(72).

711231 As displayed in the IBM Workload Scheduler for z/OS dialog. This date represents 31 December 2071.

cccccc A 6-character string to symbolize the default valid-to date, for example *HGHDAT*. The string cannot include numeric characters. Defining the PIFHD using a string of characters, means that IBM Workload Scheduler for z/OS will always interpret this as the default valid-to date.

This is a required keyword.

Note: If a PIF program specifies the HIGHDATE parameter of the INIT statement, the global PIFHD setting is overridden.

Examples

```
INTFOPTS  PIFCWB(72)        1
          PIFHD(991231)     2
```

In this example of an INTFOPTS statement:

1 PIF applications use 72 as the base year. This means that 1 January 1974 is used as 020101, and 1 January 2002 is used as 300101.

2 IBM Workload Scheduler for z/OS will present 991231 as the default valid-to date to PIF applications.

JCCOPTS

Purpose

The JCCOPTS statement defines run-time options to the job-completion-checker task. This statement is used by a tracker when OPCOPTS JCCTASK (YES) is specified.

JCCOPTS is defined in the member of the EQQPARM library as specified by the JCCPARM parameter in the OPCOPTS statement.

Format

```
►►─JCCOPTS─┬──────────────────────────────────────┬──────────────────►
           │                ┌─A──────────────┐    │
           └─CHKCLASS──(────┴─SYSOUT classes─┴──)─┘

►─┬────────────────────────────────────┬─────────────────────────────►
  └─INCDSN──(────incident file dsname────)─┘

►─┬──────────────────────────────────────┬───────────────────────────►
  │             ┌─112────────────────┐   │
  └─JCCQMAX──(──┴─maximum queue size─┴──)─┘

►─┬────────────────────────────────────────┬─────────────────────────►
  │              ┌─0──────────────────┐    │
  └─JCCREQUD──(──┴─JCC requested delay─┴──)─┘

►─┬───────────────────────────────────────────┬──────────────────────►
  │               ┌─10─────────────────────┐   │
  └─JCSAMECHK──(──┴─same sysout check limit─┴──)─┘
```

```
►─┬──────────────────────────────────┬─┬──────────────────────────────────────┬──────────────►
  │           ┌─4──────────┐         │ │              ┌─200─────────┐         │
  └─JCWAIT──(─┴─wait limit─┴──)──────┘ └─MAXDELAY──(──┴─delay limit─┴──)──────┘

►─┬───────────────────────────────┬─┬─────────────────────────────────────────┬──────────────►
  │               ┌─⌂──────┐      │ │             ┌─50──────────────┐         │
  └─SYSOUTDISP──(─┼─D──────┼──)───┘ └─UMAXLINE──(─┴─number of lines─┴──)──────┘
                  ├─H──x───┤
                  ├─R──────┤
                  └─R──x───┘

►─┬───────────────────────────────┬───────────────────────────────────────────────────────────►◄
  │            ┌─JOB────┐         │
  └─USYSOUT──(─┼─ALWAYS─┼──)──────┘
               └─NEVER──┘
```

Parameters

CHKCLASS(*SYSOUT classes***|A)**

Defines the SYSOUT classes that IBM Workload Scheduler for z/OS uses to check whether SYSOUT is available. You can specify a maximum of 16 SYSOUT classes. The SYSOUT classes are defined as a character string of valid SYSOUT classes, one character for each SYSOUT class. SYSOUT selection is also influenced by the value of the SYSOUTDISP keyword.

Any SYSOUT class can be specified in a JES2 system. In a JES3 system, you must define any SYSOUT class that is to be processed by the job completion checker:

- As an external-writer SYSOUT class
- As HOLD=EXTWTR and TYPE=PRINT in the JES3 SYSOUT initialization statement

  If you define the SYSOUT CLASS as TYPE=DSISO, IBM Workload Scheduler for z/OS will be able to process only SYSTEM SYSOUT data sets. For both JES2 and JES3, the sysout classes defined by CHKCLASS cannot be used by a sysout archival product, or JES offload, or any other process which could delete the output before the JCC has processed it.

  For example when you need to have a configuration with the data store subsystem and the tracker with the JCC task active on the same z/OS system image, there could be compatibility problems if the JCC task options ask IBM Workload Scheduler for z/OS to delete the sysout output data sets after the usual analysis. This is because the JCC task might also delete the duplicated sysout copy created for the data store before it has been successfully stored. In this specific configuration, to avoid this problem and to improve the JCC performance (that would be scanning the same sysout data sets twice), you need to provide a JES class associated to the tracker destination, to be used for the JCC processing of the sysout data sets, in CHKCLASS option of JCCOPTS. The mandatory requirement is that it must not be one of the sysout classes specified in the RCLOPTS parameter keyword DSTCLASS. In this way the JCC task will never process the output data sets meant for data store processing. For more information, see DTCLASS.

No more than one JCC task can process a particular SYSOUT class.

Note: The keyword value is valid even if IBM Workload Scheduler for z/OS or the JCC subtask is stopped. It is not changed until you specify a different value and restart IBM Workload Scheduler for z/OS or the JCC subtask.

INCDSN(*incident file dsname***)**
: Defines the name of the incident log data set. This must be a cataloged, sequential data set on a direct access storage device (DASD). Several IBM Workload Scheduler for z/OS and OPC/A systems can use the same data set. Non-IBM Workload Scheduler for z/OS functions can update or reallocate the data set while IBM Workload Scheduler for z/OS is running.

JCCQMAX(*maximum queue size***|**__**112**__**)**
: Defines the maximum number of 3P (job termination) event records that the JCC queue can hold. The default value is 112. If the value that you specify is not a multiple of 16, then it is rounded to the next multiple of 16.

If message EQQZ035E is issued on a system where the JCC is used, consider increasing the value of JCCQMAX. Refer to message EQQZ035E in *Messages and Codes* for more information.

JCCREQUD(*JCC requested delay* **| 0)**
: Defines (in hundredths of seconds) how long the job completion checker (JCC) waits before issuing the request specified in SYSOUTDISP to JES. The maximum allowed value is 500 (5 seconds); any value exceeding this limit is automatically reset to 500.

Consider that high values can delay operations' tracking, therefore you might want to use the lowest working value or the default (0).

JCSAMECHK(*same sysout check limit***|**__**10**__**)**
: Defines the number of consecutive times that the job completion checker (JCC) can accept the same data set name as the next data set to be read from the spool, before detecting a loop situation. A loop situation occurs when, for example, the data set is stored damaged in the JES spool. When a loop situation is detected, the JCC performs the following actions:

1. Stops processing the current and remaining data sets for the job being processed.
2. Applies the actions set in the SYSOUTDISP parameter, if possible.
3. Issues the message EQQW902E to notify the problem occurred to the job being processed: the job is set to error code JCCE, unless the system already set another error status for that job.
4. Switches to process the next job in the JCC queue.

The value 0 means that no loop detection is performed.

JCWAIT(*wait limit***|**__**4**__**)**
: Defines how long the JCC waits (in seconds) before rechecking with JES to see if the SYSOUT for a job is available. Rechecking can continue for the time specified by the MAXDELAY keyword.

MAXDELAY(*delay limit***|**__**200**__**)**
: Defines how long the JCC should try to retrieve SYSOUT from JES when JES indicates that the job does not have SYSOUT in any of the classes checked by the JCC. The MAXDELAY value is specified in seconds. If the delay limit is reached, the operation is set to ended-in-error.

SYSOUTDISP(*SYSOUT disposition***|**__**⌂**__**)**
: Defines the action to be taken with SYSOUT data sets that have been processed. You can select one of these values:

⌂	It represents a blank space. No disposition is specified. The JCC selects only nonheld SYSOUT, and the SYSOUT data sets are deleted after processing.

D Disposition is delete. The JCC selects only held SYSOUT, and the SYSOUT data sets are deleted after processing.

H*x* Disposition is requeued and held. The JCC selects only held SYSOUT, and the SYSOUT data sets are requeued to SYSOUT class *x* after processing. Requeued data sets are held in the new SYSOUT class *x*.

R Disposition is process, no requeue. The JCC selects only held SYSOUT, and the SYSOUT data sets remain in hold status after processing.

R*x* Disposition is requeued. The JCC selects only held SYSOUT, and the SYSOUT data sets are requeued to SYSOUT class *x* after processing. Requeued data sets are not held in the new SYSOUT class *x*.

Notes:

1. The keyword value is valid even if IBM Workload Scheduler for z/OS or the JCC subtask is stopped. It is not changed until you specify a different value and restart IBM Workload Scheduler for z/OS or the JCC subtask. If the data store is used, no requeue will be done.
2. When blank or D is specified, this keyword affects the restart and cleanup functions and RCLOPTS DSTCLASS keyword should be specified. See RCLOPTS statement for details.

UMAXLINE(*number of lines***|50)**

Defines how many lines to scan in each user SYSOUT data set. You can specify 0 through 2 147 328 000 lines. The value 0 requests the scanning of all lines.

If you write system dumps to SYSOUT, ensure dump records are not scanned.

USYSOUT(ALWAYS|NEVER|JOB)

Requests scanning of user SYSOUT data sets:

ALWAYS
User SYSOUT data sets are always scanned.

NEVER
User SYSOUT data sets are never scanned.

JOB User SYSOUT data sets are scanned only if there is a job-specific message table. See "JCC message tables" on page 299 for information about message tables.

Examples

```
JCCOPTS CHKCLASS(CDEQ)   1
        JCWAIT(5)        2
        SYSOUTDISP(RA)   3
        UMAX(1000)       4
```

In this example of a JCCOPTS statement:

1 The JCC checks SYSOUT classes C, D, E, and Q for SYSOUT data sets.

2 If the output processing of a job is delayed, the JCC waits 5 seconds before rechecking the output queue for the SYSOUT for that job.

3 If a SYSOUT data set is found, it is processed according to the message tables and requeued to class A.

4 The JCC scans up to 1000 lines in user SYSOUT data sets.

JTOPTS

Purpose

The JTOPTS statement defines how operations behave at workstations and how they are submitted and tracked. This statement is used by a primary, backup, or standby controller.

JTOPTS is defined in the member of the EQQPARM library as specified by the PARM parameter on the JCL EXEC statement.

Format

```
►►──JTOPTS──┬─────────────────────────────────────────────────────────────┬──────────────►
            │                ┌─100────────────────┐     ┌─000─┐          │
            └─ALEACTION──(───┴─alert action limit─┴──,──┼─────┼──)──────┘
                                                        └─nnn─┘

►──┬─────────────────────────────────────────┬─┬──────────────────────────┬─────────────►
   │              ┌─400──────────────┐      │ │             ┌─NO──┐      │
   └─BACKUP──(───┼─backup frequency─┼──)───┘ └─CONDSUB──(───┴─YES─┴──)──┘
                  └─NO───────────────┘

►──┬────────────────────────────┬─┬───────────────────────────────┬─────────────────────►
   │              ┌─YES─┐       │ │               ┌─CURRENT─┐     │
   └─CRITJOBS──(──┴─NO──┴──)────┘ └─CURRPLAN──(───┴─NEW─────┴──)──┘

►──┬─────────────────────────────────────────────────────┬───────────────────────────────►
   │               ┌─100─────────────────────────┐        │
   └─DLIMFDBK──(───┴─limit for deadline feedback─┴──)────┘

►──┬───────────────────────────────────────────────────────┬─┬───────────────────────┬──►
   │                  ┌─50──────────────────────┐          │ │          ┌─NO──┐      │
   └─DSMOOTHING──(───┴─deadline smoothing factor─┴──)─────┘ └─DUAL──(───┴─YES─┴──)──┘

►──┬──────────────────────────────────────┬─┬───────────────────────┬──────────────────►
   │            ┌─,──────────────┐        │ │          ┌─NO──┐       │
   │            ▼                │        │ └─ETT──(───┴─YES─┴──)───┘
   └─ERRRES──(────error code─────┴──)────┘

►──┬─────────────────────────────────────────────────────────┬──────────────────────────►
   │                   ┌─YES─┐                               │
   └─ETTGENSEARCH──(───┴─NO──┴──,──┬─────────┬──)────────────┘
                                   ├─JOBONLY─┤
                                   └─SRONLY──┘

►──┬─────────────────────────────┬─┬──────────────────────────┬─────────────────────────►
   │                ┌─NO──┐      │ │             ┌─0────┐      │
   └─ETTNEWDEP──(───┴─YES─┴──)──┘ └─EVELIM──(───┴─nnnn─┴──)───┘

►──┬─────────────────────────────┬─┬──────────────────────────┬─────────────────────────►
   │                ┌─NO──┐      │ │             ┌─YES─┐       │
   └─FIRSTFDBK──(───┴─YES─┴──)──┘ └─FTWJSUB──(───┴─NO──┴──)───┘

►──┬──────────────────────────────────────────────────┬──────────────────────────────────►
   │             ┌─4──────────────────────────┐        │
   └─HIGHRC──(───┴─highest no-error return code─┴──)──┘
```

```
►──┬─────────────────────────────┬──┬──────────────────────────┬──────────────────►
   │              ┌─YES─┐        │  │          ┌─NO──┐         │
   └─HOSTJSUB──(──┴─NO──┴────)───┘  └─ITOM──(──┴─YES─┴──)──────┘

►──┬──────────────────────────────┬──┬──────────────────────────────┬─────────────►
   │              ┌─YES──┐        │  │               ┌─YES─┐        │
   └─JOBCHECK──(──┼─NO───┼───)────┘  └─JOBSUBMIT──(──┴─NO──┴──)─────┘
                  └─SAME─┘

►──┬─────────────────────────┬──┬──────────────────────────────────┬──────────────►
   │            ┌─5──┐       │  │                     ┌─YES─┐      │
   └─JTLOGS──(──┴─nn─┴──)────┘  └─LARGEUSERBUFFER──(──┴─NO──┴──)───┘

►──┬───────────────────────────────────────────────────────┬──────────────────────►
   │             ┌─100──────────────────────────┐          │
   └─LIMFDBK──(──┴─limit for duration feedback──┴──)───────┘

►──┬────────────────────────────────────────────────────────┬─────────────────────►
   │               ┌─0───────────────────────────┐          │
   └─MAXJSFILE──(──┼─maximum size of JS data set─┼──)───────┘
                   └─NO──────────────────────────┘

►──┬──────────────────────────────────┬──┬─────────────────────────────────┬──────►
   │               ┌─32767───┐        │  │                    ┌─NO──┐      │
   └─MAXOCCNUM──(──┴─nnnnnnn─┴───)────┘  └─MCPDATASPACE──(──┴─YES─┴──)─────┘

►──┬──────────────────────────────────────────────────────────────┬───────────────►
   │                ┌─30────────────────────────────────┐         │
   └─NEWOILIMIT──(──┴─days operator instructions are new─┴──)─────┘

►──┬────────────────────────────────────────────┬─────────────────────────────────►
   │               ┌─,──────────────────┐       │
   │               ▼                    │       │
   └─NOERROR──(────────error code entry─┴──)────┘

►──┬──────────────────────────────────┬──┬──────────────────────────────┬─────────►
   │              ┌─1──────────┐      │  │                  ┌─IP──┐     │
   └─OFFDELAY──(──┴─delay time─┴──)───┘  └─OPINFOSCOPE──(───┴─ALL─┴──)──┘

►──┬────────────────────────────────────┬──┬────────────────────────────────┬─────►
   │                         ┌─Y─┐      │  │                       ┌─Y─┐    │
   └─OPREROUTEDEFAULT──(─────┴─N─┴──)───┘  └─OPRESTARTDEFAULT──(───┴─N─┴──)─┘

►──┬───────────────────────────────┬──┬───────────────────────────┬───────────────►
   │                ┌─FINAL─┐      │  │                ┌─0────┐   │
   └─OUTPUTNODE──(──┴─ANY───┴──)───┘  └─OVERCOMMIT──(──┴─nnnn─┴──)┘

►──┬─────────────────────────────────────────────┬────────────────────────────────►
   │               ┌─6─────────────────────┐     │
   └─PLANSTART──(──┴─planning period start─┴──)──┘

►──┬───────────────────────────────┬──┬─────────────────────────────────┬─────────►
   │                 ┌─YES─┐       │  │              ┌─5────────────┐   │
   └─PRTCOMPLETE──(──┴─NO──┴──)────┘  └─QUEUELEN──(──┴─queue length─┴──)┘

►──┬───────────────────────────────┬──┬─────────────────────────────────┬─────────►
   │                ┌─YES─┐        │  └─RISKCONFIDENCE──(──1 - 99──)────┘
   └─RECCPCOMPL──(──┴─NO──┴──)─────┘
```

```
►──┬──────────────────────────────────┬──┬────────────────────────────────────────────┬──►
   │                ┌─0───┐           │  │               ┌─50────────────────┐        │
   └─SHUTDOWNPOLICY─(─┴─nnn─┴──)───────┘  └─SMOOTHING──(──┴─smoothing factor──┴──)─────┘

►──┬────────────────────────────────────┬──┬──────────────────────┬────────────────────►
   │               ┌─,────────────┐     │  │          ┌─0──┐      │
   │               ▼              │     │  └─STATIM──(─┴─nn─┴──)──┘
   └─STATMSG──(───┬─CPLOCK──┬─────┴──)──┘
                  ├─EVENTS──┤
                  ├─WSATASK─┤
                  └─GENSERV─┘

►──┬──────────────────────────────┬──┬────────────────────────────────┬────────────────►
   │              ┌─NO──┐         │  │                   ┌─R──┐       │
   └─STEPINFO──(──┴─YES─┴──)──────┘  └─SUBFAILACTION──(──┼─C──┼──)────┘
                                                         ├─E──┤
                                                         ├─RH─┤
                                                         ├─XC─┤
                                                         ├─XE─┤
                                                         └─XR─┘

►──┬───────────────────────────────┬──┬─────────────────────────────────┬──────────────►
   │                    ┌─R─┐      │  │                     ┌─0───┐     │
   └─SUPPRESSACTION──(──┼─C─┼──)───┘  └─SUPPRESSPOLICY──(───┴─nnn─┴──)──┘
                        └─E─┘

►──┬────────────────────────────────────────────────────────┬──────────────────────────►
   │           ┌─ALL──────┐      ┌─ANY─────────┐            │
   └─TRACK──(──┼─OPCASUB──┼──,───┼─READYFIRST──┼──)─────────┘
               └─JOBOPT───┘      └─READYONLY───┘

►──┬────────────────────────────────┬──┬───────────────────────────┬──────────────────►
   │                ┌─OCCNAME─┐     │  │             ┌─NO──┐       │
   └─TWSJOBNAME──(──┼─JOBNAME─┼──)──┘  └─USINRC──(───┴─YES─┴──)────┘
                    ├─EXTNOCC─┤
                    └─EXTNAME─┘

►──┬─────────────────────────────────┬──────────────────────────────────────────────────►
   │                      ┌─R─┐      │
   └─UX001FAILACTION──(───┴───┴──)───┘

►──┬─────────────────────────────────────────────────────────────────────┬─────────────►
   │               ┌─LEAVE───┐      ┌─LEAVE───┐      ┌─MANUAL─┐        │
   └─WSFAILURE──(──┼─ERROR───┼──,───┼─────────┼──,───┼────────┼──)─────┘
                   └─RESTART─┘      └─REROUTE─┘      └─IMMED──┘

►──┬─────────────────────────────────────────────────────────────────────┬─────────────►
   │               ┌─LEAVE───┐      ┌─LEAVE───┐      ┌─IMMED──┐        │
   └─WSOFFLINE──(──┼─ERROR───┼──,───┼─────────┼──,───┼────────┼──)─────┘
                   └─RESTART─┘      └─REROUTE─┘      └─MANUAL─┘

►──┬──────────────────────────────┬─────────────────────────────────────────────────────►
   │              ┌─YES─┐         │
   └─ZCENJSUB──(──┴─NO──┴──)──────┘

►──┬───────────────────────────────────────────────────┬───────────────────────────────►◄
   │              ┌─YES──────────────────────────┐     │
   └─ZCHIGHRC──(──┼─NO───────────────────────────┼──)──┘
                  └─DEF (HIGHEST NO ERROR RC)─────┘
```

Parameters

ALEACTION (*alert action limit* 100,*nnn* 000)

The limit to take an alert action for an operation in the current plan that is active for an unexpectedly long time (for more details, see the DURATION keyword in "ALERTS" on page 7). If you specify ALEACTION, its value is used to select the operations for which a long duration alert must be issued. If you do not specify ALEACTION, the value set for LIMFDBK is used instead.

The values for the alert action limit are in the range 100 through 999, or 0 if the alert action is to be taken as soon as possible after detecting that the operation is active longer than its estimated duration (a delay might occur for several reasons, for example heavy workload, Workstation Analyzer tasks, Event Manager tasks, or locking conditions).

With APAR PM1887 a second parameter is available. The value ranges from 000 to 999, and it represents a lower boundary time expressed in seconds, below which the Long Duration messages (EQQE028I and EQQE038I) are not logged even if the long duration policy defined by the aleaction value is matched. For example, assuming that ALEACTION (500,060), and that the planned duration for job is equal to 10 seconds, because of the ALEACTION settings, that job is considered to have a long duration when its actual duration becomes at least 50 seconds long. Because of the second option, the Long Duration message is issued only if the actual duration exceeds 60 seconds.

Note: Both of the parameters of this keyword must be always specified in 3-digit form.

BACKUP (NO|*backup frequency*|400)

IBM Workload Scheduler for z/OS uses a primary and alternate data set for the current plan. IBM Workload Scheduler for z/OS reorganizes the current-plan data set that is in use by copying it to the inactive data set and then switching to the newly copied data set. The value you specify on the BACKUP keyword defines if the current plan should be automatically copied, and determines how frequently the automatic copy process should occur.

Specify a backup frequency if you want IBM Workload Scheduler for z/OS to perform the copy process automatically. The backup frequency value defines how many records should be written to the job-tracking log before a copy process is performed. IBM Workload Scheduler for z/OS includes both tracked events and audit records when counting the number of records written to the job-tracking log. Specify NO if you do not want IBM Workload Scheduler for z/OS to perform the copy process automatically. If you specify NO, ensure that you request backups at regular intervals, depending on the workload at your installation.

You can request that IBM Workload Scheduler for z/OS performs a copy process using the BACKUP command or EQQUSIN or EQQUSINB subroutine, regardless of the value specified on the BACKUP keyword. For more information about the BACKUP command, see *Managing the Workload*. For more information about the EQQUSIN and EQQUSINB subroutines, see Chapter 6, "Reporting events to IBM Workload Scheduler for z/OS," on page 273. If the copy process is performed automatically and you issue a BACKUP request, IBM Workload Scheduler for z/OS counts the number of records from the time of the requested backup before performing another automatic copy.

Copying of the current plan data sets also occurs when the controller is started or stopped, before and after daily planning, and during recovery of IBM Workload Scheduler for z/OS system data sets. These copies occur regardless of the value specified for BACKUP.

CONDSUB (YES|NO)
Specify YES if condition dependencies defined on status S (started) have to be evaluated as soon as the status of the conditional predecessor becomes S (started) without waiting for the job-start event reported by the tracker component.

Specify NO if conditional dependencies defines on status S (started) must wait for the job-start event reported by the tracker component before being evaluated. NO is the default value.

CRITJOBS(NO|YES)
Specify CRITJOBS(NO) to prevent the creation of the critical job table and the start of the critical path handler task at controller startup, thus deactivating the critical path capability without resetting the critical operation option and running LTP and DP batch. Consider running with CRITJOBS(NO) in the following conditions:

- During recovery procedures.
- When the critical path capability does not fit the current workload execution scenario.

To reactivate the critical path capability, perform the following steps:

1. Scrap the EQQJTABL log data set, if it is not empty.
2. Restart the controller with CRITJOBS(YES).
3. Submit a replan job to re-synchronize the critical job table with the current plan.

CURRPLAN (NEW|CURRENT)
This keyword defines from which current-plan data set IBM Workload Scheduler for z/OS starts. The default is that IBM Workload Scheduler for z/OS uses either the primary current-plan data set (ddname EQQCP1DS) or the alternate current-plan data set (ddname EQQCP2DS). If both of these data sets are damaged or contain logical errors, if the EQQCKPT data set has been reallocated, or when IBM Workload Scheduler for z/OS is started for the first time after migration, IBM Workload Scheduler for z/OS can be started from the new-current-plan data set (ddname EQQNCPDS). To do so, specify CURRPLAN(NEW). Starting from the new-current-plan data set is done automatically if you have created a new plan while IBM Workload Scheduler for z/OS was inactive.

Use CURRPLAN(NEW) only when you cannot start IBM Workload Scheduler for z/OS using the primary or the alternate current-plan data set. Do not use CURRPLAN(NEW) when you start IBM Workload Scheduler for z/OS for the first time with all data sets empty.

DLIMFDBK (*limit for deadline feedback*|100)
The limit for deadline feedback. This keyword determines if the estimated deadline in the application description run cycle or operation is updated when an occurrence of the application reaches the complete status. The DLIMFDBK keyword value you set in this keyword is used only if you did not set a value in the application description.

Feedback values are in the range 100 through 999, or 0 if the deadline must be always updated, regardless of the estimated and actual values.

The feedback limits for *ADL* are calculated as follows:
- Lower limit = *ODL* * 100/*DLF*
- Upper limit = *ODL* * *DLF*/100

Where:

ADL The actual deadline, considered as the elapsed minutes between the IA® and the completion time of the occurrence or operation.

ODL The old deadline estimated for the run cycle or operation (considered as offset in minutes from the IA) currently stored in the application description database.

DLF The limit for deadline feedback.

When the deadline feedback limit is set to 100, no new estimated deadline is stored in the application description database and no missed feedback record is generated. If the actual deadline lies within the feedback limits, a smoothing factor is applied before the application description is updated. If the limit for deadline feedback is set to 0, the application description database is always updated, unless:

- A feedback limit is specified also in the application
- The smoothing factor does not allow the change

If the completion time occurs before the IA time, the deadline is not updated and a missed feedback record is generated.

When the occurrence is generated, an identifier of the run cycle that generates the occurrence is stored in the occurrence record. This identifier is used to determine which run cycle must be updated. If the application description or the occurrence input arrival was modified, the run cycle might no longer be matchable. In this case, the deadline is not updated and a missed feedback record is generated.

DSMOOTHING (*deadline smoothing factor*|50)

The deadline smoothing factor. It determines how much the actual deadline influences the new deadline estimated for a run cycle or operation in the application description database. The smoothing factor is applied only if the actual deadline lies within the deadline feedback limits. The DSMOOTHING keyword value is used only if you did not set a smoothing factor in the application description.

The smoothing factor is in the range 0 through 999. A value of 0 means that the deadline is not updated, a value of 100 means that the actual deadline replaces the existing estimated deadline. The new deadline is calculated as follows:

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

Where:

NDL The new deadline estimated for the run cycle or operation (considered as offset in minutes from the IA) to be stored in the application description database.

ODL The old deadline estimated for the run cycle or operation (considered as offset in minutes from the IA) currently stored in the application description database.

ADL The actual deadline, considered as the elapsed minutes between the IA and the completion time of the occurrence or operation.

DSF The deadline smoothing factor.

DUAL (YES|NO)
Specify YES if IBM Workload Scheduler for z/OS should perform dual logging of the job-tracking-log data sets (EQQJT*nn*). When it is started, IBM Workload Scheduler for z/OS opens data sets pointed to by the EQQDL*nn* ddnames in the controller JCL procedure. The suffix *nn* is a number from 01 to 99. The number of EQQJT*nn* data sets and EQQDL*nn* data sets must be the same.

Specify NO if IBM Workload Scheduler for z/OS should not write job-tracking information to dual data sets. NO is the default value.

ERRRES (*error code,...,error code*)
Defines a list of error codes that, for job-tracking purposes, result in an automatic reset of an operation. The operation is reset to status A (arriving) and contains the message *Error, automatically reset* in its operation details panel.

An error code can be:
- A 4-digit job or started-task return code (*nnnn*)
- A system abend code (S*xxx*)
- A user abend code (U*xxx*)
- An IBM Workload Scheduler for z/OS-defined code

Note:
1. IBM Workload Scheduler for z/OS converts the decimal value of a user abend code to a hexadecimal error code. For example, user abend 123 is shown as error code X'U07B'.
2. The OSEQ error code is a special case and cannot be reset by ERRRES.
3. With PQ87904 APAR, the ERRRES logic does not apply if the error code is generated by the EQQCLEAN step, that is a step inserted into a restarted job by the Restart and Cleanup function.
4. The ERRRES keyword applies also to operations that are run on z-centric agent workstations.
5. The only valid error codes are those listed in the appendix of *Managing the Workload*.

For example, a user submits a job outside of IBM Workload Scheduler for z/OS for which an operation exists in the current plan. The job abends with code S806, which is specified in the ERRRES list. IBM Workload Scheduler for z/OS sets the operation to status A. If the user then resubmits the job after correcting the error, IBM Workload Scheduler for z/OS again automatically tracks the job. The status of the operation is changed from A to S when the job is started.

An operation that has been automatically reset by ERRRES processing is not resubmitted by IBM Workload Scheduler for z/OS even if SUBMIT=Y is specified for the operation. Therefore, if an operation that is normally submitted by IBM Workload Scheduler for z/OS is reset, manually change the status to R (ready) using the MCP dialog, or through PIF, if you want IBM Workload Scheduler for z/OS to submit the job again.

If you stop and restart IBM Workload Scheduler for z/OS, or if a new daily plan has been created, operations that have been reset will have their error reset indication removed and will be eligible for submission.

ETT (YES|NO)
Specify YES if the event-triggered-tracking function should be initially active when IBM Workload Scheduler for z/OS is started. Specify NO if the ETT

function should not be initially active. Note that you can activate or deactivate ETT while IBM Workload Scheduler for z/OS is running, using the Service Functions dialog.

ETTGENSEARCH (NO|YES, JOBONLY|SRONLY)

The ETTGENSEARCH contains two keyword values:

- The first keyword value can be YES (default) or NO. Specify YES if the event-triggered-tracking function must search the SI file first for an exact match or for the best match hereafter. This is because the percent sign (%) or asterisk (*) can be used in the ETT criteria definition. Specify NO if the event-triggered-tracking function must search the SI file only for an exact match. Use this value when the SI file does not contain ETT criteria using the percent sign (%) or asterisk (*).
- The second keyword value is ignored if you specify NO as first keyword value. It defines whether the best generic match has to be applied only to the special resources events or only to the jobs events:

  JOBONLY
  The best generic match has to be applied only to job events.

  SRONLY
  The best generic match has to be applied only to special resource events.

  If the first keyword value is YES and the second keyword is not specified, the best generic match is applied to both job events and special resource events.

ETTNEWDEP (NO|YES)

Determines the input arrival time used by the scheduler when solving dependencies for occurrences that:

- Are being added through ETT.
- Correspond to applications defined with a run cycle referring to the period ETTRCY1. In this condition, to resolve dependencies the scheduler uses the input arrival time associated to the run cycle, instead of using the actual input arrival time, that is the time when the occurrence is added.

The ETTNEWDEP parameter affects the selection of any successor added in the current plan in the previous conditions. ID does not apply to the resolution of mandatory successors (because the successor intervals are created before the predecessor occurrence is added).

Specify YES to have the scheduler use the ETTRCY1 input arrival time both for the occurrences that are being added to the current plan and the candidate successors, provided that the successor is an occurrence added through ETT and corresponding to an application with run cycle referring to ETTRCY1.

Specify NO to have the scheduler use the ETTRCY1 input arrival time only for the occurrences that are being added to the current plan. In this case, for the candidate successors the scheduler uses the actual input arrival time.

EVELIM (*nnnn*)

This keyword defines how often statistic messages related to the STATMSG keyword are issued.

It is considered only if the value of STATIM is 0, and it defines the number of events that the event-manager task must process before issuing a new set of messages.

Valid values are from 0 to 9999.

If the current value of STATIM is 0 and the current value of EVELIM is 0, the statistics messages are issued every *n* events, where *n* is half the BACKUP value.

The value of EVELIM can be dynamically updated using the modify command, /F *procname*,EVELIM=*nnnn*.

FIRSTFDBK (YES|NO)

First feedback for duration. If you specify YES, every new job that you define in the AD database is updated with the actual duration at its first run, regardless of the estimated values. At the next run, the duration is updated according to the values that you set for LIMFDBK and SMOOTHING.

FTWJSUB (NO|YES)

Specify YES if IBM Workload Scheduler for z/OS should submit jobs running on fault-tolerant workstations. Specify NO if IBM Workload Scheduler for z/OS should not automatically submit jobs running on fault-tolerant workstations.

The job-submit option can be changed through the Service Functions dialog while IBM Workload Scheduler for z/OS is running.

HIGHRC (*highest no-error return code*|4)

Defines the highest error code that can be generated in an IBM Workload Scheduler for z/OS job or started task without causing IBM Workload Scheduler for z/OS to process the operation as having ended in error.

Note: With PQ87904 APAR, the HIGHRC logic does not apply when the error return code is generated by the EQQCLEAN step, that is a step inserted into a restarted job by the Restart and Cleanup function. In this case the operation status is set to ended in error.

HOSTJSUB (NO|YES)

Specify YES if IBM Workload Scheduler for z/OS should submit jobs, start started tasks, and issue write-to-operator messages for WTO operations. Specify NO if IBM Workload Scheduler for z/OS should not perform these functions automatically. This parameter is incompatible with JOBSUBMIT.

The job-submit option can be changed through the Service Functions dialog while IBM Workload Scheduler for z/OS is running.

ITOM (YES|NO)

YES specifies that IBM Tivoli Output Manager and IBM Workload Scheduler for z/OS are integrated so that the job logs of operations run by IBM Workload Scheduler for z/OS can be viewed and managed with Tivoli Output Manager.

With this configuration setup, IBM Workload Scheduler for z/OS inserts a particular string in the log of every operation. The string contains a `><TWS OCCURRENCE` heading followed by this information:

- ID of the application
- Number of the operation
- Input arrival date and time

For example:

```
//TWSEF020 JOB ACCT,TWS,CLASS=A,MSGCLASS=Q
//*><TWS OCCURRENCE-->DEVAPP 020 1311050201
```

This string is then located by Tivoli Output Manager, trimmed of the heading, and used as Output Manager archive name.

The integration must be configured also on Tivoli Output Manager. For details about how to enable IBM Workload Scheduler for z/OS to integrate with Tivoli Output Manager, see IBM Workload Scheduler for z/OS Managing the Workload.

JOBCHECK (NO|SAME|YES)
The JOBCHECK keyword specifies if and how IBM Workload Scheduler for z/OS checks the job card before submitting the job.

JOBCHECK(YES) means that IBM Workload Scheduler for z/OS checks the job card only for validity. If the job card is not valid, the job is not submitted. IBM Workload Scheduler for z/OS considers the job card to be valid if it is in the following format:

```
//jobname  JOB
```

jobname
Consists of 1 to 8 alphanumeric or national characters where the first character is alphabetic or national.

JOB Must be preceded and followed by at least one blank. If the job card is valid but the job name is not the same as the job name in the current IBM Workload Scheduler for z/OS operation, a warning message is written to the IBM Workload Scheduler for z/OS message log.

JOBCHECK(NO) means that the job card is not checked at all. IBM Workload Scheduler for z/OS submits the job without checking the job card.

Note: JOBCHECK(YES) and JOBCHECK(NO) allow IBM Workload Scheduler for z/OS to submit a job with a job name that does not match the job name in the current operation. This implies that IBM Workload Scheduler for z/OS is unable to track the status of the submitted operation correctly. Use JOBCHECK(SAME) if you need the status to be tracked.

JOBCHECK(SAME) means that the job card is checked for validity, and also checked to see that the job name is the same as the job name in the current IBM Workload Scheduler for z/OS operation. The job is submitted only if it has a valid job card with the job name matching that in the current operation. No checking is performed for operations that run on a workstation with a user-defined destination ID connected through TCP/IP or APPC.

JOBSUBMIT(NO|YES)
| Specify YES if IBM Workload Scheduler for z/OS should submit the jobs running on host, z-centric, dynamic, and remote engine workstations. Specify NO if IBM Workload Scheduler for z/OS should not automatically submit the jobs running on host, z-centric, dynamic, and remote engine workstations. This parameter is incompatible with HOSTJSUB and ZCENJSUB.

| The job-submit option can be changed through the Service Functions dialog while IBM Workload Scheduler for z/OS is running.

JTLOGS(*number of JT logs*|5)
Specifies the number of auditing logs that IBM Workload Scheduler for z/OS must open when it is started. The number must be a value in the range from 2 to 99, the default value is 5.

| The job-tracking log data sets are identified by the EQQJT*nn* ddname in the controller JCL procedure. If you use the extended auditing feature (by setting AMOUNT(EXTENDED) in the AUDIT statement), the extended auditing log data sets are identified by the EQQDB*nn* ddname in the controller JCL procedure.

| IBM Workload Scheduler for z/OS attempts to open data sets starting with
| EQQJT01 or EQQDB01 and continue for the number specified in the JTLOGS
| keyword. For example, if you specify a value of 3 for JTLOGS, IBM Workload
| Scheduler for z/OS attempts to open logs EQQ*xx*01, EQQ*xx*02, and EQQ*xx*03,
| where *xx* can be JT or DB.

LARGEUSERBUFFER(NO|YES)

The default value (YES) allocates memory buffers sized to 64KB in the common storage area (CSA) to improve the communication rate with the IBM Workload Scheduler for z/OS engine from all user interfaces (ISPF, DWC, or PIF). This improves performances when running sizeable queries on the plan, but be aware that 64KB are allocated for every connected user, and you must weight the impact of this on your environment (for example, 300 concurrent users consume 19MB of CSA).

Specify NO if you want to keep 32KB as the size of the allocated memory buffers. This is the size used by default until Version 8.6 SPEs.

LIMFDBK(*limit for duration feedback*|100)

Limit for duration feedback. This parameter is ignored for shadow jobs.

IBM Workload Scheduler for z/OS job tracking automatically monitors actual durations. These can be used to modify estimated operation durations in the application description database. IBM Workload Scheduler for z/OS uses two factors, limit for duration feedback and duration smoothing, to control how actual durations are used.

The LIMFDBK value determines if estimated durations in the application description are updated when an occurrence of the application reaches complete status. The LIMFDBK keyword value is used only if no value is specified in the application description.

Feedback values are in the range 100 through 999, or 0 if the duration must be always updated, regardless of the estimated and actual values. The feedback limits are calculated as follows:

Limits for duration feedback

```
Lower limit = OD * 100/LF
Upper limit = OD * LF/100
```

where:

OD	The old estimated duration currently stored in the application description database.
LF	The limit for duration feedback.

If the limit for duration feedback is set to 0, the application description database is always updated, unless:

- A feedback limit is specified also in the application
- The smoothing factor does not allow the change

If an estimated duration lies within feedback limits, a smoothing factor is applied before the application description is updated. See the SMOOTHING keyword, which is described in the list of JTOPTS "Parameters" on page 80.

Table 5 shows examples of how the limit-for-feedback algorithm works.

Table 5. Limit for feedback examples

LF value	Result
100	No new estimated duration are stored in the application-description database.

Table 5. Limit for feedback examples (continued)

LF value	Result
110	The new estimated duration is stored if the actual duration is approximately between 90% and 110% of the old estimated duration.
200	The new estimated duration is stored if the actual duration is between half and double the old estimated duration.
500	The new estimated duration is stored if the actual duration is between one-fifth and five times the old estimated duration.
999	The new estimated duration is stored if the actual duration is between one-tenth and 10 times the old estimated duration.

The feedback limit used to select the operations for which a long duration alert must be issued is the value specified for ALEACTION. If ALEACTION is not set, the value for LIMFDBK is used instead. In this case, the value for the feedback limit that you can optionally enter in the application description is ignored.

MAXJSFILE(NO|*maximum size of JS data set*|0)

IBM Workload Scheduler for z/OS uses a primary and alternate data set for the JCL repository. IBM Workload Scheduler for z/OS reorganizes the JCL repository data set that is in use by copying it to the inactive data set and then switching to the newly copied data set. The value you specify on the MAXJSFILE keyword defines whether the JCL repository should be automatically copied and determines how frequently the automatic copy process should occur.

Specify a maximum size if you want IBM Workload Scheduler for z/OS to copy automatically. This value also defines how large the current JCL repository data set is allowed to become before it is automatically copied to the alternate data set. The size must be specified in megabytes (1MB equals 1,024 kilobytes). The maximum value you can specify is 67 108 864 megabytes. Any greater values produce unpredictable results. The value specified is converted into cylinders and rounded to the next whole number. Any value equivalent to less than 2 cylinders (other than the default value) is set to 2 cylinders. If you do not specify MAXJSFILE or specify the default value 0, IBM Workload Scheduler for z/OS performs a copy after the first 50 jobs have been inserted since it was started. The size of the data set (converted into cylinders) after this first copy, plus the equivalent of one cylinder, is then used as the value for MAXJSFILE. After every 50 inserts, IBM Workload Scheduler for z/OS checks the size of the JS file using an algorithm that is based on the high_used_RBA. If the high_used_RBA is equal to or greater than the value of MAXJSFILE, a copy is performed. Specify NO if you do not want IBM Workload Scheduler for z/OS to copy automatically. If you specify NO, ensure that you request backups at regular intervals, depending on the workload at your installation.

You can request that IBM Workload Scheduler for z/OS performs a copy process using the BACKUP command or EQQUSIN or EQQUSINB subroutine, regardless of the value specified on the MAXJSFILE keyword. For more information about the BACKUP command, see *Managing the Workload*. For more information about the EQQUSIN and EQQUSINB subroutines, see Chapter 6, "Reporting events to IBM Workload Scheduler for z/OS," on page 273.

MAXOCCNUM(*nnnnnnn*|32767)

IBM Workload Scheduler for z/OS maintains an upper limit on the number of occurrences in the current plan. When this limit is reached, no more occurrences can be added, either by dialog, the program interface, the event triggered tracking, or the automatic recovery. If the keyword is omitted, the limit is 32767 occurrences. It is advisable not to set the parameter to a larger number than required by actual workload needs, due to the increased overhead incurred. Doubling the default value of MAXOCCNUM, setting it to 65534, should not cause any noticeable performance problems; however any change to values greater than this number must be done gradually. IBM Workload Scheduler for z/OS can start with a current plan exceeding the limit and also take over a plan exceeding the limit from daily planning.

MCPDATASPACE(YES|NO)
Specify YES for the Modify Current Plan to load portions of the in-storage operations and occurrences into the data space, when performing MCP actions.

Setting this parameter to YES is particularly helpful when you modify operations and occurrences belonging to big networks, because it optimizes the use of storage. If the current plan was generated with BATCHOPT CPDATASPACE(YES), this parameter must also be set to YES.

With a current plan that includes more than one million operations, you must also allocate the following CP data sets as extended VSAM:

- EQQACPDS
- EQQCP1DS
- EQQCP2DS
- EQQNCPDS
- EQQSCPDS

For more details about managing a current plan with one million jobs, see *Managing the Workload*.

NEWOILIMIT(*days operator instructions are new*|30)
Defines the number of days that IBM Workload Scheduler for z/OS considers an operator instruction record to be new after it is changed. The number of days between the occurrence input arrival and the last update of the operator instruction is calculated. If the result is less than the value specified for NEWOILIMIT, or if the occurrence input arrival is earlier than the last update of the operator instruction, the operator instruction is treated as a new instruction. New operator instructions are represented in tailorable lists by a plus character (+) in the OI column.

NOERROR(*error code entry*,...,*error code entry*)
Defines a list of error codes that, for job-tracking purposes, are treated as normal completion codes. You can also specify error codes on the NOERROR statement. See "Purpose" on page 106.

This parameter follows the same rule as the LIST parameter of the NOERROR statement. For a description of these rules, see the list of NOERROR "Parameters" on page 107.

Note: Do not use this parameter to dynamically rebuild the NOERROR table using a modify command with the NEWNOERR or NOERRMEM option. If you need to dynamically rebuild the NOERROR table, use the NOERROR statement as described in "Purpose" on page 106.

OFFDELAY(*delay time*|1)
The OFFDELAY parameter defines, in minutes, the time that IBM Workload Scheduler for z/OS delays actions defined in the WSOFFLINE parameter when a workstation changes status to offline. The status of the workstation changes immediately as a response to the offline event being received at the controller, but IBM Workload Scheduler for z/OS does not take reroute or restart actions

until the time specified for OFFDELAY has elapsed. If an event that changes the status of the workstation to available is received during the delay time, no WSOFFLINE actions are performed.

OFFDELAY is used only when a workstation changes status to offline, not for a failure indication. The OFFDELAY parameter also functions as the delay time for setting a workstation to offline during IBM Workload Scheduler for z/OS controller startup. The controller initially keeps the status of the workstation as it was when the controller subsystem was stopped. The OFFDELAY parameter defines the length of time that the controller waits for a tracker to establish communication. If it is not performed during the specified time, the workstation represented by this tracker is set to OFFLINE status.

Note: If you have workstations that specify a user-defined destination ID, ensure that the OFFDELAY keyword is set high enough to allow sufficient time to set the destination to active status when IBM Workload Scheduler for z/OS is started.

OPINFOSCOPE(ALL|IP)

Defines how IBM Workload Scheduler for z/OS selects an operation when an event is created that updates the USERDATA field. The event can be created through an OPINFO TSO command, EQQUSIN or EQQUSINO subroutine, or API CREATE request.

Specify IP (in progress), which is the default, if IBM Workload Scheduler for z/OS should select the operation only from operations in status R, A, *, S, I, and E. If there is more than one operation that matches the selection criteria, IBM Workload Scheduler for z/OS chooses the operation by investigating these characteristics in the stated order:

1. The operation has priority 9.
2. Earliest latest start time.
3. Priority 8-1.
4. Input arrival time specified for the operation or the occurrence input arrival if the operation does not have input arrival specifically defined.
5. Longest in Ready status.

Specify ALL if IBM Workload Scheduler for z/OS should also check operations in status C and W, if no matching in-progress operation was found. The operation with the earliest latest-start-time is selected.

OPREROUTEDEFAULT(N|Y)

Defines the default for operations that have a blank value specified for the reroutable option in the operation details.

Specify N if operations that have reroutable set to blank are not eligible for rerouting if the workstation becomes inactive. Specify Y if ready operations should be rerouted to the alternate workstation if a blank value is specified, and the installation default action allows operations to be rerouted when the workstation status is set to failed or offline. The default action can be specified in:

- The MCP dialog when the workstation is manually varied to offline or failed.
- The WSSTAT command or EQQUSIN or EQQUSINW subroutine when the workstation is set to offline or failed.
- The second value of the WSOFFLINE or WSFAILURE keywords on the JTOPTS statement. This default applies to all workstations.

OPRESTARTDEFAULT(N|Y)
Defines the default for operations that have a blank value specified for the restartable option in the operation details.

Specify N if operations that have restartable set to blank are not eligible for automatic restart if the workstation becomes inactive. Specify Y if started operations should restart on the alternate workstation if a blank value is specified, and the installation default action allows operations to be restarted when the workstation status is set to failed or offline. The default action can be specified in:

- The MCP dialog when the workstation is manually varied to offline or failed.
- The WSSTAT command or EQQUSIN or EQQUSINW subroutine when the workstation is set to offline or failed.
- The first value of the WSOFFLINE or WSFAILURE keywords on the JTOPTS statement. This default applies to all workstations.

OUTPUTNODE(ANY|FINAL)
Defines whether IBM Workload Scheduler for z/OS should process A3P (JES2 job termination) events from any NJE node that job SYSOUT is spooled to, or from only the NJE node that is the final destination. The OUTPUTNODE keyword is valid only for JES2 environments.

Because you can send JES2 job SYSOUT, or parts of the SYSOUT, to several different NJE nodes, more than one job termination (A3P) event could be produced for the same job. Also, if the job completion checker (JCC) is used, each event can also have different job-completion-code information depending on the output sent to a particular node and the checking that the JCC performs at that node. The status assigned to the operation depends on which of the A3P events was first processed by the controller. Specify FINAL if you use the JCC to scan SYSOUT and set error codes. Then, only the part of the SYSOUT that contains the JESYSMSG (previously $SYSMSGS, DSID=4), which has reached its final NJE node, is used to change the status of the corresponding computer operation from status S (started) to status C (complete) or E (ended in error). FINAL is the default value.

Specify ANY if the JCC is not used to scan SYSOUT and set error codes. OUTPUTNODE defaults to FINAL if RCLEANUP(YES) is specified in the OPCOPTS statement.

If SYSOUT is routed to an NJE node that is not controlled by IBM Workload Scheduler for z/OS, the A3P event from the executing node is used to change the status of the corresponding operation, regardless of the value you specify for OUTPUTNODE.

OVERCOMMIT(*nnnn*|0)
Defines the number of job, started-task, and WTO operations that can be started on the automatically reporting workstations besides the number specified as the parallel server capacity for the workstation. For example, if a computer workstation has 10 parallel servers defined and OVERCOMMIT specifies 2, then up to 12 operations can be started for that workstation.

The workstation must use control on parallel servers for OVERCOMMIT to have meaning. The default value is 0, maximum 9999.

PLANSTART(*planning period start*|6)
Defines the time-of-day in hours when the daily planning period starts. This value must be the same as the value you specify for PLANHOUR on the BATCHOPT statement.

PRTCOMPLETE(NO|YES)
Specify YES if IBM Workload Scheduler for z/OS should set print operations to complete when a batch job is purged from the JES spool. Specify NO if IBM Workload Scheduler for z/OS should not set print operations to complete when a batch job is purged from JES. Here, print operations are set to complete only by print-end events.

Consider setting PRTCOMPLETE to YES if some of your jobs or started tasks conditionally create SYSOUT, or if FREE=CLOSE is specified on the DD statement.

QUEUELEN(*queue length*|5)
Defines the maximum number of ready operations that the workstation analyzer (WSA) subtask starts each time it has control of the current plan resource. The default value is 5. If you specify a value less than 5, the default value is used.

If you specify a high value for QUEUELEN and there are many ready operations, this could affect the performance of other tasks that use the current plan resource.

The value of QUEUELEN can be dynamically updated using the modify command, `/f` *procname*`,QUELEN=`*nnnn*

RECCPCOMPL(NO|YES)
Set RECCPCOMPL(N) to avoid path recalculation when an operation on the critical path completes and its successor on the same critical path has an uncompleted predecessor.

Use the default RECCPCOMPL(Y) to have the critical paths recalculated for all the available recalculation triggers.

RISKCONFIDENCE(1-99)
The value of this keyword influences the trigger that sets the high risk level for a critical job.

When the confidence value of a critical job is lower than the RISKCONFIDENCE value, the critical path handler task sets the critical job to a high risk level and notifies it to the controller EQQMLOG. If this keyword is not specified, the critical path handler task sets the critical job to a high risk level when its estimated end time becomes later than the deadline.

Setting RISKCONFIDENCE to 50 generates the closest behavior to the versions earlier than V9.3, that is a high risk level is set when the estimated end time becomes later than the deadline.

SHUTDOWNPOLICY(*nnn*|0)
The SHUTDOWNPOLICY value is a percentage in the range 0 to 999. It lets you specify whether IBM Workload Scheduler for z/OS should start an operation when there is little time left before a workstation is closed. A workstation must have `CONTROL ON SERVERS=Y` for this keyword to have any effect.

The estimated duration of an operation is multiplied by the SHUTDOWNPOLICY percentage, and the result is compared to the time remaining in the workstation-open interval. If the result is greater than the time remaining in the open interval for the workstation and a nonzero factor is specified, IBM Workload Scheduler for z/OS does not start the operation.

The following examples show how SHUTDOWNPOLICY is used. In these examples, an operation has an estimated duration of 60 minutes, and the workstation it uses will close down in 45 minutes.

SHUTDOWNPOLICY(0)
The operation is started regardless of the end of the current workstation-open interval.

SHUTDOWNPOLICY(50)
The operation is started because 30 minutes (50% of 60 minutes) is less than the 45 minutes remaining in the workstation-open interval.

SHUTDOWNPOLICY(100)
The operation is not started because 60 minutes (100% of 60 minutes) is greater than the time remaining in the workstation-open interval.

SHUTDOWNPOLICY(200)
The operation is not started because 120 minutes (200% of 60 minutes) is greater than the time remaining in the workstation-open interval.

SMOOTHING(*smoothing factor***|**<u>**50**</u>**)**
The smoothing factor determines how much the actual duration of an operation influences the new estimated duration that is stored in the application description database. The smoothing factor is applied only if the actual duration lies within the limits determined by feedback (see the LIMFDBK keyword in the list of JTOPTS "Parameters" on page 80). This parameter is ignored for shadow jobs.

| **Note:** When the controller has the Dynamic Critical Path feature active, any
| SMOOTHING value that is greater than 100 is internally managed as if the
| smoothing factor default value was set (50). For example, if the smoothing
| factor is set to 200 and the Dynamic Critical Path is active, the affected
| durations will be updated by applying the SMOOTHING default value 50.

IBM Workload Scheduler for z/OS uses the value you specify on the SMOOTHING keyword if you do not specify a smoothing factor in the details of an operation. The smoothing factor is in the range 0 to 999. A value of 0 means that the operation is not updated. A value of 100 means that the actual duration replaces the existing estimated duration of the operation. The new estimated duration is calculated as follows:

```
New estimated duration
ND = OD + ((AD - OD) * SF/100)
```

where:

ND	The new estimated duration to be stored in application description database.
OD	The old estimated duration currently stored there.
AD	The actual duration.
SF	The smoothing factor.

Table 6 provides examples of how the smoothing factor algorithm works.

Table 6. Smoothing factor examples

Factor	Result
0	There is no feedback.
10	The new estimated duration is the old estimated duration, plus one-tenth the difference between the actual and old estimated duration.
50	The new estimated duration is the old estimated duration, plus one-half the difference between the actual and old estimated duration.
100	The actual duration replaces the old estimated duration.

Table 6. Smoothing factor examples (continued)

Factor	Result
999	The new estimated duration is the old estimated duration, plus 10 times the difference between the actual and old estimated duration.

STATMSG(*option list*)
Defines the status message types that IBM Workload Scheduler for z/OS will produce. You can specify one or more of the these values:

CPLOCK
The event-manager subtask issues messages EQQE004I and EQQE005I, which describe how often different tasks have referenced the current-plan data set.

EVENTS
The event-manager subtask issues messages EQQE000I, EQQE006I, and EQQE007I, which describe how many events were processed and provide statistics for different event types.

WSATASK
The event manager task issues messages EQQE008I and EQQE009I, which describe statistic information collected by the WSA task.

All these messages are issued according to the following criteria:

- If STATIM has been set to a value different from 0 (by specifying STATIM(*n*) in the JTOPS keyword, or by using the modify command `/f` *procname*`,STATIM=`*n*), the message is issued approximately every *n* minutes, if any events have been processed.
- Otherwise, if EVELIM has been set to a nonzero value (by specifying EVELIM(*n*) in the JTOPS keyword, or by using the modify command `/f` *procname*`,EVELIM=`*n*), the message is issued approximately every *n* events.
- Otherwise, the message is issued approximately once every *n* events, where *n* is half the JTOPTS BACKUP keyword value (default BACKUP value is 400).

GENSERV
The general-service subtask issues messages EQQG010I to EQQG013I, which describe how often different tasks have been processed and how long the general-service queue has been. IBM Workload Scheduler for z/OS issues these messages every 30 minutes, or every *n* minutes if the value of STATIM is nonzero (by specifying STATIM(*n*) in the JTOPTS keyword, or by using the modify command `/f` *procname*`,STATIM=`*n*), if any requests have been processed.

For more information about any of these messages, refer to *Messages and Codes.*

STATIM(*nn*)
Defines the time interval, in minutes, to be used for issuing the statistic messages related to the STATMSG keyword. Valid values are from 0 to 99.

If this keyword is omitted, or if 0 is specified, the time interval is not used, and statistic messages are issued every *n* events, where *n* is the EVELIM values or half of the BACKUP value if EVELIM is set to 0. The value for STATIM can be dynamically updated using the modify command, `/f` *procname*`,STATIM=`*nn*.

STEPINFO(YES|NO)

If you specify YES, the primary controller stores the list of step events that was generated and sent by the tracker, enabling you to browse the information. STEPINFO(YES) is meaningful only if you set EWTROPTS STEPINFO(YES) on the tracker.

If you specify NO the list of step events is only logged on the primary controller, which, if a backup controller is configured, sends it to the backup controller, where the data is stored.

Set STEPINFO(YES) on the backup controller to ensure that when the backup controller takes over from the primary controller, you can browse the list of steps for all the jobs that are in plan.

SUBFAILACTION(C|E|R|RH|XC|XE|XR)

Defines what action IBM Workload Scheduler for z/OS should take if a failure occurs while retrieving the JCL during the submission of a job or the starting of a started task. Specify one of these actions:

C	The operation is set to complete. If EQQUX001 is called and returns an error code, then the error code is ignored.
E	The operation is set to ended-in-error with error code OSUF. If EQQUX001 is called and returns an error code, then the error code is ignored.
R	The operation remains on the ready list with status R (ready) and extended status E (error). If the failure was reported by the operation-initiation exit (EQQUX009), the operation remains on the ready list with status S (started) and extended status E (error). If EQQUX001 is called and returns an error code, then the error code is ignored.
RH	This acts in the same way as **R**, unless EQQUX001 is called and returns an error code different from '0000', in which case the operation is set to ready and manual hold.
XC	This acts in the same way as **C**, unless EQQUX001 is called and returns an error code different from '0000', in which case the operation ends in error with the error code returned by the user exit.
XE	This acts in the same way as **E**, unless EQQUX001 is called and returns an error code different from '0000', in which case the operation ends in error with the error code returned by the user exit.
XR	This acts in the same way as **R**, unless EQQUX001 is called and returns an error code different from '0000', in which case the operation ends in error with the error code returned by the user exit.

SUPPRESSACTION(C|E|R)

Defines what action IBM Workload Scheduler for z/OS should take if a suppress-if-late time-dependent operation becomes late. Specify one of these actions:

C	The operation is set to complete.
E	The operation is set to ended-in-error with error code OSUP.
R	The operation remains on the ready list with status R (ready) and extended status L (late).

The suppress actions are not all applicable to the operations defined on fault-tolerant workstations in an end-to-end with fault tolerance capabilities environment: for operations with the centralized script option, the applicable suppress actions are C, E, and R; for the other operations defined on fault-tolerant workstations, the applicable suppress action is only R. If a different value is specified, the default value R is used for these operations.

Note:

1. For operations running on fault-tolerant workstations, status E is allowed only for centralized scripts. However, for the controller and fault-tolerant agent the status does not match until a batch DP job (Symphony Renew, CP extend, CP replan) is run.
2. IBM Workload Scheduler for z/OS considers the time buffer created by the SUPPRESSPOLICY keyword when deciding if a time-dependent operation is late.

Note:

SUPPRESSPOLICY(*nnn*|0)

Specifies how IBM Workload Scheduler for z/OS should control time-dependent operations with the suppress-if-late option. You can use SUPPRESSPOLICY to create an input-arrival buffer for these time-dependent operations rather than a specific input-arrival time.

The SUPPRESSPOLICY value is a percentage in the range 0 to 999. A value of 0 means that IBM Workload Scheduler for z/OS does not try to start the operation if its input-arrival time has passed. If you specify a nonzero value, IBM Workload Scheduler for z/OS multiplies the estimated duration of the operation by this percentage. If the resulting figure lies within the time remaining until the operation deadline, the operation is started. Otherwise, the operation is considered late and is not started by IBM Workload Scheduler for z/OS.

The following examples show how SUPPRESSPOLICY is used. In these examples, an operation has become ready. The input-arrival time of the operation passed 15 minutes ago, and its deadline will expire in 45 minutes. The operation has an estimated duration of 60 minutes.

SUPPRESSPOLICY(0)

The operation is considered late because the input-arrival time has passed. The action specified on the SUPPRESSACTION keyword is taken.

SUPPRESSPOLICY(50)

The operation is started because 30 minutes (50% of 60 minutes) is less than the 45 minutes remaining to the deadline.

SUPPRESSPOLICY(100)

The operation is considered late because 60 minutes (100% of 60 minutes) is greater than the time remaining to the deadline. The action specified on the SUPPRESSACTION keyword is taken.

SUPPRESSPOLICY(200)

The operation is considered late because 120 minutes (200% of 60 minutes) is greater than the time remaining to the deadline. The action specified on the SUPPRESSACTION keyword is taken.

The suppress-if-late option on operations defined on fault-tolerant workstations with no centralized script option and not using any special resource is not handled directly by the Controller, but is stored in the Symphony® file as "until time", see the *IBM Workload Scheduler for z/OS: Reference Guide*. The until time value is set using the SUPPRESSPOLICY with the same algorithm as for the other operations if the expected operation end time is earlier than the deadline time, otherwise it is set to the operation input arrival time plus one minute.

TRACK(OPCASUB|JOBOPT|ALL, READYFIRST|READYONLY|ANY)
The TRACK parameter contains two keyword values:

- The first value specifies which jobs or started tasks IBM Workload Scheduler for z/OS tracks. If a job is tracked, it means that events for that job cause the current plan to be updated. The status of the operation in the current plan that represents the job is updated by events such as *job start* and *job completion*. For example, when a job ends execution on a system (computer workstation), the status of the corresponding operation in the current plan is changed from S (started) to C (completed).

  If you use event-triggered tracking with job-name replace (J-type events with JR=Y), both operands of the TRACK parameter are ignored for jobs submitted from outside of IBM Workload Scheduler for z/OS. Such jobs are always tracked.

  Specify OPCASUB when all your IBM Workload Scheduler for z/OS-planned jobs and started tasks are submitted by IBM Workload Scheduler for z/OS; that is, when they are all defined with the SUBMIT option set to YES. This is the recommended method of handling submission. IBM Workload Scheduler for z/OS tracks work that it submitted itself; work submitted outside of IBM Workload Scheduler for z/OS is not tracked, even if it corresponds to an operation that is defined in the current plan.

  Specify JOBOPT when some of your IBM Workload Scheduler for z/OS-planned jobs are submitted by IBM Workload Scheduler for z/OS and some are submitted outside of IBM Workload Scheduler for z/OS, but you always know in advance which are submitted by each method. You must ensure that:

  - Jobs submitted by IBM Workload Scheduler for z/OS have their SUBMIT option set to YES, and these jobs are *not* submitted outside of IBM Workload Scheduler for z/OS.
  - Jobs submitted outside of IBM Workload Scheduler for z/OS have their SUBMIT option set to NO.

  When you use JOBOPT, IBM Workload Scheduler for z/OS tracks jobs it submitted itself, and jobs that were submitted outside of IBM Workload Scheduler for z/OS but were defined with their SUBMIT option set to NO. JOBOPT causes IBM Workload Scheduler for z/OS to track jobs where the mode of submission correctly matches the SUBMIT option for the job. If you define a job with the SUBMIT option set to YES but the job is submitted outside of IBM Workload Scheduler for z/OS, the job is *not* tracked.

  Specify ALL when you do not know in advance if IBM Workload Scheduler for z/OS-planned jobs will be submitted by IBM Workload Scheduler for z/OS or outside of IBM Workload Scheduler for z/OS. All IBM Workload Scheduler for z/OS-planned jobs are tracked, regardless of how they were submitted or what the job SUBMIT option is set to.

- The second keyword value determines how IBM Workload Scheduler for z/OS selects the matching operation when a job or started task is submitted outside of IBM Workload Scheduler for z/OS. The product uses this value only when you specify JOBOPT or ALL for the first keyword value.

  Specify READYFIRST to select the ready operation with the earliest latest-start time. If no ready operation is found, IBM Workload Scheduler for z/OS selects the waiting operation with the earliest latest-start time. IBM Workload Scheduler for z/OS sets the status of this operation to E (ended in error) with error code OSEQ.

Specify READYONLY to select the ready operation with the earliest latest-start time. If no ready operation is found, the job or started task is not tracked by IBM Workload Scheduler for z/OS.

Specify ANY to select the operation with the earliest latest-start time, which is in either waiting or ready status. ANY is the default value.

TWSJOBNAME(EXTNAME|EXTNOCC|JOBNAME|OCCNAME)

Defines the criterion used to generate the job name in the Symphony file.

If you set OCCNAME, the job name in the Symphony file is made up according to either of the following formats:

X_Num_ApplicationName
    When the job is created.

X_Num_Ext_ApplicationName
    When the job is deleted and re-created in the current plan.

where:

X Can be one of the following values:
- `J`, for normal operations
- `P`, for jobs representing pending predecessors
- `R`, for recovery jobs

Num The operation number.

Ext A sequential decimal number that is increased every time an operation is deleted and re-created

ApplicationName
    The name of the occurrence to which the operation belongs.

If you set EXTNAME, EXTNOCC, or JOBNAME the job name in the Symphony file is made up according to either of the following formats:

X_Num_JobInfo
    When the job is created.

X_Num_Ext_JobInfo
    When the job is deleted and re-created in the current plan.

where:

X Can be one of the following values:
- `J`, for normal operations
- `P`, for jobs representing pending predecessors
- `R`, for recovery jobs

For jobs representing pending predecessors, the job name is in all cases generated by using the OCCNAME criterion. This is because, in the case of pending predecessors, the current plan does not contain the required information (excepting the name of the occurrence) to build the Symphony name according to the other criteria.

Num The operation number.

Ext The hexadecimal value of a sequential number that is increased every time an operation is deleted and re-created

JobInfo Depends on the chosen criterion:

For EXTNAME
    JobInfo is filled with the first 32 characters of the extended job

name associated to that job (if it exists) or with the 8-character job name (if the extended name does not exist). Note that the extended job name, in addition to being defined in the database, must also exist in the current plan.

For EXTNOCC
JobInfo is filled with the first 32 characters of the extended job name associated to that job (if it exists) or with the application name (if the extended name does not exist). Note that the extended job name, in addition to being defined in the database, must also exist in the current plan.

For JOBNAME
JobInfo is filled with the 8-character job name.

The criterion used to generate an IBM Workload Scheduler job name are maintained throughout the entire life of the job.

To choose the EXTNAME, EXTNOCC, or JOBNAME criterion, the EQQTWSOU data set must have a record length of 160 bytes. Before using any of the above keywords, you must migrate the EQQTWSOU data set. Sample EQQMTWSO is available to migrate this data set from 120 to 160 bytes.

Limitations when using the EXTNAME and EXTNOCC criteria:

- The job name in the symphony file can contain only alphanumeric characters, dashes and underscores. All the other characters accepted for the extended job name are converted into dashes. Note that a similar limitation applies also with JOBNAME: when defining members of partitioned data sets (such as the script or the job libraries), national characters can be used, but they are converted into dashes in the symphony file.
- The job name in the symphony file must be in uppercase. All lowercase characters in the extended name are automatically converted to uppercase by IBM Workload Scheduler for z/OS.

Note: Using the job name (or the extended name as part of the job name) in the symphony file implies that it becomes a key for identifying the job. This means also that the extended name or job name is used as a key for addressing all the events directed to the agents. For this reason, be aware of the following facts for the operations included in the symphony file:

- Editing the extended name is inhibited for operations created when the TWSJOBNAME keyword is set to EXTNAME or EXTNOCC.
- Editing the job name is inhibited for operations created when the TWSJOBNAME keyword is set to EXTNAME or JOBNAME.

| **USINRC(YES|NO)**
| Specify YES if you want that the ERROR_CODE field set by a status change
| program (for example, the EQQUSIN subroutine invoked by System
| Automation) is processed, even if the operation status was changed to C
| (Complete). The error code, when set, is used to evaluate any conditional
| dependency defined for the operation.

| The default value is No, meaning that the ERROR_CODE field set by a status
| change program for a complete operation is always ignored.

UX001FAILACTION(R)
This keyword is specified when the EQQUX001 exit is involved.

If EQQUX001 is called and returns an error code different from `0000`, operations remain in ready list with extended status E. The only value of UX001FAILACTION is R. The UX001FAILACTION and SUBFAILACTION(XR/XE) keywords are exclusive.

`WSFAILURE(ERROR|RESTART|LEAVE, REROUTE|LEAVE, IMMED|MANUAL)`

Defines the actions to be taken when a workstation failure occurs. A workstation is set to failed status either when there is no communication with the z/OS system or if it is set manually in the IBM Workload Scheduler for z/OS dialogs. Workstations that specify a user-defined destination ID can have failed status reported by the WSSTAT command, or EQQUSIN or EQQUSINW subroutine.

The WSFAILURE parameter contains three keyword values:

- The first keyword value defines the restart actions to be taken when a workstation fails and the started operation is restartable. Specify ERROR to set started operations to ended-in-error status. The error code will be OSSI, OSSQ, OSSS, or OSSC. Operations whose error code is OSSS have a stepcode of OSYS. Specify RESTART to immediately reset the started operations at this workstation to ready. Specify LEAVE to leave started operations on a failed workstation in started status; this is the default value.

  Note:
  1. If you set the restartable option of an operation to NO, the operation is not processed. It remains in the started status.
  2. Consider this note if you use a standby controller, running with the TAKEOVER(HOSTFAIL) parameter in the XCFOPTS statement. If you selected the WSFAILURE(RESTART,REROUTE,IMMED) options and the controller abnormally ends or is cancelled, while the LPAR that the controller is running on remains active, jobs might be submitted again even though they are currently running, resulting in the same job being run twice.
  3. For dynamic and remote engine workstation types, this keyword supports only the value LEAVE. If you specify any other value, it is forced to LEAVE.
- The second keyword value defines reroute actions for workstation failure situations. Specify REROUTE to route operations, whose reroutable option is YES, to the alternate workstation. Specify LEAVE to leave operations to be scheduled on the original workstation; this is the default value. Rerouting does not occur for these operations.
- The third keyword value defines the action to be taken when a workstation becomes active again after a failure situation. Specify IMMED to automatically set the status of the workstation to available and withdraw any rerouting *immediately* when an event indicates that the workstation is operational. Specify MANUAL to indicate that the status of the workstation should be changed manually when a *workstation available* indication is received; this is the default value. IBM Workload Scheduler for z/OS issues an MLOG message to inform the operator that the event has been received.

`WSOFFLINE(ERROR|RESTART|LEAVE, REROUTE|LEAVE, MANUAL|IMMED)`

Defines the actions that are to be taken when a workstation offline situation occurs (unless the workstation is 'waiting for connection' at start and no previous offline situation occurred). This means that the controller cannot communicate with the tracker at the destination defined for the workstation. This might occur because the tracker has not been started yet (having experienced a previous offline condition status) or has ended abnormally, or because the controller has not received an ID event from the destination for two consecutive pulse intervals. Pulse intervals are specified by the PULSE keyword of ROUTOPTS.

Workstations that specify a user-defined destination ID are set to offline status when IBM Workload Scheduler for z/OS is started. Offline status for these workstations can also be reported by the WSSTAT command or the EQQUSIN or EQQUSINW subroutine.

The WSOFFLINE parameter contains three keyword values:

- The first keyword value defines restart actions for a workstation whose status has been changed to offline. Specify ERROR to set started operations, whose restartable option is YES, to ended-in-error status. The error code will be OFSI, OFSQ, OFSS, or OFSC. Operations whose error code is OFSS, have a stepcode of OFFL. Specify RESTART to immediately reset the started operations at this workstation to ready. Specify LEAVE to leave started operations at an offline workstation in started status; this is the default value.

  Note:

  1. If you set the restartable option of an operation to NO, the operation is not processed. It remains in the started status.
  2. Consider this note if you use a standby controller, running with the TAKEOVER(HOSTFAIL) parameter in the XCFOPTS statement. If you selected the WSOFFLINE(RESTART,REROUTE,IMMED) options and the controller abnormally ends or is cancelled, while the LPAR that the controller is running on remains active, jobs might be submitted again even though they are currently running, resulting in the same job being run twice.
  3. For dynamic and remote engine workstation types, this keyword supports only the value LEAVE. If you specify any other value, it is forced to LEAVE.
- The second keyword value defines reroute actions for workstation offline situations. Specify REROUTE to route operations, whose reroutable option is YES, to the alternate workstation. Specify LEAVE to leave operations to be scheduled on the original workstation; this is the default value. Rerouting does not occur for these operations.
- The third keyword value defines the action to be taken when a workstation becomes active again. Specify MANUAL to indicate that the status of the workstation should be changed manually when a *workstation available* indication is received. IBM Workload Scheduler for z/OS issues an MLOG message to inform the operator that the event has been received. Specify IMMED to automatically set the status of the workstation to available and withdraw any rerouting *immediately* when an event indicates that the workstation is operational; this is the default value.

| **ZCENJSUB (NO|YES)**

| Specify YES if IBM Workload Scheduler for z/OS should submit jobs running
| on z-centric, dynamic, and remote engine workstations. Specify NO if IBM
| Workload Scheduler for z/OS should not automatically submit jobs running on
| z-centric, dynamic, and remote engine workstations. This parameter is
| incompatible with JOBSUBMIT.

| The job-submit option can be changed through the Service Functions dialog
| while IBM Workload Scheduler for z/OS is running.

ZCHIGHRC (NO|DEF (HIGHEST NO-ERROR RC)|YES)

The ZCHIGHRC keyword defines the highest error code that can be generated in an IBM Workload Scheduler for z/OS job that runs on z-centric or Dynamic Workstations without causing IBM Workload Scheduler for z/OS to set the operation in error. Negative return codes are not supported, in the same way as for HIGHRC keyword in the JTOPTS.

YES Default value. All the jobs running on z-centric or dynamic workstations are handled the same way as z/OS jobs, that is IBM Workload Scheduler for z/OS applies the value of the highest in JTOPTS to these non-z/OS jobs unless a different value is specified at operation level.

NO The HIGHRC feature is deactivated for jobs running on z-centric and dynamic workstations. Any nonzero return code is handled as an error condition, regardless of the value specified at operation level in the HIGHRC.

DEF (HIGHEST NO-ERROR RC)
The value specified is used as global HIGHRC for all the jobs running on z-centric and dynamic workstations, unless a different value is specified at operation level.

Examples

```
JTOPTS BACKUP(1000)                            1
       ETT(YES)                                2
       HIGHRC(0)                               3
       JOBCHECK(SAME)                          4
       NOERROR(U001,ABC123.*.*.0016,*.P1.S1.U*) 5
       SHUTDOWNPOLICY(50)                      6
       STATIM(25)                              7
       STATMSG(CPLOCK,EVENTS,WSTASK)           8
       SUBFAILACTION(E)                        9
       SUPPRESSACTION(C)                      10
       SUPPRESSPOLICY(50)                     11
       TRACK(JOBOPT)                          12
       WSFAILURE(LEAVE,LEAVE,IMMED)           13
       WSOFFLINE(LEAVE,LEAVE,IMMED)           14
```

In this example of a JTOPTS statement:

1 Whenever the number of records in the current plan data set reaches 1000, the data set is copied to the job-tracking log.

2 The event-triggered-tracking function is initially active when IBM Workload Scheduler for z/OS starts.

3 If the operation completion code is greater than 0, the job or started task is treated as having ended in error.

4 IBM Workload Scheduler for z/OS submits jobs only if they have a valid job card where the job name matches the operation name.

5 Any job or started task with user abend code U001 is not considered to be in error. Operation ABC123 is not considered to be in error if it has error code 0016. Any user abend code for any job or started task in procedure step P1, step S1, is not considered an error.

6 An operation is started if the time remaining in the workstation-open interval is not less than 50% of the estimated duration of the operation.

7 The statistics message will be issued approximately every 25 minutes.

8 The event-manager subtask issues messages showing:
- How often different tasks have referenced the current plan data set
- Which tasks have updated the current plan
- How many events have been processed
- Statistics collected by the WSA task

9 IBM Workload Scheduler for z/OS sets an operation to ended-in-error if a failure occurs during the submission of a job or the starting of a started task.

10 If a time-dependent operation becomes late (after the SUPPRESSPOLICY buffer has expired) and the suppress-if-late option is set to Y for this operation, the operation will be set to complete.

11 In the example, a time-dependent operation with the suppress-if-late option is started if the time remaining until its deadline is not less than 50% of the estimated duration of the operation.

12 IBM Workload Scheduler for z/OS tracks jobs only if the mode of submission matches the JOB SUBMIT option in the application description.

13 If the status of a workstation is changed to FAILED, started operations at the workstation are left in the started status. Ready operations are not rerouted to an alternate workstation. They will be scheduled on the original workstation when it becomes active again. When the workstation becomes active again, it is immediately made available.

14 If the status of a workstation is changed to OFFLINE, started operations at the workstation are left in the started status. Ready operations are not rerouted to an alternate workstation. They will be scheduled on the original workstation when it becomes active again. When the workstation becomes active again, it is immediately made available.

MONOPTS

Purpose

The MONOPTS statement defines monitoring options needed for IBM Workload Scheduler for z/OS to work with IBM Tivoli Monitoring. This statement is used by a controller.

The MONOPTS statement defines the information needed to activate an IBM Tivoli Monitoring activity. If this statement is provided, the controller starts the monitoring task used to send events to the Tivoli Enterprise Portal client interface.

MONOPTS is defined in the member of the EQQPARM library as specified by the PARM parameter on the JCL EXEC statement. Defining this statement requires the definition of an OMVS segment for the controller user ID.

Format

```
►►─MONOPTS─┬─────────────────────────┬─MONHOSTNAME─(─┬─hostname───┬─)──────►
           │                ┌─NO──┐  │               └─IP address─┘
           └─BULKDISC─(─────┴─YES─┴─)┘
```

```
►─┬─────────────────────────────┬──────────────────────────────────────────►◄
  │               ┌─7500─┐      │
  └─MONPORT──(────┴─port─┴───)──┘
```

Parameters

BULKDISC YES|NO)
Describes the bulk discovery policy. Valid values are YES and NO, the default is NO.

NO IBM Workload Scheduler for z/OS does not perform an automatic discovery. A discovery is possible only manually using the TSO command.

YES A bulk discovery is performed each time a daily planning action is requested. A discovery can also be requested manually.

MONHOSTNAME(*hostname*|*IP address*)
Specifies the hostname or the IP address of the remote monitoring application. For the integration with IBM Tivoli Monitoring, this is the hostname of the system where the IBM Tivoli Monitoring agent is installed. This parameter is mandatory.

MONPORT(*value*|7500)
This parameter identifies the port number of the remote monitoring application. This is the port number used by the IBM Tivoli Monitoring agent. If not specified, the default value of 7500 will be used. The value specified here must correspond to the value defined on the Tivoli Monitoring agent.

Examples

```
MONOPTS  MONHOSTNAME('9.122.227.72')         1
         MONPORT(7500)                       2
         BULKDISC(YES)                       3
```

In this example of MONOPTS statement:

1 This is the IP address of the Tivoli Monitoring agent (remote monitoring application).

2 This is the port number of the Tivoli Monitoring agent.

3 A bulk discovery of the monitored resources will be initiated automatically at every planning action (extend, replan).

MONPOL

Purpose

The MONPOL statement defines the monitoring policy based on which IBM Workload Scheduler for z/OS operations are automatically monitored. The operations satisfying the specified policy, together with all operations manually flagged using EXTERNAL MONITOR=YES, constitute the current set of monitored operations. One or more values can be specified for the MONPOL statement.

This statement is only used in conjunction with the MONOPTS statement (used by IBM Tivoli Monitoring) or when EXTMON = YES has been specified in the OPCOPTS statement (used by Tivoli Business Systems Manager).

Format

```
                          ┌ERROR────┐
►►──MONPOL──OPERATION──(──┼CRITICAL─┼──)────────────────────────────────────────►◄
                          ├LATE─────┤
                          ├DURATION─┤
                          └MANUAL───┘
```

Parameters

(OPERATION CRITICAL||ERROR|LATE|DURATION|MANUAL)

Defines the criteria used to select operations to be monitored by IBM Tivoli Monitoring or by Tivoli Business Systems Manager

CRITICAL

The scheduler will monitor operations that are:

- Eligible for WLM assistance.
- Included in a critical network. In particular, when the scheduler recalculates a critical path or changes the risk level of an operation, it sends an event to the IBM Tivoli Monitoring agent.
- Added to the hot list.

ERROR

All operations that ended in error will be monitored. This is the default value.

LATE

All late operations will be monitored. An operation is considered late if it reaches its latest start time and is not started, complete, or deleted.

DURATION

All operations that run beyond their estimated duration time will be monitored.

MANUAL

Only those jobs explicitly flagged by the user as "monitored" will be selected. This value excludes all other values specified, with the exception of CRITICAL.

Note:

1. The MONPOL statement is not retroactive. Based on the specified policy, operations will be selected for monitoring only when their next change of status matching the MONPOL criteria occurs. If an operation is in one of the specified states before MONPOL is in effect, it will not be monitored.
2. When an operation has been promoted to "monitored" for a LATE or DURATION condition, it will continue to be monitored even if the EXTERNAL MONITOR flag for the operation is subsequently changed to NO.

Examples

```
MONPOL   OPERATION (CRITICAL        1
                    LATE
                    DURATION)
```

In this example of MONPOL statement:

1 Operations eligible for WLM assistance, included in a critical network, and added to the hot list will be automatically selected for monitoring.

NOERROR

Purpose

The NOERROR statement defines a list of error codes that, for job-tracking purposes, are treated as normal completion codes. This statement is used by a controller or standby controller. It can be specified more than once.

The NOERROR statement performs the same function as the NOERROR keyword on the JTOPTS statement. You can use NOERROR statements with, or instead of, the NOERROR keyword. By creating NOERROR statements, you can specify a greater number of error codes than is possible with only the NOERROR keyword of the JTOPTS statement. You might also find it useful to group error codes on different NOERROR statements.

NOERROR is defined in the member of the EQQPARM library as specified by the PARM parameter on the JCL EXEC statement.

To dynamically rebuild the NOERROR table, based on updated NOERROR statements in the main parameter library member (that is, the PARM value for EXEC PGM=EQQMAJOR), you can run either one of the following commands:

```
F ssnm, NEWNOERR
    or
F ssnm, NOERRMEM(member)
```

For example, to delete all NOERROR codes defined by member M1, change M1 to contain only comments and enter the command:

```
F ssnm, NOERRMEM(M1)
```

Note:

1. If you used either of the previous commands to dynamically update the NOERROR table, and you need to restart the controller with CURRPLAN(NEW) in the JTOPTS initialization statement, update the NOERROR definitions also in the EQQPARM library, before stopping and restarting the controller. If you do not do this, the scheduler does not keep the updated data when restarting.
2. If you use the command:
   ```
   F ssnm, NOERRMEM(member)
   ```
   the scheduler replaces any entry for the same member in the current NOERROR table, provided that the entry is correct and consistent with the previous ones.
3. If you want to delete an entry from the current NOERROR table, perform the following steps:
   a. Delete or comment out that entry in the same member you used to define it, for example M1 member.
   b. Use the command
      ```
      F ssnm, NOERRMEM(M1)
      ```
4. Do not use the asterisk (*) and percent sign (%) as part of the error code when it is not a numeric return code (for example, CAN, JCLI, and so on) and when relational operators, different from EQ or NE, are specified.

Format

```
                        ┌─,──────────────────┐
                        ▼                    │
►►──NOERROR──LIST──(────error code entry─────┴──)──────────────────────────►◄
```

Parameters

LIST(*error code entry***, ...,***error code entry***)**

Specify one or more error codes that, for job-tracking purposes, are treated as normal completion codes. Entries in the list are in two formats: a general format that applies to all jobs and started tasks; and a specific format that applies to a single job or started task, or to a set of jobs or started tasks.

A general entry can be:
- A 4-digit job or started-task return code (*nnnn*)
- A system abend code (S*xxx*)
- A user abend code (U*xxx*)
- An IBM Workload Scheduler for z/OS-defined code.

A specific entry has the following format:

jobname.stepname.procstepname.errorcode

jobname
: The name of a job or started task as specified in the job-name field of the operation. The maximum length is 8 characters.

stepname
: The name of a step that invokes an in-stream or cataloged procedure. This is always the name of an EXEC PROC statement. The maximum length is 8 characters.

This value is meaningless for an IBM Workload Scheduler for z/OS Agent workstation.

procstepname
: The name of a step that invokes a program. This is always the name of an EXEC PGM statement. The maximum length is 8 characters.

This value is meaningless for an IBM Workload Scheduler for z/OS Agent workstation.

errorcode
: The error code that is not to be considered an error. The error code can be a positive or negative number. For negative numbers, you can specify -*n* where *n* is a number starting with the minus (-) symbol with a maximum of 4 digits. For positive numbers, you can specify *n* where *n* is a number with a maximum of 4 digits; in this case the plus (+) symbol can be omitted.

operator
: The name of the relational operator to be used together with the specified error code. This field is optional and can be up to two characters in length. Valid values are:

EQ	Equal to (this is the default).
GE	Greater than or equal to.
GT	Greater than.
LE	Less than or equal to.

LT Less than.

NE Not equal to.

TO Indicates a range between two values. Use it by specifying an expression with the form

```
errorcode.TO.errorcode2
```

where the extreme values are inclusive.

The default is EQ for capability with earlier versions.

errorcode2
: The error code that is not to be considered as an error, as second boundary in a range expressed by the TO operator. errorcode corresponds to the first element of the boundary. It can be up four characters in length. It is required when the operator is TO. Do not use it when you specify another operator.

When you include an entry in the specific format, the first four names are required. That is, there must be at least three periods.

You can use the asterisk (*) and percent sign (%) as part of the names to create a generic name that can match many values.

An asterisk can represent a character string of unknown length. A single asterisk matches against any name, including a blank name. Two adjacent asterisks match the same values as a single asterisk. More than two adjacent asterisks match a specific error code range, but for this purpose you are recommended to use the operator TO. For example, to match the error codes from 0 to 999, from 1000 to 1999, from 2000 to 2999, and from 3000 to 3999, you specify either of the following:

```
NOERROR LIST(********.********.********.0*** ,
             ********.********.********.1*** ,
             ********.********.********.2*** ,
             ********.********.********.3*** )

      OR

NOERROR LIST(*.*.*0.TO.3999)
```

The second statement, using the operator TO, is preferred and does not require the processing of redundant multiple asterisks.

Use a percent sign to represent a single character. A single percent sign matches any character in a specific position, except blank.

The scheduler parsing process performs the following logical checks:

- Duplicate entries, for example
  ```
  (*.*.*.10.EQ, *.*.*.10.EQ)
  ```
- Overlapping and consistent entries, for example
  ```
  (*.*.*.10.TO.50, *.*.*.15.EQ)
  ```
- Inconsistent entries, for example:
  -
    ```
    (*.*.*.100.EQ, *.*.*.100.NE)
    ```
  -
    ```
    (J*.S*.P%S*.120.GE,*.*.*.115.GT)
    ```
  -

```
(*.*.*.0C4.GE ,*.*.*.0C6.GT)
```

As a general rule, when processing error events, the scheduler sequentially checks the NOERROR table. As soon as the process finds a NOERROR matching condition, it stops scanning the NOERROR table. Therefore avoid specifying ambiguous conditions, such as GE or GT comparisons in the previous examples: decide which value is to be used for the comparison and define a unique statement.

The check results are returned in specific messages written in the controller message log.

Note:

1. IBM Workload Scheduler for z/OS-defined error codes OSUB, OSUF, OSUP, OJCV, OSEQ, and JCLI are always treated as errors.
2. To use the NOERROR keyword for a specific job or started task steps, the event writer options, as specified in the EWTROPTS initialization statement, must be set as follows:
   - The STEPEVENTS keyword must specify either ALL or NZERO.
   - The RETCODE keyword must specify HIGHEST.
3. With PQ87904 APAR, error codes generated by the EQQCLEAN step, that is a step inserted into a restarted job by the Restart and Cleanup function, cannot be suppressed by the NOERROR logic.
4. When you set the **errorcode** name to an IBM Workload Scheduler for z/OS-defined code (for example JCLI), set the **operator** name to EQ.
5. The only acceptable error codes are those listed in Appendix E of *Managing the Workload*.
6. If JCC EID processing can be done for a job, you must not perform a NOERROR checking for the same job, even for a different EID return code.

Examples

```
NOERROR LIST(JOBSUBEX.*.*.S806)        1
NOERROR LIST(CAN)                      2
NOERROR LIST(*.*.*.U001)                   3
NOERROR LIST(TWSJOB2.*.*.0032.LE)          4
NOERROR LIST(TWSJOB 1.*.*.S806.TO.S810)    5
NOERROR LIST(*.*.*.10.TO.11,*.*.*.13.TO.15) 6
NOERROR LIST(*.*.*.0C6.GT)                 7
```

In this example of NOERROR statements:

1 Any job called JOBSUBEX that abends with system abend code S806 is treated as having ended normally.

2 Any job that is canceled by the operator or by a TSO user before execution is treated as having ended normally.

3 Any job that abnormally ends with user abend code U001, is treated as having ended normally.

4 Any job called TWSJOB2, that ends with an error code less than or equal to 0032, is treated as having ended normally.

5 Any job called TWSJOB1, that abnormally ends with a system code in the range S806 (inclusive) and S810 (inclusive), is treated as having ended normally.

6 Any job that ends with an error code in the range 10 (inclusive) and 15 (inclusive), excluding 12, is treated as having ended normally.

Note: You cannot manage such a case by combining the following conditions:

```
*.*.*.12.NE
```

and

```
*.*.*.10.TO.15
```

. For example, if the NOERROR table already contains

```
*.*.*.12.NE
```

and you are adding

```
*.*.*.10.TO.15
```

, the scheduler issues a warning message and does not add the second entry.

7 Any job that ends with an abend code greater than 0C6, is treated as having ended normally.

OPCOPTS

Purpose

The OPCOPTS statement defines run-time options to IBM Workload Scheduler for z/OS. This statement is used by a tracker, controller, standby controller, or backup controller.

OPCOPTS is defined in the member of the EQQPARM library as specified by the PARM parameter in the JCL EXEC card of the controller and tracker started-task procedures. If no PARM parameter is specified on the JCL EXEC card, the default member name STDPARMS is used.

Format

```
►►──OPCOPTS──┬──────────────────────────────┬─┬─────────────────────────┬──────────────►
             │              ┌─NO──┐         │ │          ┌─NO──┐        │
             └─APPCTASK──(──┴─YES─┴──)──────┘ └─ARM──(───┴─YES─┴──)─────┘

►─┬──────────────────────────────────┬─┬─────────────────────────────────────┬─────────►
  │            ┌─STDAR───────┐       │ │                      ┌─MLOG───┐     │
  └─ARPARM──(──┴─member name─┴──)────┘ └─AUTOMATIONMSG──(─────┼─SYSLOG─┼──)──┘
                                                              ├─BOTH───┤
                                                              └─NONE───┘

►─┬───────────────────────────────────┬─┬──────────────────────────────────────────┬───►
  └─BUILDSSX──(──┬─MERGE───┬──)───────┘ │               ┌─IBM-037──────────────┐   │
                 └─REBUILD─┘            └─CODEPAGE──(───┴─host system codepage─┴──)┘

►─┬───────────────────────────────────────────────────┬────────────────────────────────►
  │                       ┌─this subsystem─┐          │
  └─CONTROLLERTOKEN──(────┴─subsystem name─┴──)───────┘

►─┬───────────────────────────────────────────────────┬────────────────────────────────►
  │             ┌─40──────────────────┐               │
  └─CPBPLIM──(──┴─CP buffer pool size─┴──,──PS──)─────┘
```

```
►──┬─────────────────────────────────────────────────┬──────────────────────────────────────────────►
   │                ┌─50───────────────────────┐     │
   └─CPDTLIM──(──┴─CP data set size percentage─┴──)─┘

►──┬──────────────────────────────────────────┬──┬─────────────────────────────────┬───────────────►
   └─DB2SYSTEM──(────DB2 subsystem────)─┘  │               ┌─NO──┐                 │
                                                  └─E2EOSEQ──(──┴─YES─┴──)─┘

►──┬───────────────────────────────────────────┬──┬────────────────────────────────────────────────┬─►
   │                   ┌─STDERDR──────────┐     │  │              ┌─1───────────────────────┐     │
   │                   │ ┌─,──────────┐   │     │  └─ERDRTASK──(──┴─number of event readers─┴──)─┘
   │                   │ ▼            │   │     │
   └─ERDRPARM──(───┴──member name───┴───┴──)─┘

►──┬───────────────────────────────────────────┬──┬───────────────────────────────┬───────────────►
   │               ┌─0──────────────────┐      │  │            ┌─NO──┐            │
   └─EXIT01SZ──(──┴─number of JCL lines─┴──)─┘  └─EXTMON──(──┴─YES─┴──)─┘

►──┬───────────────────────────────────────┬──┬─────────────────────────────────┬───────────────►
   │               ┌─STDEWTR─────┐         │  │              ┌─NO──┐            │
   └─EWTRPARM──(──┴─member name─┴──)─┘  └─EWTRTASK──(──┴─YES─┴──)─┘

►──┬──────────────────────────────────┬──┬──────────────────────────────────────┬──────────────►
   │                ┌─YES─┐           │  │                ┌─0────────────┐      │
   └─GDGNONST──(──┴─NO──┴──)─┘  └─GMTOFFSET──(──┴─offset value─┴──)─┘

►──┬──────────────────────────────────────────────────┬──┬──────────────────────────────────┬──►
   │             ┌─5──────────────────────────────┐    │  │             ┌─GLOBAL───┐          │
   └─GSTASK──(──┴─number of parallel requests─┴──)─┘  └─GTABLE──(──┴─table ID─┴──)─┘

►──┬──────────────────────────────────────┬──┬───────────────────────────────┬──────────────────►
   │              ┌─STDJCC──────┐         │  │             ┌─NO──┐           │
   └─JCCPARM──(──┴─member name─┴──)─┘  └─JCCTASK──(──┴─YES─┴──)─┘

►──┬──────────────────────────────────────┬──┬──────────────────────────────────────────────────┬─►
   │              ┌─,───────────┐         │  │                     ┌─5000───────────┐           │
   └─JESPLEX──(──┴─System Name─┴──)─┘  └─MAXHISTORYROWS──(──┴─number of rows─┴──)─┘

►──┬──────────────────────────────────────────┬──┬─────────────────────────────────────────────────┬─►
   │                 ┌─0──────────────┐       │  │                    ┌─EQQMLOG────────┐          │
   └─MAXSUBJOBS──(──┴─number of jobs─┴──)─┘  └─MLOGPROCNAME──(──┴─procedure name─┴──)─┘

►──┬──────────────────────────────────────┬──┬───────────────────────────────┬──────────────────►
   └─NCFAPPL──(────VTAM LU name────)─┘  │             ┌─NO──┐           │
                                              └─NCFTASK──(──┴─YES─┴──)─┘

►──┬───────────────────────────────────────┬──┬─────────────────────────────────┬────────────────►
   │                     ┌─YES─┐           │  │             ┌─YES─────┐         │
   └─NOERRCONCHECK──(──┼─NO──┼──)─┘  └─OPCHOST──(──┼─NO──────┼──)─┘
                         └─MSG─┘                            ├─BACKUP──┤
                                                             ├─STANDBY─┤
                                                             └─PLEX────┘

►──┬────────────────────────────────────┬──┬──────────────────────────────────────┬─────────────►
   │                   ┌─NO──┐          │  │               ┌─STDOSLC─────┐         │
   └─OPERHISTORY──(──┴─YES─┴──)─┘  └─OSLCPARM──(──┴─member name─┴──)─┘

►──┬──────────────────────────────────────────┬──┬────────────────────────────────────┬─────────►
   └─OUTCOL──(────started task name────)─┘  │                  ┌─NO──┐           │
                                                  └─OUTPUTWTRID──(──┴─YES─┴──)─┘

►──┬──────────────────────────────────┬──┬──────────────────────────────┬───────────────────────►
   │                ┌─NO──┐           │  │              ┌─NO──┐          │
   └─RCLEANUP──(──┴─YES─┴──)─┘  └─RCLPASS──(──┴─YES─┴──)─┘

►──┬──────────────────────────────────────────────────────┬──┬────────────────────────────────┬──►
   │                   ┌─400──────────────────────────┐   │  │              ┌─NO──┐            │
   └─REAPPLYCOUNT──(──┴─number of reapplied events─┴──)─┘  └─RECOVERY──(──┴─YES─┴──)─┘

►──┬──────────────────────────────────────────┬──┬──────────────────────────────────────┬──────►
   │                         ┌─NO──┐          │  │               ┌─STDRODM─────┐        │
   └─REMJCLDIRECTIVES──(──┴─YES─┴──)─┘  └─RODMPARM──(──┴─member name─┴──)─┘
```

```
►─┬──────────────────────────────┬─┬──────────────────────────────┬──────────────────────►
  │               ┌─NO──┐        │ └─SAVARFAIL──(───&, %     ──)─┘
  └─RODMTASK──(───┴─YES─┴──)─────┘

►─┬─────────────────────────────────┬─┬──────────────────────────────┬───────────────────►
  │             ┌─NO────────┐       │ │                ┌─NO──┐       │
  └─SECHECK──(──┼─ALL───────┼──)────┘ └─SERESETJS──(───┴─YES─┴──)────┘
                └─OPERONLY──┘

►─┬──────────────────────────────────────────────┬─┬─────────────────────────┬───────────►
  │              ┌─,───────────────────────┐     │ │           ┌─NO──┐       │
  │              ▼                         │     │ └─SPIN──(───┴─YES─┴──)────┘
  └─SERVERS──(───────Started task name─────┴──)──┘

►─┬─────────────────────────────────────────────────────────┬─┬──────────────────────────────┬─►
  │                                      ┌─TEMPORARY─┐      │ │                  ┌─NO──┐     │
  └─SSCMNAME──(───module name───,────────┴─PERMANENT─┴──)───┘ └─SUPPRESSENF──(───┴─YES─┴──)──┘

►─┬───────────────────────────────────────────────┬─┬──────────────────────────────────┬──►
  │                    ┌─0─────────────────┐      │ │                ┌─1────────────┐  │
  └─SWITCHMLOGLIM──(───┴─number of records─┴──)───┘ └─SYSPLEXID──(───┴─Sysplex Id───┴─)┘

►─┬─────────────────────────────┬─┬───────────────────────────────┬──────────────────────►
  │             ┌─YES─┐         │ └─TPLGYSRV──(───server name───)─┘
  └─TASKUSR──(──┴─NO──┴──)──────┘

►─┬─────────────────────────────┬─┬────────────────────────────┬─┬──────────────────────────┬─►
  │            ┌─SCAN─┐         │ └─VARFAIL──(───& % ?     ──)─┘ │              ┌─NO──┐     │
  └─VARSUB──(──┼─YES──┼──)──────┘                                └─VARPROC──(───┴─YES─┴──)──┘
               └─NO───┘

►─┬──────────────────────────────────────┬───────────────────────────────────────────────►
  │             ┌─0────────────┐         │
  └─WAITREL──(──┴─waiting time─┴──)──────┘

►─┬──────────────────────────────────────────────────────────────────────────────────────┬──►◄
  │                                                                 ┌─20──────┐         │
  │         ┌─WLMJOBPR───┐                       ┌─CONDITIONAL──(───┴─percent─┴──)─┐     │
  └─WLM──(──┴─class name─┴──,──policy name───────┼─────────────────────────────────┼──)──┘
                                                 ├─DURATION────────────────────────┤
                                                 ├─DEADLINE────────────────────────┤
                                                 └─LATESTSTARTTIME─────────────────┘
```

Parameters

APPCTASK(YES|NO)
: Specify YES if an AS400 tracker is used or there are programs using the API interface. The appctask is **not** required for the IBM Workload Scheduler for z/OS Server communication. You can specify APPCTASK for a tracker and a controller.

ARPARM(*member name*|STDAR)
: Defines the name of the member in the EQQPARM data set that contains an AROPTS statement for the automatic job-recovery task.

ARM(YES|NO)
: The z/OS Automatic Restart Manager (ARM) can reduce the impact of an unexpected error to IBM Workload Scheduler for z/OS because z/OS can restart it automatically, without operator intervention.

Specify YES if automatic restart of the failed IBM Workload Scheduler for z/OS component should be activated, and define the component in the ARM policy. The element name comprises the string "OPC" , the SYSTEMID, and the SUBSYSTEM name. ARM recovery of the failed IBM Workload Scheduler for z/OS component is possible in the same image (restart-in-place). This feature allows the recovery of the tracker and a fast restart of the controller and the server. In addition, restart-in-place does not reduce the number of standby controllers when there is a controller failure. You can customize the number of restarts and the period of a restart by setting the parameters for each IBM Workload Scheduler for z/OS component in the z/OS ARM policy.

AUTOMATIONMSG(MLOG|SYSLOG|BOTH|NONE)
Defines where the messages issued by System Automation are to be logged. You can specify one of the following:

MLOG
IBM Workload Scheduler for z/OS message log, this is the default.

SYSLOG
System log.

BOTH IBM Workload Scheduler for z/OS message log and system log.

NONE
None of the messages issued by the System Automation are logged.

Note that message EQQE123I is logged into MLOG, regardless of the value you set for this keyword. This option is valid only if you run System Automation version 3.1 (with the appropriate maintenance level installed), or later.

BUILDSSX(MERGE|REBUILD)
Defines if the subsystem communication vector table (CVT) extension for IBM Workload Scheduler for z/OS, the SSX, should be rebuilt at a new level when the address space is started. The SSX is created at subsystem initialization by the EQQINITM module. If the EQQINITM module has since been updated, by maintenance or because you are installing a new release or modification level of IBM Workload Scheduler for z/OS, use the BUILDSSX keyword to avoid a z/OS IPL.

Specify MERGE when operational data, such as the event-writer queue, should be copied to the new SSX. This ensures that the new event-writer queue is primed with events queued to the old SSX block. Use this option when starting an IBM Workload Scheduler for z/OS address space after installing maintenance updates.

Specify REBUILD when you are migrating to, or falling back from, a new release or modification level of IBM Workload Scheduler for z/OS. The event-writer queue from the old SSX is not referenced in the new SSX. Ensure you also identify the new subsystem communication module name using the SSCMNAME keyword.

Note:
1. The PTF coverletter `++HOLD` section identifies the service updates that require the SSX be rebuilt.
2. MERGE **cannot** be used when the old and new SSX blocks are built for different FMIDs. Do not use MERGE when migrating to, or falling back from, a new release or modification level of IBM Workload Scheduler for z/OS.
3. If you specify BUILDSSX(REBUILD) to migrate to, or fallback from, a new release or modification level of IBM Workload Scheduler for z/OS, ensure you also specify the SSCMNAME keyword.
4. The BUILDSSX parameter should be removed after the next IPL of the z/OS system as it is no longer required.

CODEPAGE(*host system codepage*|IBM – 037)
The name of the host code page. This value is used by the monitoring task to

convert the monitoring data to be sent to the monitoring agent. You can provide the IBM–*nnn* value, where *nnn* is the EBCDIC code page. The default value, IBM–037, defines the EBCDIC code page for US English, Portuguese, and Canadian French. The following is a list of the EBCDIC code pages:

IBM–939
Japan Extended

IBM–937
Taiwan

IBM–935
China

IBM–933
Korea

IBM–975
Greece

IBM–971
Iceland

IBM–970
Latin 2

IBM–838
Thai

IBM–500
International

IBM–424
Israel

IBM–297
France

IBM–285
UK

IBM–284
Spain - Latin America

IBM–280
Italy

IBM–278
Sweden - Finland

IBM–277
Denmark - Norway

IBM–274
Belgium

IBM–273
Germany

IBM–1388
China

IBM–1122
Estonia

IBM–1112
Baltic

IBM–1047
Open Systems

IBM–1026
Latin 5 (Turkey)

IBM–1025
Cyrillic

The following is a list of the EBCDIC code pages for EURO support:

IBM–1140
Finland, Sweden

IBM–1141
: Austria, Germany

IBM–1142
: Denmark, Norway

IBM–1143
: USA

IBM–1144
: Italy

IBM–1145
: Spain, spanish-speaking Latin America

IBM–1146
: UK

IBM–1147
: France

IBM–1148
: Belgium, Switzerland

IBM–1149
: Iceland

CONTROLLERTOKEN(*subsystem name***|this subsystem)**

This keyword is used for the history function. The subsystem name is a key for the history information. When the current plan is extended, the BATCHOPT CONTROLLERTOKEN value is used to write the information. When the information is retrieved from the database, the OPCOPTS CONTROLLERTOKEN value is used, so you can rerun jobs initiated from another subsystem current plan (for example, when a standby system takes over) by specifying the correct token after takeover.

If you specify OPERHISTORY(NO) or omit the OPERHISTORY keyword, the CONTROLLERTOKEN keyword is ignored.

CPBPLIM (*CP buffer pool size* **|40, PS)**

The CP buffer pool size is set to half the size of the current plan, but it is limited to a certain percentage of the amount of storage available; this percentage is given by the CPBPLIM parameter.

| When the current plan is closed (for example, when you make a backup or a new CP is generated) the buffer is deallocated and allocated again according to the percentage determined by CPBPLIM. This might result in a performance degradation. To avoid this problem, define the CPBPLIM size as a Permanent Storage by specifying the PS keyword, for example CPBPLIM(40,PS). In this way, you reserve a percentage of the available storage to be used for the buffers until the controller is recycled. The best value for the buffer pool size depends on your environment.

Values allowed are 0-99. Values 1-99 are considered as a percentage; value 0 causes the old restriction, 400 blocks. The default is a percentage of 40.

CPDTLIM (*CP data set size percentage***|50)**

This keyword determines the percentage of size of the current plan used as maximum for buffer pool size. If CPDTLIM is omitted, the default 50 is used.

Values allowed are 0-100. Values 1-100 are considered as a percentage; value 0 causes the default of 50 to be used. Recommended values are between 50 and 100.

DB2SYSTEM(*DB2 subsystem***)**

This keyword is required for the history function. It specifies the DB2 subsystem, as in the IEFSSNxx member of SYS1.PARMLIB.

If you specify OPERHISTORY(YES) but omit the DB2SYSTEM keyword, an error message is issued.

ERDRPARM(*member name***|STDERDR)**
Defines the names of the members in the EQQPARM data set that contain an ERDROPTS statement for an event-reader task. The number of member names in this list must equal the number of event-reader tasks as defined by the ERDRTASK keyword. This keyword is required if more than one event reader is started.

ERDRTASK(*number of event readers***|1)**
Defines the number of event-reader tasks to be started by IBM Workload Scheduler for z/OS. The maximum number is 16.

Note:

1. If no event-reader tasks should be started, you must specify ERDRTASK(0).
2. On controller side, separate event-reader tasks must be started when there are trackers connected via shared DASD.
3. On tracker side, it is recommended (for performance reasons) not to start a separate event-reader task by setting ERDRTASK(0). The event reader function should be activated in an event-writer task by specifying EWSEQNO on the EWTROPTS statement.

EWTRPARM(*member name***|STDEWTR)**
Defines the name of the member in the EQQPARM data set that contains an EWTROPTS statement for the event-writer task.

EWTRTASK(YES|NO)
Specify YES if the IBM Workload Scheduler for z/OS subsystem should start an event-writer task to create events in an event data set.

GDGNONST(YES|NO)
If you set this keyword to No, IBM Workload Scheduler for z/OS tests the JCFGDG section to recognize if a data set is a GDG member. This means that the data set triggering function does recognize as GDG the data set referenced in the JCL in expanded format.

Note: The restart and cleanup function might not work properly wih GDG.
If you set this keyword to Yes, IBM Workload Scheduler for z/OS assumes that a data set having its last qualifier syntax compatible with a GDG member, is a GDG member.

| **E2EOSEQ(YES|NO)**
| In end-to-end scheduling with fault tolerance capabilities, determines if the event related to a manual status change is sent. If you set it to NO (default), the events are not sent; this prevents an error code OSEQ from being issued when you manually modify an operation within an end-to-end with fault tolerance environment, after a Symphony renew was already run.

| If you set it to YES, the events related to a manual status change are always sent. This might result in additional AWSBHT019E messages issued, for example when the event about an operation status change is sent and that operation in the Symphony file is already in that status; the status change event is then discarded.

EXIT01SZ(*number of JCL lines***|0)**
This keyword, if specified with a nonzero value, tells IBM Workload Scheduler for z/OS that the JCL size can be changes using user exit 01, and that the new JCL size cannot be greater than the value specified.

For example, a value of 5000 sets the limit to 5000 JCL lines. Each JCL line is considered to be 80 characters in length.

EXTMON (YES|NO)
Specify YES to enable IBM Workload Scheduler for z/OS to communicate with Tivoli Business Service Manager. Tivoli Business Service Manager is notified of any alerts or changes in status related to monitored jobs. Specify NO if no monitoring is wanted.

GMTOFFSET(*offset value*|0)
This keyword is used to request IBM Workload Scheduler for z/OS to correct the GMT clock which is recognized as incorrect. The keyword parameter value defines how much the clock must be corrected. The value must be a number, positive or negative, less than 1440 (24 hours), that defines the number of minutes that must be added to the GMT clock value, on this system (where the keyword is used), to get the true GMT time.

The default value is zero (GMTOFFSET(0), that is, the GMT clock shows the true GMT time.

Note: This parameter only enables IBM Workload Scheduler for z/OS to bypass GMT synchronization processing, and does not enable the correct interpretation of the timestamp in events received from the tracker when the connection is established. So the actual start time and end time of related operations are deviated by the amount of the specified offset.

GSTASK(*number of parallel requests*|5)
Defines the number of dialog requests that can be handled simultaneously by the general-service subtask to a maximum of 5. So GSTASK(1) does not permit parallel processing of requests, but GSTASK(5), the default value, allows maximum parallel processing.

GTABLE(*table ID*|GLOBAL)
Defines the name of the global JCL-variable table for the IBM Workload Scheduler for z/OS complex. This table contains JCL variable definitions that can be used for any operation within the IBM Workload Scheduler for z/OS complex. The global variable table is searched when a table (or variable within a table) that is referenced by an operation cannot be found.

If you schedule end-to-end with fault tolerance capabilities, set this keyword to the same value as the value of the GTABLE keyword in the BATCHOPT statement

You can specify only one table ID for the IBM Workload Scheduler for z/OS complex. IBM Workload Scheduler for z/OS uses the default name GLOBAL if you do not specify a table ID.

JCCPARM(*member name*|STDJCC)
Defines the name of the member in the EQQPARM data set that contains a JCCOPTS statement for the job-completion-checker task.

JCCTASK(YES|NO|)

Specify JCCTASK (YES) if the job-completion checker function is to be used.

Specify JCCTASK (NO) if the job-completion checker function is not to be used.

JESPLEX(*list of system names*)
Provides a list of the system names within the JESplex where the tracker belongs. Each system name can be up to 8 characters.

MAXHISTORYROWS(*number of rows***|**<u>**5000**</u>

This keyword, if specified with a nonzero value, tells IBM Workload Scheduler for z/OS the maximum number of rows that can be retrieved from DB2 by using the HIST command. The default value is 5000. the maximum value is 999999. If you specify 0, all the rows are retrieved without limitations.

For example, by specifying 1000 you set 1000 as the maximum number of rows that can be retrieved by the HIST command processing. If less than 1000 rows are retrieved, the rows are shown in the requesting panel. If more that 1000 rows are retrieved, the HIST command processing is interrupted, an error message is issued, and no data is shown in the requesting panel. To reduce the amount of data to retrieve, set the appropriate filtering criteria.

MAXSUBJOBS(*number of jobs***|**<u>**0**</u>

This keyword specifies the maximum number of jobs that IBM Workload Scheduler for z/OS can submit consecutively without verifying the processes related to time-dependent events (for example, WAIT workstation delay, critical path, and long duration alert). The default value is 0. The maximum value is 9999. If you leave 0, IBM Workload Scheduler for z/OS does not interrupt job submission to process time-dependent events.

This value can be modified also with the following operator command:

```
/F procname, MAXSUB=nnnn
```

where *procname* is the name of the IBM Workload Scheduler for z/OS JCL procedure and *nnnn* is the maximum number of jobs.

MLOGPROCNAME(*procedure name***|**<u>**EQQSMLOG**</u>**)**

The name of the procedure that the controller must run to copy the inactive MLOG file into the GDG data set (previously defined by the EQQPCS12 sample) following an MLOG switching routine.

EQQSMLOG is the default procedure name. It is available in the EQQSMLOG installation sample that you can install with the EQQJOBS installation aid, customize, and add in the `user.proclib` to make it accessible by the controller.

See also `Using two message log (MLOG) data sets` in *IBM Workload Scheduler for z/OS: Planning and Installation*.

NCFAPPL(*VTAM LU name***)**

Specify the VTAM LU name associated with the subsystem. This keyword is required if NCSK(YES) is specified.

NCFTASK(YES|<u>**NO**</u>**)**

Specify YES if the subsystem should start a network communication function (NCF) task. That is, communication is through VTAM.

NOERRCONCHECK(<u>**YES**</u>**|NO|MSG)**

Defines which level of check the scheduler performs when adding entries to the NOERROR table. Specify one of the following:

YES A consistency check is active. The scheduler does not add inconsistent entries to the NOERROR table and issues EQQN069W messages in the controller message log to indicate the inconsistencies. It is the default setting.

NO To force the scheduler to update the NOERROR table even after detecting inconsistent entries. Related EQQN069W messages are not issued. EQQN098I message (informing that the table has been updated anyway, based on NOERRCONCHECK setting) is not issued either. Setting this keyword to NO prevents any consistency check from being performed.

MSG Like the NO setting, it forces the scheduler to update the NOERROR table even after detecting inconsistent entries. Unlike the NO setting, related EQQN069W messages are issued. EQQN098I message is issued informing that the table has been updated anyway, based on NOERRCONCHECK setting.

`OPCHOST(YES|NO|BACKUP|STANDBY|PLEX)`

Defines the role of the subsystem. Specify one of the following values:

YES This subsystem has to start as the controlling system. It supports dialog users and contains the databases and plans.

NO This subsystem is not the controlling system.

BACKUP
This subsystem has to start as the backup controlling system.

STANDBY
This subsystem has to be prepared to change from running in non-host mode to host mode. OPCHOST(STANDBY) is valid only in an XCF environment. You cannot specify OPCHOST(STANDBY) and EWTRTASK(YES) on the same OPCOPTS statement.

PLEX This subsystem has to start as the controlling system. If there is already an active controller in the XCF group, the start continues as standby controller. OPCHOST(PLEX) is valid only when XCF GROUP and MEMBER have been specified.

If IBM Workload Scheduler for z/OS is started with OPCHOST(YES), NMM initializes the CKPT data set with FMID and level corresponding to SSX.

`OPERHISTORY(YES|NO)`

This keyword is required for the history function. It specifies that IBM Workload Scheduler for z/OS stores completed operation information in the DB2 database. If you specify YES, you must also specify the DB2SYSTEM keyword.

If you specify YES but omit the DB2SYSTEM keyword, an error message is issued.

If you specify NO or omit the OPERHISTORY keyword, but specify related keywords (such as DB2SYSTEM or CONTROLLERTOKEN) a warning message is issued to inform you that the history function is not active.

`OSLCPARM(`*`member name`*`|STDOSLC)`

Defines the name of the member in the EQQPARM data set that contains the OSLCOPTS statement for the OSLC integration.

`OUTCOL(`*`started task name`*`)`

Specify the name of the output collector started task to be started and stopped by this controller. Use this parameter in a sysplex environment to enable a standby controller to automatically take over the communication with Output collector for the automatic retrieval of job logs in a z-centric environment.

| **`OUTPUTWTRID(YES|NO)`**

| Default is NO. Specify YES to set, for each z-centric job output, its job name as
| the writer ID (SDSF WTR or IOF WTRID) when the job output is written to the
| specified JES SYSOUT class.

RCLEANUP (YES|NO)
Default is NO. Specify YES to use the restart and cleanup function. The value of RCLEANUP must match the value in the BATCHOPT for RCLEANUP. If RCLEANUP(YES) is coded, the CP16 records created by the batch job (if the BATCHOPT has RCLEANUP(YES) specified) are deleted as part of "catch up" processing. The FL task and the PSU task are started, the local data store is activated, and the scheduler changes the jobs to duplicate the output to OPC DEST ID.

When YES is specified, OUTPUTNODE(FINAL) is forced in the JTOPTS statement.

REAPPLYCOUNT(*number of re-applied events***|400**
This keyword specifies after how many re-applied events message `EQQN059I JT APPLY IS IN PROGRESS. NN RECORDS HAVE BEEN PROCESSED.` is issued. Valid values are from 1 to 32000. The default is 400.

RCLPASS (YES|NO)
Default is NO. Specify YES to use the restart and cleanup function for JCLs with DISP=(,PASS) in the last step.

RECOVERY(YES|NO)
Specify YES if the controller should start an automatic-job-recovery task to manage automatically operations that end in error.

Note: If you specify RECOVERY(NO), you cannot use the Service Functions dialog to activate the automatic-job-recovery task.

REMJCLDIRECTIVES(YES|NO)
Defines if JCL directives are to be removed from the JCL at submission time.

Specify YES to remove the directives from the JCL at submission time. In this case, the scheduler removes the directives before submitting the JCL, therefore the job output does not contain the directives.

Specify NO to keep the directives in the JCL at submission time. In this case the job output contains the directives.

RODMPARM(*member name***|STDRODM)**
Specifies the member in the EQQPARM data set that contains RODMOPTS statements for the RODM (resource object data manager) subtask. RODMPARM is not used by a tracker.

RODMTASK(YES|NO)
IBM Workload Scheduler for z/OS support for RODM (resource object data manager) lets you associate fields of special resources in the current plan with fields of RODM classes or RODM objects. Support for RODM requires that:

- A tracker is started on the same z/OS image as the RODM subsystem that requests are sent to, and RODMTASK(YES) is specified for both the tracker and the controller.
- An event writer is started in the IBM Workload Scheduler for z/OS address space that communicates with RODM. This address space creates resource events (type S) from RODM notifications, which IBM Workload Scheduler for z/OS uses to update the current plan.
- The controller is connected to the tracker through XCF, NCF, TCP/IPor a submit/release data set.
- Each address space has a unique RACF user ID if more than 1 IBM Workload Scheduler for z/OS address space communicates with a RODM

subsystem, such as when you start production and test systems that subscribe to the same RODM subsystem.

IBM Workload Scheduler for z/OS can communicate with several RODM subsystems.

SAVARFAIL(&, %)

This keyword tells IBM Workload Scheduler for z/OS not to consider unresolved variables in the System Automation command as an error, preventing the operation from completing with error code OJCV. The allowed variable types are the ampersand (&) and percent sign (%). You can use one or both of them, in any order, to bypass the substitution failure. For example if you set SAVARFAIL(&), IBM Workload Scheduler for z/OS does not consider as error the failing substitution of variables beginning with &.

Any combination of the two variables types is allowed, for example SAVARFAIL(&, %) or SAVARFAIL(%, &), but you must specify at least one type. If SAVARFAIL is not set, the failing substitution of a variable is considered as an operation error.

SECHECK(ALL|OPERONLY|NO)

This keyword applies to trackers and controllers where the Submit function is active (a blank destination is used to submit jobs locally). To activate WLM scheduling environments, you must specify a value other than NO.

When you activate this function, you must do so for all the trackers and for the controller within the same sysplex. This is needed to correctly implement the ENF listener mechanism, because each tracker activates the listener exits only in the MVS system where it is located.

The possible values are:

ALL All the submitted JCLs are to be checked for an associated scheduling environment name. The check is the following:

1. If it is defined, the scheduling environment name associated to the operation is used. In this case, the JCL is tailored by adding or replacing the SCHENV keyword of the JOB card statement. Any pre-existing value in the JCL is overwritten.
2. If the above condition is not true, the scheduling environment name specified in the SCHENV keyword of the JOB card statement is used, if it is present.
3. If the above condition is not true, no scheduling environment name is used (WLM is not queried).

When a scheduling environment is found, WLM is queried to find if the environment is available. If it is, the job is submitted, otherwise:

- If the environment is not available, the operation is set to READY, WAITING FOR SE
- If the environment is not defined, the operation is set to error (SEUN error code)

OPERONLY

Only the JCLs of operations with a scheduling environment defined in the IBM Workload Scheduler for z/OS database are to be checked. The check is the following:

- The scheduling environment name associated to the operation is used to insert or update in the JCL the SCHENV keyword of the JOB card statement. When this happens, the existing value in the JCL is overwritten.

- WLM is queried to find if the environment is available. If it is, the job is submitted, otherwise:
  - If the environment is not available, the operation is set to READY, WAITING FOR SE
  - If the environment is not defined, the operation is set to error (SEUN error code)

NO No checks are performed. This is the default setting.

Note: When the JOB card is tailored to insert or replace the SCHENV=new value keyword, any comments on the right might be ignored and could show truncated in the submitted JCL.

SERESETJS(YES|NO)
Specifies if the JCL must be replaced in the JS file, when the WLM scheduling environment becomes available; in this way, any variables contained in the JCL are updated with the most up-to-date values.

The possible values are:

YES The JCL is replaced in the JS file when the WLM scheduling environment becomes available.

NO (Default). The JCL is not replaced in the JS file when the WLM scheduling environment becomes available.

SERVERS(*Started task name***,** *Started task name***,...)**
Identifies one or more servers to be started, and stopped, by this controller. Use this parameter in a sysplex environment to enable a standby controller to automatically take over the communication.

SPIN(YES|NO)
This keyword enables or disables the usage of the JESLOG SPIN functionality available with z/OS 1.2 or later. If you specify YES, the usage of the JESLOG SPIN functionality is enabled. Because the Restart and Cleanup or Job Completion Checker functions do not support this functionality, when you use them a warning message is issued.

If you specify NO, the usage of the JESLOG SPIN functionality is disabled by adding JESLOG=NOSPIN to the JOB card of every submitted job. SPIN(NO) is effective only if RCLEANUP(YES) or JCCTASK(YES) is specified. If you use the Restart and Cleanup function, a JOB card is added automatically to each submitted JCL when missing. If you do not use the Restart and Cleanup function (RCLEANUP(NO)), a warning message is issued for each submitted JCL that does not have a job card. The default is NO.

Note: When the JOB card is tailored to insert or replace the JESLOG=NOSPIN keyword, any comments on the right might be ignored and could show truncated in the submitted JCL.

SSCMNAME(*module name***,PERMANENT|TEMPORARY)**
The first keyword value defines the name of the subsystem communication module to be used instead of EQQSSCMM that was loaded at IPL. The second keyword value specifies if the module should replace the one loaded at IPL. Use this keyword to load an updated version of the module before a scheduled IPL. The module you specify must reside in an APF-authorized library defined by either the STEPLIB ddname or LNKLST*nn* concatenation. If SSCMNAME is not specified or specifies a module that cannot be located in an authorized library, IBM Workload Scheduler for z/OS events continue to be generated by the EQQSSCMM loaded at IPL.

Specify PERMANENT as the second keyword value to replace the subsystem communication module loaded at IPL with the module identified in the first keyword value. In this case the module specified must reside in an APF-authorized library defined by the STEPLIB ddname.

When TEMPORARY is specified or defaulted as the second keyword value, the module you specify generates IBM Workload Scheduler for z/OS job-tracking events only while the IBM Workload Scheduler for z/OS address space is active. When the address space is stopped, events continue to be generated by the EQQSSCMM module loaded at IPL.

Note:

1. The PTF cover letter `++HOLD` section identifies service updates that require a new subsystem communication module to be loaded.
2. Ensure you specify this keyword when BUILDSSX(REBUILD) option is used to migrate to, or fallback from, a new release or modification level of IBM Workload Scheduler for z/OS.
3. The SSCMNAME keyword should be removed after the next IPL as it is no longer required.

SUPPRESSENF(YES|NO)
Specifies if activation of the ENF 57 and ENF 41 listener exits must be cancelled.

YES Exits ENF 57 and ENF 41 are not to be activated although it is marked for activation (because a value other than NO was specified for SECHECK).

NO Activation of the exits is not to be cancelled. This is the default.

You should not set this parameter to YES on trackers located in a sysplex that is not the one where the controller is. You can cancel activation of the exits on trackers that belong to the same sysplex as the controller only if there is a tracker in the same MVS image as the controller and the ENF exits are active (SECHECK is not set to NO).

SWITCHMLOGLIM(*number of records*|0)
Specifies how many records must be in the active message log (MLOG) before the MLOG switching function is started. The MLOG switching function stops the running MLOG file, copies it into a GDG data set, and starts the alternate MLOG file. The alternate MLOG file records upcoming messages until the SWITCHMLOGLIM value is reached again, and the process is repeated.

The default value is 0. It means that the MLOG switching function is not to be run. If the value is greater than 0, the function is activated; however, if no DD EQQMLOG2 is defined in the started task JCL, or if the name is already locked, the function cannot be activated and the following message is issued:

```
EQQZ408W SWITCHMLOGLIM PARAMETER IS SPECIFIED BUT EQQMLOG2 DATA SET IS NOT AVAILABLE.
         MLOG SWITCH FUNCTION IS NOT ACTIVATED.
```

The maximum possible value is 999999999 records.

See also `Using two message log (MLOG) data sets` in *IBM Workload Scheduler for z/OS: Planning and Installation*.

SYSPLEXID (nnn)
Identifies the sysplex where the controller or tracker is located. nn is an integer ranging from 1 to 9999. The default is 1.

TASKUSR(NO|YES)
Specifies if a started task is to be run with the user ID associated with the task, instead of the user ID associated with the job name.

YES The task is run with the user ID associated with the started task name. This is the default.

NO The task is run with the user ID associated with the job name.

TPLGYSRV (*server_name*)
Activates the end-to-end scheduling with fault tolerance capabilities feature in the controller. If you specify this keyword the IBM Workload Scheduler Enabler task is started. Defining this parameter requires the definition of an OMVS segment for the controller user ID.

The specified ***server_name*** is that of the server that handles the events to and from the distributed agents. Only one server can handle events to and from the distributed agents.

VARSUB(YES|NO|SCAN)
This keyword tells IBM Workload Scheduler for z/OS whether JCL variable substitution should be performed. YES means that variable scanning is performed for all computer operations. NO means that variable scanning does not occur. SCAN tells IBM Workload Scheduler for z/OS to search JCL for variables only if the //*%OPC SCAN directive has been found in the JCL.

This keyword applies also to variable substitution in the System Automation command text. YES and SCAN mean that the command text is scanned for variable substitution. NO means that variable scanning is not performed.

VARFAIL(&, %, ?)
This keyword tells IBM Workload Scheduler for z/OS whether or not unresolved variables in the JCL would cause a JCL error (OJCV). You can use from one to three of the following characters, in any order, to bypass substitution failure (&, %, ?).

For example, if VARFAIL(&) is specified IBM Workload Scheduler for z/OS does not consider the failure of a substitution of variables beginning with an & to be an error. Any combination of the three types is allowed, for example VARFAIL(&, %) or VARFAIL(?), but at least one value must be specified while any repetition of characters is rejected.

If VARFAIL is not specified, all the lack of substitution of variables is treated as errors, as previously.

Note that the error bypass is not applicable to IBM Workload Scheduler for z/OS Directive and Dependent variables.

VARPROC(YES|NO)
This keyword tells IBM Workload Scheduler for z/OS whether online procedures are to consider variable substitution. If VARPROC(YES) is specified, variables in online procedures are resolved.

The default is NO.

WAITREL(*waiting time*|0)
Defines, in tenth of seconds, the time that IBM Workload Scheduler for z/OS waits before releasing a held job. Valid values are from 0 to 99; the default value is 0.

You can dynamically update the WAITREL value by issuing the modify command `/F subsys,WAITREL=`*nn*.

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

The WLM keyword defines the options that are to be applied for workload manager intervention. The WLM function can be used to assign extra resource to critical jobs that are running late. IBM Workload Scheduler for z/OS does this by promoting a late critical job to a higher performance WLM service class. WLM is defined in the member of the EQQPARM library as specified by the PARM parameter on the JCL EXEC statement.

Class Name

The name of the WLM service class to which late critical jobs are promoted. It can be an existing service class or a new service class created for this purpose.

Policy Name

The means by which IBM Workload Scheduler for z/OS decides whether a critical job is running late. Specify one of the possible choices, and that is the default policy that IBM Workload Scheduler for z/OS applies. This default can be overwritten at the individual job level. The following list shows the valid policy names:

DURATION

When a critical job exceeds the expected duration (as specified in the current plan), IBM Workload Scheduler for z/OS moves it to the higher-performance service class, according also to the ALEACTION or LIMFDBK value set in the JTOPTS statement. (For details about the ALEACTION and LIMFDBK keywords, refer to "JTOPTS" on page 77.)

Note: The fact that a job exceeds its expected duration does not necessarily imply that it is running late or that it will negatively impact the plan as a whole. Specifying this policy could mean that extra resource is dedicated to a job when not strictly necessary.

DEADLINE

When a critical job runs beyond its deadline time (calculated as Latest Start Time + Duration), IBM Workload Scheduler for z/OS moves it to a higher-performance service class.

Note: If this policy is used, then extra resources are dedicated to a job only when it really needs it. The extra resources help to reduce the delay.

LATESTSTARTTIME

When a critical job starts after its latest start time, IBM Workload Scheduler for z/OS promotes it to a higher-performance service class.

Note: Using this policy the gains are greater, as extra resources are dedicated to the job for its entire execution time. However, with this policy, it is possible to over-compensate, dedicating extra resources to a job for longer than necessary. For example, a long running job might start only a few minutes late, but get enough extra resources that let it finish early.

CONDITIONAL (NN)

This policy uses a simple test that decides whether the deadline or the lateststarttime policy is applied when a critical job starts after its latest start time.

The numeric value (NN) specified in this policy defines a threshold, which is used as follows:

```
If(Delay÷Time to Deadline)×100 > NN
```

then IBM Workload Scheduler for z/OS moves the job immediately to the higher-performance service class. This effectively applies the lateststarttime policy.

```
If (Delay÷Time to Deadline)×100 <= NN
```

then IBM Workload Scheduler for z/OS takes no action until the job runs beyond its deadline time. This is to avoid over-compensation.

Note: If you use the default member name for ARPARM, ERDRPARM, EWTRPARM, JCCPARM, or RODMPARM, you must define the member.

Examples

```
OPCOPTS OPCHOST(YES)                     1
        NCFTASK(YES)                     2
        NCFAPPL(NCFAPPL1)                3
        ERDRTASK(2)                      4
        ERDRPARM(ERDR1,ERDR2)            5
        GMTOFFSET(60)                    6
        GTABLE(GVARTAB)                  7
        SERVERS(OPCBCOM1,OPCBCOM2)...    8
```

In this example of an OPCOPTS statement:

1 IBM Workload Scheduler for z/OS is started as a controller. This system supports dialog users.

2 The network communication function (NCF) is started.

3 The controller has the VTAM LU name NCFAPPL1.

4 Two event-reader tasks are started.

5 Run-time options for the event-reader tasks are specified in the ERDR1 and ERDR2 members of the parameter library.

6 IBM Workload Scheduler for z/OS adds one hour to GMT clock values retrieved from the running system.

7 A global JCL-variable table, called GVARTAB, is used by this IBM Workload Scheduler for z/OS complex.

8 A server is started for handling the communication to a certain controller. The server verifies that the inbound transactions are requests to the controller for which the server is started.

OSLCOPTS

Purpose

Use this statement to configure the integration with products that provide you with an OSLC interface to create and manage tickets, for example IBM SmartCloud Control Desk.

Format

```
►►──OSLCOPTS──PASSWORD──(──pwd──)──┬────────────────────────────────┬──────────►
                                   │            ┌─ALL───────┐       │
                                   └─POLICY──(──┼─MONITORED─┼──)────┘
                                                └─CRITICAL──┘

►──┬─────────────────────────┬──┬─────────────────────────────┬────────────────►
   │              ┌─3─┐      │  └─TDWCURI──(──basic url──)────┘
   └─PRIORITY──(──┴─n─┴──)───┘

►──┬─────────────────────────┬──TKTURI──(──uri──)──USER──(──userid──)──────────►
   └─TKTDESC──(──string──)───┘

►──┬──────────────────────────┬────────────────────────────────────────────────►◄
   │             ┌─NO──┐      │
   └─USERFLD──(──┴─YES─┴──)───┘
```

Parameters

PASSWORD (*pwd*)
The password associated with the user set in USER, it can be up to 31 characters. The password is stored in plaintext, meaning that it is *not* encrypted. To encrypt it, run the sample EQQBENCO JCL provided in the EQQSAMP library. For details about how to encrypt the password, see *Managing the Workload*.

This parameter is required and does not have a default value.

POLICY(ALL|MONITORED|CRITICAL)
The policy applied to open tickets for jobs that ended in error. The default value is ALL, meaning that a ticket is opened for any job that ends in error. Set this parameter to MONITORED to open tickets only for the jobs that have the monitored flag; set it to CRITICAL to open tickets only for critical jobs or jobs that belong to a critical network.

PRIORITY (*n*|3)
The priority assigned to the tickets that will be opened for jobs that ended in error. Valid values are from 1 (highest) to 5 (lowest), the default is 3.

TDWCURI(*basic url*)
The basic URL to launch the Dynamic Workload Console in context, allowing you to view the job in error and take the appropriate action. This parameter is required if you want to use the &TWDCLINK variable in the string set for TKTDESC.

For example, a basic URL might be:

```
https://mypc:29443/ibm/console/xLaunch.do?pageID=
com.ibm.tws.WebUI.External.navigation&showNavArea=false
&action=BrowseJobs&hostname=webuidev&port=31217&server=C851
```

For detailed information about how to create a basic URL for the Dynamic Workload Console, see the *Administration Guide*.

TKTDESC('*string*')
A detailed description for the ticket, included between single quotation marks. This parameter is valid only if you selected the **DESCRIPTION-LONGDESCRIPTION** check box in the SmartCloud Control Desk (for detailed information about how to set this check box, see *Managing the Workload*).

You can use the following variables:

- `&APPLIA`
- `&APPLNAME`
- `&ERRCODE`
- `&JOBNUM`
- `&OPERNUM`
- `&OPTEXT`
- `&SSNAME`
- `&TWDCLINK` (provided that you set the TDWCURI parameter)
- `&TWSJOBN`
- `&WSNAME`

You can also use user field operation info, provided that you set USERFLD(YES). This parameter is optional.

For example, you might set TKTDESC as follows:

```
TKTDESC('Ticket opened for &APPLNAME &OPERNUM &APPLIA job &TWSJOBN &JOBNUM
ended in error with error code &ERRCODE. Follow the instructions &USRFLD1.
Then use launch in context &TDWCLINK')
```

TKTURI(*uri*)
The URI provided by the product that manages the tickets and provides that OSLC interface. This parameter is required.

USER (*userid*)
The user ID to open the ticket. Ensure that you specify a user authorized to create and manage tickets on the OSLC interface.

This parameter is case-sensitive and can be up to 47 characters. It is required and does not have a default value.

USERFLD(YES|NO)
Specify YES to allow user field operation info in the TKTDESC parameter. The default value is NO.

OUCOPTS

Purpose

The OUCOPTS statement defines the options for storing in a JES SYSOUT class the job logs retrieved by Output collector from the z-centric environment. See *IBM Workload Scheduler for z/OS: Scheduling End-to-end with z-centric Capabilities* for information about sending job logs to the JES SYSOUT.

OUCOPTS must stay in a separate member of the EQQPARM library that is parsed by the output collector alone.

Format

```
►►──OUCOPTS──CNTPARMS──(──member name──)──┬──────────────────────────────┬──────────►
                                          └─JESCLASS──(──JES class name──)─┘

►──┬──────────────────────────────────────────────────┬────────────────────────────►
   │                       ┌─10────────────────┐      │
   └─MINTHREADSPOOL──(─────┴─number of threads─┴──)───┘

►──┬─────────────────────────────────────────────────────┬─────────────────────────►
   │                          ┌─5──────────────────┐     │
   └─MAXSOCKETFORDEST──(──────┴─number of job logs─┴──)──┘

►──┬──────────────────────────────────────────────────┬────────────────────────────►
   │                       ┌─100───────────────┐      │
   └─MAXTHREADSPOOL──(─────┴─number of threads─┴──)───┘

►──┬─────────────────────────────────┬──SUBSYS──(──controller subsystem name──)────►
   │                    ┌─24────┐    │
   └─RETAINPEND──(──────┴─hours─┴──)─┘

►──┬────────────────────────────────┬─────────────────────────────────────────────►◄
   └─WRITER──(──WRITER task name──)─┘
```

Parameters

CNTPARMS(*member name*)
The name of the member in the EQQPARM data set that contains the OPCOPTS, ROUTOPTS, and HTTPOPTS initialization statements.

JESCLASS(*JES class name*)
The name of the JES SYSOUT class where Output collector is to write the job logs after retrieving them from the z-centric environment. The default is the CLASS name specified in the EQQJOBSC (Create sample job JCL) panel or the default class defined for the JES spool.

MINTHREADSPOOL(*number of threads*|**10**)
The minimum number of threads that Output collector can open to copy the job logs to JES.

MAXSOCKETFORDEST(*number of job logs*|**5**)
The maximum number of job logs that Output collector can retrieve concurrently from the same agent or dynamic domain manager in the z-centric environment.

MAXTHREADSPOOL(*number of threads*|**100**)
The maximum number of threads that Output collector can open to copy the job logs to JES.

RETAINPEND(*hours*|**24**)
The time (in hours) that Output collector will attempt to get the log of a job run by a dynamic agent that is temporarily unavailable.

SUBSYS(*controller subsystem name*)
The name of the controller subsystem for which Output collector is started.

WRITER(*WRITER name*)
The user ID associated with the WRITER task that Output collector calls to write the job logs to the JES spool after retrieving them from the z-centric environment. If no value is specified here or in the EQQJOBSC (Create sample job JCL) panel, the default WRITER task is used.

RCLDDP

Purpose

The RCLDDP statement defines a list of protected DD names so that the data set related to them is not deleted by the restart and cleanup function.

RCLDDP is defined in the member of the EQQPARM library as specified by the DDPRMEM parameter in the RCLOPTS statement.

Format

```
                          ┌─,─────────────────┐
                          ▼                   │
►►──RCLDDP──DDNAME──(───────list of DD names──┴────)──────────────────────────►◄
```

Parameters

DDNAME(*list of DD names***)**
: Defines the list of the protected DD names.

RCLDSNP

Purpose

The RCLDSNP statement defines a list of protected data set names that are not deleted by the restart and cleanup function.

RCLDSNP is defined in the member of the EQQPARM library as specified by the DSNPRMEM parameter in the RCLOPTS statement.

Format

```
                            ┌─,──────────────────────┐
                            ▼                        │
►►──RCLDSNP──DSNAME──(────────list of data set names─┴────)─────────────────►◄
```

Parameters

DSNAME (*list of data set names***)**
: Defines the list of the protected data set names. You can use the asterisk (*) as a wild character for partial matching, as explained for the RCLOPTS DSNPROT keyword

RCLOPTS

Purpose

The RCLOPTS statement defines all the options used by IBM Workload Scheduler for z/OS during the restart and cleanup functions. It is used only by the controller.

RCLOPTS is defined in the member of the EQQPARM library as specified by the PARM parameter in the JCL EXEC statement.

Note: Because in some cases EQQCLEAN might delete a data set by mistake, it is recommended that you protect critical data sets from deletion by using either the RCLOPTS parameters (DDPROT, DDPRMEM, DSNPROT, DSNPRMEM) or the EQQUXCAT exit.

Format

```
►►──RCLOPTS──┬──────────────────────────────────────────┬──────────────────►
             │                  ┌─'OPC'─────────┐       │
             └─CLNJOBCARD──(──┴─Job card info─┴──)─┘

►─┬──────────────────────────────────┬─┬──────────────────────────────┬──────►
  │               ┌─EQQCL────┐       │ └─DDALWAYS──(────DDlist────)─┘
  └─CLNJOBPX──(──┴─Job name─┴──)─┘

►─┬────────────────────────────┬─┬─────────────────────────────┬────────────►
  └─DDNEVER──(────DD List────)─┘ └─DDNOREST──(────DD List────)─┘

►─┬─────────────────────────────────┬─┬────────────────────────────┬────────►
  └─DDPRMEM──(────member name────)─┘ └─DDPROT──(────DD List────)─┘

►─┬──────────────────────────────────┬─┬──────────────────────────────┬─────►
  └─DSNPRMEM──(────member name────)─┘ └─DSNPROT──(────DSN List────)─┘

►─┬───────────────────────────────────────────┬─────────────────────────────►
  └─DSTCLASS──(──destination:class──)─┘

►─┬─────────────────────────────────────────┬─┬─────────────────────────┬───►
  └─DSTDEST──(────OPC destination────)─┘    │            ┌─NO──┐       │
                                              └─DSTRMM──(──┴─YES─┴──)─┘

►─┬─────────────────────────────────────────────────┬───────────────────────►
  └─DUMMYLASTSTEP──(────STEPNAME/'IBM50941'────)─┘

►─┬────────────────────────────┬─┬──────────────────────────────────────┬───►
  │               ┌─NO──┐      │ │                 ┌─FIRSTSTEP─┐        │
  └─GDGSIMAUTO──(──┴─YES─┴──)─┘ └─IMMEDLOGIC──(──┴─BESTSTEP──┴──)─┘

►─┬─────────────────────────────────────────┬───────────────────────────────►
  │                      ┌─ONDEMAND─┐       │
  └─JOBLOGRETRIEVAL──(──┴─ONERROR──┴──)─┘

►─┬───────────────────────────────────────────────┬─────────────────────────►
  │                            ┌─ONDEMAND─┐       │
  └─RESTARTINFORETRIEVAL──(──┴─ONERROR──┴──)─┘

►─┬──────────────────────────────────┬─┬──────────────────────────┬─────────►
  └─SKIPINCLUDE──(──member name──)─┘ └─STEPLIB──(──dsname──)─┘

►─┬────────────────────────────┬───────────────────────────────────────────►◄
  │               ┌─YES─┐      │
  └─STEPRESCHK──(──┴─NO──┴──)─┘
```

Parameters

CLNJOBCARD(*job card info***|'OPC')**

Defines the Job card to be used by the controller while creating stand-alone cleanup jobs. The default job card is the following:

```
//jobname JOB 'OPC'
```

where jobname is built from CLNJOBPX value combining the prefix and a sequential alphanumerical number (for instance EQQCL0005). If you need to change 'OPC' with account information, you can use the CLNJOBCARD keyword that will substitute 'OPC' with the specified value.

It is optional and the default is 'OPC'. The string can be up to 40 characters long.

Do not use it to set the submitter of the stand-alone cleanup job. For that purpose, refer to the RUSER parameter of the job-submit exit.

CLNJOBPX(*job name***|**__**EQQCL**__**)**
Defines the job name prefix to be used for stand alone cleanup.

It is optional and the default is EQQCL.

If specified, it must be five characters long, otherwise the default value is used. In order to track correctly the stand alone cleanup job in a DASD connection, the SU/RE data set must always be defined.

DDALWAYS(*DD List***)**
Defines the list of the DD names that make a step always re-executable. This keyword is optional and valid only for Step Restart; for Job Restart it is ignored.

DDNEVER(*DD List***)**
Defines the list of the DD names that make a step never re-executable. This keyword is optional and valid only for Step Restart; for Job Restart it is ignored.

DDNOREST(*DD List***)**
Defines the list of the DD names that make a step not restartable. This keyword is optional and valid only for Step Restart; for Job Restart it is ignored.

DDPRMEM(member name)
When specified, it contains the name of the partitioned data set member of the parameter library that lists the protected DD names. The syntax in the member is the following:

- **RCLDDP**
- **DDNAME (list of DD names)**

DDPRMEM is mutually exclusive with DDPROT. It can be refreshed using the following MODIFY operator command:

```
/F procname, PROT(DD=name)
```

where *procname* is the IBM Workload Scheduler for z/OS JCL procedure name.

DDPROT(*DD List***)**
Defines the list of the DD names that identify the protected data sets. It is optional.

DSNPRMEM (member name)
When specified, it contains the name of the partitioned data set member of the parameter library that lists the protected data set names. The syntax inside the member is the following:

- **RCLDSNP**
- **DSNAME (list of data set names)**

You can use the * (asterisk) as a wild character for partial matching as explained for the DSNPROT keyword. DSNPRMEM is mutually exclusive with

DSNPROT. You can refresh the DSN list by changing the DSNPRMEM member name with the following MODIFY operator command:

```
/F procname, PROT(DS=name)
```

where *procname* is the IBM Workload Scheduler for z/OS JCL procedure name.

DSNPROT(DSN List)

An optional keyword that defines the list of data set names protected from unintentional deletion. The data sets specified in this list are excluded from the cleanup action list (you will see these data sets marked as X, excluded, in the ISPF panel listing the data sets to be deleted).

You can use the asterisk (*) as a wildcard character for partial matching, provided that you specify it as the last character of the data set name. Any character following the asterisk is ignored. For example, to protect the data set name MYDSN.DATASET and all the GDGs with root MYGDG.ROOT, you can specify the following:

```
DSNPROT (MYDSN.DATASET,MYGDG.ROOT*)
```

The result will be that all the GDG data set names MYGDG.ROOT.G*nnnn*V*nn* are protected in addition to the data set name MYDSN.DATASET.

If you specify:

```
DSNPROT (P*.PROD.FILE)
```

all the files beginning with the letter P are taken into account, regardless of what is specified in the rest of the data set name. It has the same result as if you specify `DSNPROT (P*)`.

DSTCLASS(*destination:class*)

When you need to have a configuration with the data store subsystem and the tracker with the JCC task active on the same z/OS system image, there could be compatibility problems if the JCC task options ask IBM Workload Scheduler for z/OS to delete the SYSOUT output data sets after the usual analysis. This is because the JCC task might also delete the duplicated SYSOUT copy created for the data store before it has been successfully stored. In this specific configuration, to avoid this problem and to improve the JCC performance that would be scanning the same SYSOUT data sets twice, you need to provide a JES class associated to the tracker destination, to be used for the duplicated copy of the SYSOUT data sets created for data store processing.

The duplicated copy of the SYSOUT will be temporarily added to this class until the data store retrieves, stores, and deletes it. It does not need to be a reserved class: the only mandatory characteristic is that it must not be one of the classes specified in the JCC parameter CHKCLASS. In this way the JCC task will never process the output data sets meant for data store processing.

In a JES3 system, you must define the DSTCLASS as HOLD=EXTWTR and TYPE=PRINT.

You should use the same class for all the trackers and systems, as it is in general easier and less exposed to potential error situations. This suggestion becomes a requirement if the configuration is in a JES MAS complex or if the user specified NJE statements to route the output after the job execution.

The keyword value is specified in pairs; the tracker destination followed by the JES class.

Values of destination and class pairs are specified as in this format:

Dest1:**Class1**

where *Dest1* is the tracker destination as specified in the ROUTOPTS statement and *Class1* is the JES class to use for the SYSOUT data sets duplicated for the data store processing. A colon separates destination and class values; a comma separates destination and class pairs. The destinations specified must be consistent with those defined in the ROUTOPTS statement. The tracker destination of 8 asterisks (********) identifies the system where the controller and a local tracker run.

Note: This mechanism will allow JCC and data store to work correctly at the same time but it will not prevent the JCC task from deleting the user SYSOUT of the jobs. So, the user SYSOUT, after being checked by JCC, will not be displayed with the rest of the system sysout.

When the scheduler selects an operation for submission, the DSTCLASS() values are checked to determine if a CLASS= parameter has to be inserted in the output JCL statement that is automatically added to the original JCL. Remember to specify a pair in the ********:X format to obtain the CLASS=X parameter generated for jobs scheduled on the same processor as the controller, that is, on computer workstations with blank destination. For example:

```
RCLOPTS DSTCLASS(********:Q,TRKRSYS1:Q,TRKRSYS2:Q)
```

This example specifies a pair list suitable for a two-member JES2 MAS, with a tracker on the same LPAR where the controller runs and another tracker on the other LPAR. The tracker destinations are TRKRSYS1 and TRKRSYS2 respectively. If you omit the first pair, jobs scheduled on workstations with blank destinations will be submitted without CLASS=Q added to the generated output statement.

DSTDEST(*OPC destination*)
Defines the destination required in the JCL to create a sysout copy for data store. Its value must be equal to the data store SYSDEST parameter of the DSTOPTS statement.

The DSTDEST value should be reserved for the scheduler's use.

If the JES2 DESTDEF statement specifies NODENAME=REQUIRED, then the destination specified in the DSTDEST parameter must be defined as JES2. You can dynamically define it as JES by using the following command:

```
$ADD DESTID(XXXXXXXX),DEST=XXXXXXXX
```

where `XXXXXXXX` is the destination specified in the DSTDEST parameter. If JES2 DESTDEF specifies NODENAME=OPTIONAL, then the destination defined on DSTDEST does not need to be specified.

DSTRMM(YES|NO)
If you specify YES, Removable Media Manager (RMM) activates and the cleanup uses RMM API against tape volumes defined to RMM.

DUMMYLASTSTEP(*STEPNAME*/'*IBM50941*')
Use this keyword to allow RESTART and CLEANUP function correct handling of temporary data set when they are referenced with DISP=PASS in the last step of the submitted JCL. The problem is that in this specific situation, RESTART and CLEANUP could suggest to delete the wrong data sets. To avoid this problem, you can use one of the following solutions:

- (1) Change the disposition to keep, by setting on trackers: OPCOPTS RCLPASS(YES)

  OR

- (2) Add a new dummy step as the final step to all jobs submitted by IBM Workload Scheduler for z/OS (using this keyword).

Solution (1) results in temporary files being retained and this might eventually cause DAD problems as volumes might be become full since these files are not deleted. Use this solution when EWTROPTS RETCODE(LAST) is used.

Solution (2) can be used if EWTROPTS RETCODE(HIGHEST) is used.

If you specify this keyword, a dummy last step is added to each job submitted by IBM Workload Scheduler for z/OS as follows:

```
// STEPNAME  EXEC PGM=IEFBR14,COND=EVEN
```

where,

STEPNAME
A string of characters with maximum length of 8. If you leave it blank, the default value, `IBM50941`, is used.

Note:

- No check is performed to determine if a last dummy step already exists with the same name.
- If an end of job is found within the JCL, all the JCL lines starting from end of job are commented out except the last dummy step you just added.
- No check is performed to determine if the maximum number of steps within the JCL has been reached.

  If you have a JCL with this problem:

  – If the JCL does not need data set cleanup actions, or the JCL has no DISP=PASS files, add the following line to the JCL to prevent the tailoring and comment it out with the asterisk (*):

  ```
  // EQQJHDMY SKIP
  ```

  You can add this statement anywhere in the JCL, even after the end of the job, except not inside an INCLUDE member or an external or nest procedure. Ensure that there is only one blank between EQQJHDMY and SKIP.

  – If cleanup is needed, consider restructuring the JCL to reduce the number of steps.

GDGSIMAUTO (YES|NO)
This keyword applies only to operations having clean up type automatic. It defines if the GDG simulation process is applied when operations are rerun internally and expanded jcl is not used. For details about the GDG simulation process, see *Managing the workload - restart and clean up - GDG resolution*. When rerun is started by dialogue, the GDG simulation is done accordingly to the dialogue rules that are:

STEP RESTART
Always applied if expanded jcl is not used

JOB RESTART
Never applied

Note that, when we say that the GDG simulation is applied, it means that whenever needed by the submitted job jcl definitions, the GDG data set is simulated. NO is the default. YES means that we apply the GDG simulation process to all operations having automatic clean up type for internal rerun and that expanded jcl is not used.

IMMEDLOGIC(BESTSTEP|FIRSTSTEP)
Defines the logic for running cleanup actions when an operation ends in error and the cleanup type is Immediate. FIRSTSTEP means that the cleanup actions are performed considering the first step as the starting step. However, if the automatic recovery actions suggest a restart step, this will be considered as the starting step.

BESTSTEP means that the cleanup actions are performed considering the best step as the starting step. However, if the automatic recovery actions suggest a restart step, this will be considered as the starting step. For details about how the best step is determined, refer to *Managing the Workload*.

Note: If you specify BESTSTEP and rerun a job with Cleanup=Immediate without using the Restart and Cleanup-Step Restart path, the cleanup actions might not be coherent with the step range.

JOBLOGRETRIEVAL(ONERROR|ONDEMAND)
This parameter defines the joblog retrieval policy for operations running on computer automatic workstations producing an MVS Joblog. The valid values are:

ONDEMAND
The default value. The user must explicitly make a request to retrieve the joblog.

ONERROR
After a job runs and ends in error, the joblog is automatically retrieved and sent. Manual changes to the error do not trigger the retrieval process to start.

RESTARTINFORETRIEVAL(ONERROR|ONDEMAND)
To execute restart and cleanup actions on an operation, such as step restart or data set cleanup, the restart information must be extracted from the job logs. This parameter defines the restart information retrieval policy for operations running on computer automatic workstations producing an MVS Joblog.

ONDEMAND
The default value. The restart information retrieval process is started as soon as a restart and cleanup action, which requires the restart information, is either requested specifically by a user, or automatically.

ONERROR
The restart information retrieval process is started when an operation ends in error. Manual changes to the error do not trigger the retrieval process to start.

SKIPINCLUDE(member name)

During the JCL tailoring process, the EQQCLEAN pre-step and the //TIVDSTxx OUTPUT statements are added at the beginning of the JCL before the first EXEC statement. They are inserted also before the INCLUDEs that might precede the first EXEC statement, unless they are featured in the SKIPINCLUDE list. In this case, the insertion of the pre-step and other statements is made after the INCLUDEs listed by this keyword. SKIPINCLUDE states the name of the partitioned data set member of the parameter library containing the list of INCLUDE statements that are to be skipped in the JCL tailoring process.

Consider that the scanning of the JCL to be submitted stops at the first INCLUDE statement that precedes the first EXEC statement, which is not listed in the RCLSKIP INCLNAME() list. Thus, if your JCL has OUTPUT statements with JESDS=ALL, which are placed AFTER an INCLUDE, that INCLUDE must be listed in the INCLNAME() list. The subsequent OUTPUT statement is shown and a superfluous TIVDSTAL OUTPUT statement will not be inserted into the JCL.

You should use SKIPINCLUDE if you have a JCL where the first EXEC is preceded by INCLUDEs that contain JCL statements which cannot be placed after an EXEC statement and that do not contain EXEC statements. Examples of JCL statements that cannot be placed after an EXEC statement are JOBLIB and JOBCAT statements, and OUTPUT statements with the JESDS keyword.

This type of INCLUDEs causes an EQQCLEAN step misplacement if you are using normal JCL (expanded JCL usually does not contain INCLUDEs, unless you intentionally add them before you resubmit the JCL).

You can avoid JCL errors caused by the misplacement of EQQCLEAN by inserting these INCLUDEs in the partitioned data set member defined with SKIPINCLUDE.

If an INCLUDE statement contains both JCL statements, which cannot be placed after an exec statement, and an EXEC statement, it must be split into two separate INCLUDEs.

For example, if an INCLUDE statement contains both JOBLIB and EXEC statements, it must be split into two separate INCLUDEs: one with JOBLIB and one with EXECs.

The syntax that is to be used in the partitioned data set member is described in the RCLSKIP statement section.

To refresh SKIPINCLUDE, enter the following MODIFY operator command:

```
/F procname,SKIPINC(membername)
```

where `procname` is the IBM Workload Scheduler for z/OS JCL procedure name.

STEPLIB (dsname)
Defines the name of the data set that has to be over written by the one specified in the //STEPLIB DD card of your EQQCLEAN procedure. This keyword is optional, but if you specify it, you must always define a STEPLIB in your EQQCLEAN procedure, as a DD dummy one. If this parameter is not specified, no change would be made to the EQQCLEAN procedure at all.

STEPRESCHK (NO|YES)
Specifies the possibility to select a step restart range overriding the product logic (for example, a possible restart step could be a non-restartable step). If you specify NO, the product logic check is not performed. The default value is YES. This kind of behavior might lead to JCL errors and it is up to the user to decide when the override is needed.

RCLSKIP

Purpose

The RCLSKIP statement lists the INCLUDEs that you want to keep at the beginning of a JCL when it is tailored by the Restart and Cleanup function. You must write RCLSKIP in the partitioned data set member whose name is specified by the RCLOPTS SKIPINCLUDE keyword (for details about SKIPINCLUDE, see “SKIPINCLUDE(member name)” on page 136).

Format

```
                           ┌─,───────────┐
                           ▼             │
►►──RCLSKIP──INCLNAME──(─────member name─┴───)─────────────────────────►◄
```

Parameters

INCLNAME (*member name***)**
Lists the member names specified on the MEMBER=*keyword* of one or more INCLUDE statements. You can use the asterisk (*) as a wild character for partial matching as follows:

As the last character
For example, if you specify ABC* all the INCLUDEs whose names begin with ABC are left at the top of the JCL.

As the first character
For example, if you specify *ABC all the INCLUDEs whose names end with ABC are left at the top of the JCL.

As the only character
If you specify only the asterisk (*), all the INCLUDEs are left at the top of the JCL.

As a wildcard, you can use the asterisk only once. For example, if you specify:

```
RCLSKIP INCLNAME(*A*B)
```

all the INCLUDEs whose names end with A*B are considered as matching the search criteria, because the second asterisk is considered as a normal character.

If you specify:

```
RCLSKIP INCLNAME(A*B)
```

all the INCLUDEs whose names begin with A are considered as matching the search criteria, because the asterisk is considered as the last character.

If you specify:

```
RCLSKIP INCLNAME(JOBINCL,ZZZ*)
```

the INCLUDE named JOBINCL and all the INCLUDEs whose names start with ZZZ (for example, ZZZ1, ZZZABC, and so on) are considered as matching the search criteria.

The EQQCLEAN pre-step will be added only after these INCLUDEs by the JCL tailoring process (for details about SKIPINCLUDE, see "SKIPINCLUDE(member name)" on page 136).

RESOPTS

Purpose

The RESOPTS statement defines special resource options that the controller uses to process ready operations and special resource events.

RESOPTS is defined in the member of the EQQPARM library as specified by the PARM parameter in the JCL EXEC statement.

Format

```
►►──RESOPTS──┬──────────────────────────────────────────────┬──────────────────►
             │                  ┌─30─────────────────┐      │
             └─CONTENTIONTIME──(─┴─number of minutes─┴──)─┘

►─┬─────────────────────────────────┬─┬──────────────────────────────────────┬─►
  │                ┌─YES───┐        │ │                   ┌─NOCHANGE─┐       │
  └─DYNAMICADD──(──┼─EVENT─┼──)─────┘ └─DYNONCOMPLETE──(──┼─YES──────┼──)───┘
                   ├─OPER──┤                              ├─NO───────┤
                   └─NO────┘                              └─RESET────┘

►─┬───────────────────────────────────┬─┬─────────────────────────────────────┬─►
  │             ┌─0──────────┐        │ │                  ┌─NOCHANGE─┐       │
  └─LOOKAHEAD──(─┴─percentage─┴──)────┘ └─ONCOMPLETE──(────┼─YES──────┼──)───┘
                                                           ├─NO───────┤
                                                           └─RESET────┘

►─┬──────────────────────────────┬──────────────────────────────────────────────►◄
  │             ┌─FREESR──┐      │
  └─ONERROR──(──┼─FREESRS─┼──)───┘
                ├─FREESRX─┤
                └─KEEPSR──┘
```

Parameters

CONTENTIONTIME(*number of minutes*|30)

CONTENTIONTIME determines how long an operation remains on the waiting queue for a special resource before IBM Workload Scheduler for z/OS issues message EQQQ515W.

Specify a number of minutes (1 to 9999) that an operation must wait before IBM Workload Scheduler for z/OS issues message EQQQ515W. Once issued, the message is not repeated for the same special resource and operation, although IBM Workload Scheduler for z/OS can issue more than one message for an operation if it is on more than one waiting queue.

Note: You must also specify an alert action for resource contention on the ALERTS statement or the message will not be issued. The ALERTS statement is described in “ALERTS” on page 7.

DYNAMICADD(EVENT|OPER|NO|YES)

If a special resource is not defined in the current-plan extension file or special resource database, DYNAMICADD determines if IBM Workload Scheduler for z/OS creates a special resource in response to an allocate request from a ready operation or to a resource event created through the EQQUSIN or EQQUSINS subroutine, SRSTAT TSO command, API CREATE request, or a RODM notification.

Specify YES, which is the default value, if IBM Workload Scheduler for z/OS should create a special resource in the current plan; the special resource database is not updated. IBM Workload Scheduler for z/OS uses defaults to create the resource. When creating the resource, IBM Workload Scheduler for z/OS selects field values in this order:

1. Values supplied by the allocating operation or event. An operation can specify a quantity, an event can specify quantity, availability, and deviation.
2. IBM Workload Scheduler for z/OS defaults.

Specify NO if IBM Workload Scheduler for z/OS should not dynamically create a special resource. If an operation attempts to allocate the special resource, it receives an allocation failure, and the operation remains in status A or R with the extended status of X. If a resource event is received for the undefined resource, an error message is written to the controller message log.

Specify EVENT if IBM Workload Scheduler for z/OS should create a special resource in the current plan, only in response to a resource event. Resources are not created by operation allocations. But if the CREATE keyword of an SRSTAT command has the value NO, the special resource is not created.

Specify OPER if IBM Workload Scheduler for z/OS should create a special resource in the current plan, only in response to an allocate request from a ready operation. Resources are not created by events.

A dynamically created resource has these values if no description is found in the database:

Special resource
The name specified by the allocating operation or resource event.

Text Blank.

Specres group ID
Blank.

Hiperbatch
No.

Used for
Control.

On error
Blank. If an error occurs, IBM Workload Scheduler for z/OS uses the value specified in the operation details or, if this field is also blank, the value of the ONERROR keyword of RESOPTS.

Available
The value specified by an event (Y or N) or blank.

Quantity
The value specified by an event (1 to 999999) or blank.

Deviation
The value specified by an event (-999999 to 999999) or blank.

Default values
The resource has these values that are defaults for quantity and availability:

Quantity
1. Or the quantity specified by an allocating operation. The default quantity is automatically increased if contention occurs, but only for dynamically created resources.

Available
Yes.

Intervals
No intervals are created.

Workstations
The resource has default value asterisk (*), which means all workstations. Operations on all workstation can allocate the resource.

Also see the DYNAMICADD keyword of BATCHOPT in the list of BATCHOPT "Parameters" on page 22, which controls the dynamic creation of undefined special resources during planning.

Note:

1. If IBM Workload Scheduler for z/OS subscribes to a RODM class or object for a resource that does not exist in the current plan, the event created from the data returned by RODM causes a dynamic add of the resource, if DYNAMICADD has the value YES or EVENT.
2. It is strongly recommended that, if the feature of dynamic addition of special resources is used, because almost always a special resource dynamically added does not match the previously listed criteria of being automatically deleted, DYNAMICDEL(YES) be specified in the BATCHOPT statement of the DP batch job.

DYNONCOMPLETE(YES|NO|RESET|NOCHANGE)

This keyword defines the value to which the global availability of the special resource is reset when the operation that uses that resource completes. It applies only to special resources that are dynamically added. This value is used by IBM Workload Scheduler for z/OS only if the On Complete field is blank in the operation definition and special resource definition.

You can specify these values:

NOCHANGE
No action is taken. This is the default.

YES The global availability of the special resource is reset to YES.

NO The global availability of the special resource is reset to NO.

RESET
The global availability of the special resource is reset to blank.

LOOKAHEAD(*percentage*|0)

Specify this keyword if you want IBM Workload Scheduler for z/OS to check before starting an operation whether there is enough time before the resource becomes unavailable. You specify the keyword as a percentage of the estimated duration. For example, if you do not want IBM Workload Scheduler for z/OS to start an operation unless the required special resource is available for the whole estimated duration, specify 100. Specify 50 if at least half the estimated duration must remain until the resource is due to be unavailable. If you specify LOOKAHEAD(0), which is also the default, the operation is started if the special resource is available, even if it will soon become unavailable.

IBM Workload Scheduler for z/OS uses this keyword only if the special resource is used for control.

ONCOMPLETE(YES|NO|RESET|NOCHANGE)

This keyword defines the value to which the global availability of the special resource is reset when the operation that uses that resource completes. This value is used by IBM Workload Scheduler for z/OS only if the On Complete field is blank in the operation definition and special resource definition.

You can specify these values:

NOCHANGE
No action is taken. This is the default.

YES The global availability of the special resource is reset to YES.

NO The global availability of the special resource is reset to NO.

RESET
: The global availability of the special resource is reset to blank.

If you use a value different from the default NOCHANGE, then the global availability of all special resources is reset each time an operation using them completes.

ONERROR(FREESRS|FREESRX|KEEPSR|FREESR)

This keyword defines how special resources are handled when an operation using special resources is set to ended-in-error status. The value of the ONERROR keyword is used by IBM Workload Scheduler for z/OS only if the ONERROR field of a special resource in the current plan is blank and the *Keep On Error* value in the operation details is also blank.

You can specify these values:

FREESR
: IBM Workload Scheduler for z/OS frees all special resources allocated by the operation.

FREESRS
: IBM Workload Scheduler for z/OS frees shared special resources and retains exclusively allocated special resources.

FREESRX
: IBM Workload Scheduler for z/OS frees exclusively allocated special resources and retains shared special resources.

KEEPSR
: No special resources allocated by the operation are freed.

IBM Workload Scheduler for z/OS frees or retains only the quantity allocated by the failing operation. Other operations can allocate a special resource if the required quantity is available. Special resources retained when an operation ends in error are not freed until the operation gets status complete.

You can specify exceptions for individual resources in the Special Resources database and in the current plan.

Examples

```
RESOPTS CONTENTIONTIME(10)          1
        DYNAMICADD(YES)             2
        ONERROR(FREESRS)            3
        LOOKAHEAD(200)              4
```

In this example of a RESOPTS statement:

1 IBM Workload Scheduler for z/OS issues message EQQQ515W if an operation has waited 10 minutes to allocate a special resource.

2 If a special resource is not defined in the current plan, IBM Workload Scheduler for z/OS creates the special resource in response to an allocate request from a ready operation or to a special resource event.

3 Shared special resources are freed if the allocating operation ends in error. Exclusively allocated special resources are kept.

4 If there is less than twice (200%) an operation's estimated duration left before the resource is due to become unavailable, IBM Workload Scheduler for z/OS will not start the operation.

RESOURCE

Purpose

The RESOURCE statement identifies special resources that reports are required for. You can specify more than one RESOURCE statement. RESOURCE statements are used when a daily planning job requests special resource reporting. A special resource is included in a report if it exists and if its name matches a value on a RESOURCE statement. If you do not specify RESOURCE, no reports are produced.

You specify RESOURCE statements in the member of the parameter library that contains the BATCHOPT statement.

Format

```
                         ┌─,──────────────────────┐
                         ▼                        │
►►──RESOURCE──FILTER──(────special resource name──┴──)──────────────────►◄
```

Parameters

FILTER(*special resource name,...,special resource name***)**

FILTER values identify special resources that IBM Workload Scheduler for z/OS includes when planned or actual resource-utilization reports are requested by a daily planning job. Each value is compared with special resources known at planning time. If a special resource name matches a filter value, it is reported on. A special resource is selected only once if it matches more than one filter value, even if the values are on different RESOURCE statements. Each value is 1–44 characters and must not contain embedded blanks.

You can specify asterisk (*) and percent (%) anywhere in a value. An asterisk matches any character and any number of characters. A percent sign matches any 1 character.

Examples

```
RESOURCE FILTER(TAPE*,34%%)   1
RESOURCE FILTER(*80)          2
```

In this example of RESOURCE statements, reports are generated for special resources whose names contain:

1 TAPE as the first 4 characters followed by any number of characters, and 34 as the first 2 characters followed by 2 more characters.

2 80 as the last or only characters.

RODMOPTS

Purpose

RODMOPTS statements are used by a controller to monitor special resources through the Resource Object Data Manager (RODM). You can use RODM to track the status of real resources used by IBM Workload Scheduler for z/OS operations.

You specify RODMOPTS statements in the member of the EQQPARM library specified on the RODMPARM keyword of OPCOPTS. A RODMOPTS statement is required for each field in every resource that you want to monitor.

Note:

1. The names of RODM classes, objects, and fields are case-sensitive. Ensure you preserve the case when specifying RODMOPTS statements in the parameter library. Also, if a name contains anything other than alphanumeric or national characters, you must enclose the name in double quotation marks.
2. If IBM Workload Scheduler for z/OS subscribes to RODM for a resource that does not exist in the current plan and the DYNAMICADD keyword of RESOPTS has the value YES or EVENT, the event created from the data returned by RODM causes a dynamic add of the resource. DYNAMICADD is described in the list of RESOPTS"Parameters" on page 139.

Format

```
►►─RODMOPTS─┬──────────────────────────────────────────┬───────────────────────────►
            └─DESTINATION──(───tracker destination ID───)─┘

►─OPCFIELD──(─┬─AVAILABLE─┬─)──OPCRESOURCE──(───Special resource name───)──────────►
              ├─DEVIATION─┤
              └─QUANTITY──┘

►─RODMCLASS──(───RODM class name───)──RODMFIELD──(───RODM field name───)───────────►

►─┬──────────────────────────────────┬─┬────────────────────────────────────────┬──►
  │                 ┌─LAST────────┐  │ └─RODMOBJECT──(───RODM object name───)───┘
  └─RODMLOST──(─────┼─RESET───────┼─)┘
                    └─field value─┘

►─┬────────────────────────────┬─RODMSYSTEM──(───RODM subsystem name───)───────────►
  │               ┌─YES─┐      │
  └─RODMRM2XE──(──┴─NO──┴──)───┘

►─┬──────────────────────────────┬─────────────────────────────────────────────────►
  └─RODMUSER──(───USERID───)─────┘

►─┬──────────────────────────────────────────────────────────────────────────┬─────►◄
  │                ┌─,───────────────────────────────────────────────────┐ │
  │                ▼                                                     │ │
  └─TRANSLATE──(─────┬─C──'──from value──'─┬──:──┬─C──'──to value──'─┬───┴─)┘
                     ├─N──'──from value──'─┤     ├─N──'──to value──'─┤
                     ├─X──'──from value──'─┤     └─X──'──to value──'─┘
                     └─G──'──*──'──────────┘
```

Parameters

DESTINATION(*tracker destination ID*)
: Specifies the destination ID of a tracker that is started on the same z/OS image as the RODM subsystem. This is the same name that you specify on the ROUTOPTS statement.

  Do not specify DESTINATION if the tracker and controller are started in the same address space.

OPCFIELD(AVAILABLE|DEVIATION|QUANTITY)
: Specify the field name in the special resource that you want to monitor through RODM. When RODM notifies a change, IBM Workload Scheduler for z/OS updates:

AVAILABLE
: The *Available* field in the resource. This value overrides the default and interval values.

QUANTITY
: The *Quantity* field in the resource. This value overrides the default and interval values.

DEVIATION
: The *Deviation* field. You use this field to make a temporary adjustment to quantity. IBM Workload Scheduler for z/OS adds quantity and deviation together to decide the amount that operations can allocate. For example, if quantity is 10 and deviation is -3, operations can allocate up to 7 of the resource.

OPCRESOURCE(*Special resource name***)**
Specify the name of the special resource that you want to monitor through RODM.

RODMCLASS(*RODM class name***)**
IBM Workload Scheduler for z/OS can subscribe to a RODM class field or a RODM object field to monitor a special resource. Specify the name of a RODM class field that is used to monitor the special resource. Or specify the name of the class that the object is in, if you monitor the resource through an object field.

RODMFIELD(*RODM field name***)**
Specify the field name in the RODM class or RODM object that is used to monitor the field in the special resource.

RODMLOST(RESET|*field value***|LAST)**
This keyword determines the value that IBM Workload Scheduler for z/OS sets for the special resource field when no communication is possible with the RODM subsystem. If the controller cannot communicate with a RODM subsystem either because a tracker does not respond or RODM does not respond, it issues a warning message. The controller updates all special resources that subscribe to the lost RODM subsystem, according to the value of RODMLOST.

Specify RESET if IBM Workload Scheduler for z/OS uses the value produced by daily planning for the current time.

Specify LAST if IBM Workload Scheduler for z/OS uses the last known value.

If you want IBM Workload Scheduler for z/OS to set a specific value, specify that value for RODMLOST. The value you can specify depends on the field name in the OPCFIELD keyword. You can specify:

AVAILABLE
: A character value Y or N

QUANTITY
: An integer value 1–999999

DEVIATION
: An integer value -999999–999999.

RODMOBJECT(*RODM object name***)**
Specify the name of a RODM object if you use an object field to monitor the special resource. The object name must exist in the class that you specified in RODMCLASS.

RODMRM2XE(NO|YES)
Use this keyword to send a special WTO message EQQRM2XE to the z/OS

console when the connection to an RODM object fails. This message is displayed with one of the following messages issued to EQQMLOG: EQQRM24, EQQRM2, EQQRM26, or EQQRM27. YES is the default. Specify NO to prevent the message from being sent.

RODMSYSTEM(*RODM subsystem name*)
Identifies the RODM subsystem that IBM Workload Scheduler for z/OS sends subscription requests to. This is the name of the RODM address space that you want IBM Workload Scheduler for z/OS to connect to. You can specify it in the &NAME parameter in the RODM startup procedure. If the &NAME parameter is omitted, the default name is the RODM started task name.

RODMUSER (*user ID*)
Defines the user ID used by IBM Workload Scheduler for z/OS to connect to the RODM system. Specify it when the RODM parameter SEC_CLASS is set to *TSTRODM. The default is blank.

TRANSLATE(*from value*:*to value*,......,*from value*:*to value*)
Specify this keyword if values for the RODM field are different to those for the special resource field. For example, an IBM Workload Scheduler for z/OS value `Y` might be represented in RODM by a value 1. Special resource fields have these values:

AVAILABLE
A character value Y or N

QUANTITY
An integer value 1–999999

DEVIATION
An integer value -999999–999999.

Specify a RODM field value that is translated to an IBM Workload Scheduler for z/OS field value when IBM Workload Scheduler for z/OS receives notification of a change from RODM. Translation values are in pairs separated by a colon. Value pairs are separated by a comma. Specify:

C'value'
For character values.

N'value'
For numeric values.

X'value'
For hexadecimal values.

G'*'
For a generic match. A value received from RODM that is not specified in a *from value* is translated to the *to value*. Specify the generic match if you do not know, or have not specified, all values that the RODM field can contain. If IBM Workload Scheduler for z/OS receives a RODM value that is not a valid IBM Workload Scheduler for z/OS field value, a message is written to the controller message log, and the field is not changed.

If you do not specify TRANSLATE, no RODM values are translated.

Examples

```
RODMOPTS RODMSYSTEM(RODB)                        1
         DESTINATION(SYSBTRK)                    2
         OPCRESOURCE(SYSB.TAPE.UNITS)            3
         OPCFIELD(AVAILABLE)                     4
         RODMCLASS(z/OSSYSB_TAPE_UNITS)          5
         RODMFIELD(TAPES_ONLINE)                 6
         TRANSLATE(N'0':C'N',                    7
                   N'1':C'Y',
                   G'*':C'N')
         RODMLOST(Y)                             8
         RODMRM2XE(NO)                           9
         RODMUSER(USERID)                       10
```

In this example of a RODMOPTS statement, RODM is used to monitor the availability of tape units used by IBM Workload Scheduler for z/OS operations:

1	IBM Workload Scheduler for z/OS sends this request to a RODM subsystem, RODB.
2	The tracker on the same system as RODB has destination ID SYSBTRK. The controller sends the request to the tracker, which communicates with RODB through the subsystem interface. RODMTASK(YES) is specified for the tracker.
3	SYSB.TAPE.UNITS is the name of the IBM Workload Scheduler for z/OS special resource. It represents all tape units on SYSB.
4	RODM monitoring is required for the available field in the SYSB.TAPE.UNITS special resource.
5	In RODM, the tape units on SYSB are represented by the class name z/OSSYSB_TAPE_UNITS.
6	IBM Workload Scheduler for z/OS subscribes to class field TAPES_ONLINE of z/OSSYSB_TAPE_UNITS. If the value subfield of TAPES_ONLINE changes, all subscribers are notified.
7	The translate keyword is specified because this example assumes that the value subfield contains only numeric values. In IBM Workload Scheduler for z/OS only values Y and N are permitted for the available field. IBM Workload Scheduler for z/OS translates 0 to N (N'0':C'N'), 1 to Y (N'1':C'Y'), and all other values to N (G'*':C'N') when RODM reports the subfield value.
8	If communication with RODM is lost, the available field of the special resource is set to Y.
9	You do not want message EQQRM2XE to appear.
10	The scheduler uses USR1 to connect to the RODM system.

ROUTOPTS

Purpose

The ROUTOPTS statement defines routing options to a primary, backup, or standby controller. ROUTOPTS defines how a destination is reached.

A destination is used to represent:

- A tracker system connected to the primary or standby controller through shared DASD, SNA (VTAM), XCF, or TCP/IP.

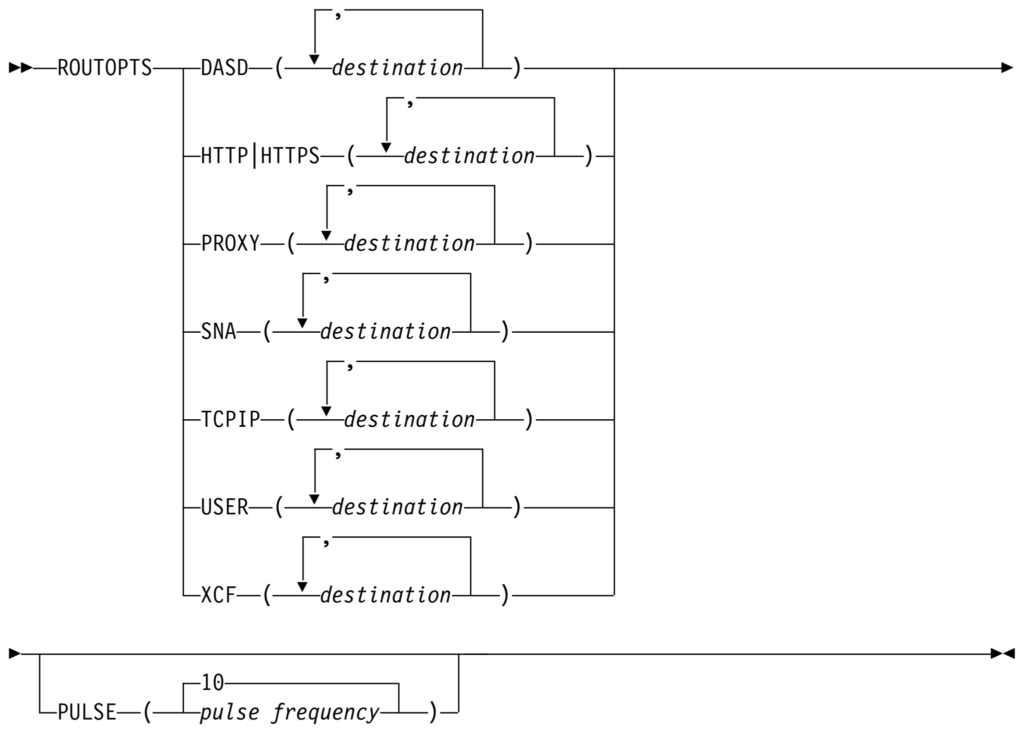
- A tracker that is connected to a backup controller through TCP/IP. In this case, the only connections supported between the tracker and the primary controller are XCF and TCP/IP.
- A user-defined environment where communication is handled using the operation-initiation exit, EQQUX009.
- Remote engines, IBM Workload Scheduler for z/OS Agents, and dynamic domain managers. In this case you use HTTP or HTTPS destination.
- IBM Workload Scheduler for z/OS Agents that are connected to the controller through a proxy server. In this case you use the PROXY destination, along with the HTTP or HTTPS destination.

You can specify more than one ROUTOPTS statement to define routing options, but each destination must be unique. Do not specify the same name in DASD, USER, SNA, XCF, TCP/IP, or HTTP parameters. If a destination is repeated on a following ROUTOPTS statement, the last definition is used. You can specify a combined total of 1000 destinations, but you cannot specify more than 16 destinations for the DASD keyword.

ROUTOPTS is defined in the member of the EQQPARM library as specified by the PARM parameter on the JCL EXEC statement.

You can include as many destinations as you want within the parentheses. They must be separated by commas.

Format



Parameters

DASD(*destination,...,destination***)**
This keyword identifies DASD connections. Each destination name is a submit/release ddname in the controller JCL procedure.

HTTP|HTTPS(*destination,...,destination***)**
Defines the network addresses for HTTP-connected agent workstations, typically remote engines, IBM Workload Scheduler for z/OS Agents, or dynamic domain managers. Use HTTPS to define the HTTP connections as SSL-secure connections.

The syntax of each *destination* is as follows:

```
dest name:'IP address or hostname'/port[/type][/pulseivl][/proxy name]
```

Where:

dest name
The name assigned to the destination, up to 8 alphanumerical characters.

IP address or hostname
The fully qualified host name or IP address used to communicate with the agent workstations. It can be up to 52 alphanumeric characters.

port The port number used to communicate with the agent workstations.

type The HTTP destination type is required only if the destination is used to define a remote engine workstation or a dynamic domain manager. It can be one of the following:

D for a distributed remote engine

Z for a z/OS remote engine

B for a dynamic domain manager

pulseivl
The frequency in minutes that the controller checks the status of the specific z-centric agent or dynamic domain manager. The value can range from 0 (feature inactive) to 1440 minutes and overrides any value specified globally by the PULSEIVL keyword of the HTTPOPTS statement.

Remember: For z-centric agents the `type` is blank, so you must write two consecutive slash (/) characters before the `pulseivl` value.

proxy name
Valid for z-centric agents only. The destination that is defined in the "PROXY(destination,...,destination) " parameter.

You can modify, add, or delete an HTTP or HTTPS destination while IBM Workload Scheduler for z/OS is running.

PROXY(*destination,...,destination***)**
Defines the destinations of the proxies through which the z-centric agents are connected to the controller.

The syntax of each *destination* is as follows:

```
proxy name:'IP address or hostname'/port
```

Where:

proxy name
The name assigned to the destination, up to 4 alphanumerical characters. This value must be also specified in the "HTTP|HTTPS(destination,...,destination) " on page 149 parameter.

IP address or hostname
The fully qualified host name or IP address used to communicate with the proxy server. It can be up to 52 alphanumeric characters.

port The port number used to communicate with the proxy server.

PULSE(*pulse frequency***|<u>10</u>)**
This keyword defines the duration between handshakes (ID events) initiated by trackers. The ID event describes the system environment and options used by the tracker. If the controller does not receive an ID event from the destination for two consecutive pulse intervals, the destination is forced offline by the controller.

Specify a number of minutes from 1 to 60, or specify 0 if handshaking is not required. The default is 10 minutes. PULSE works only with OPC/ESA Release 3 or higher trackers and comes into effect when the controller and the tracker have synchronized at startup. A tracker must have a non-blank destination ID unless the tracker and controller are started in the same address space.

PULSE lets you use workload restart and reroute for DASD-connected trackers and for other connection types in cases where the connection is available but the event writer at the destination is not active. The handshaking process is not performed for user-defined destinations.

Note: The OFFDELAY keyword of JTOPTS is considered after a workstation is varied offline and before offline actions are initiated. In the case of XCF-connected trackers, timing issues might cause conflicts between the pulse parameter and the workstation offline reroute actions.

SNA(*destination,...,destination***)**
This keyword identifies all SNA connections. Each destination is the VTAM LU name of a tracker system. If the SNA keyword is specified in ROUTOPTS, the NCFAPPL must also be specified in the OPCOPTS keyword or the controller issues an error message and ends initialization.

Note: There is no automatic cross-checking between VTAM cross-domain resource definitions and the LU names specified in the SNA keyword. You must ensure the LUs can communicate across VTAM domains, if required.

TCPIP(*destination,...,destination***)**
Defines the network addresses for TCP/IP-connected trackers that can communicate with the controller for job-tracking purposes. Each destination consists of a destination name, a host name or IP address, and optionally a port number.

Defining this parameter requires the definition of an OMVS segment for the controller user ID.

The following rules apply to the destination sub-values:

- The destination name can be up to 8 alphanumeric characters. In association with the host name or IP address, it is used to route the submitted work. This sub-value is required.
- The host name or IP address can be up to 52 alphanumeric characters. It can contain a host name or IP address in IPv4 or IPv6 format. Enclose this value in single quotation marks. This sub-value is required.

- The port number can be up to 5 numeric characters. Valid values are from 0 to 65535. This sub-value is optional. The default is 0, meaning that any port number is accepted.
- The destination names and the host name or IP address must be separate by a colon.
- The required values and the port number must be separated by a slash.

On the primary and backup controller, the destination name set for ROUTOPTS XCF and ROUTOPTS TCPIP respectively, must be the same.

USER(*destination,...,destination*)
This keyword identifies user-defined destinations. Each destination name is an alphanumeric name consisting of 1-8 characters, where the first character is alphabetic. The communication protocol and session control handling is defined in the operation-initiation exit, EQQUX009. The exit is located in the IBM Workload Scheduler for z/OS controller. It is called when operations are ready to be started at a workstation that specifies a destination defined in the USER keyword.

This keyword also defines the destination names specified on the automation workstation to identify the target NetView domain ID where the System Automation commands are to be run.

XCF(*destination,...,destination*)
This keyword identifies all XCF destinations. Each destination is the XCF member name of a tracker. The XCF members that you list must be part of the same XCF group as the controller. If the XCF keyword is present in ROUTOPTS, the XCFOPTS statement must also be present. If no XCFOPTS statement is found, the controller issues an error message and ends initialization.

If a destination listed in the XCF keyword is not an active member of the same XCF group as the controller, no work is transmitted to this destination.

On the primary and backup controller, the destination name set for ROUTOPTS XCF and ROUTOPTS TCPIP respectively, must be the same.

Examples

```
ROUTOPTS XCF(SYS1,SYS2)                                                       1
         SNA(SYSA)                                                            2
         TCPIP(DEST1:'1.111.111.111'/4444)                                    3
         DASD(SYSB)                                                           4
         USER(USRGRP1)                                                        5
         HTTP(ZCENT1:'192.27.144.12'/44112, ZCENT2:'192.14.55.231'/61424//5)  6
         HTTPS(REMZ1:'192.27.144.13'/44113/Z)                                 7
```

In this example of a ROUTOPTS statement, the controller is connected to four trackers running on z/OS. Work is also submitted to an OS/2 system through the operation-initiation exit:

1 SYS1 and SYS2 are connected through XCF communication links and are defined in the XCF keyword.

2 Communication with SYSA is through a VTAM link defined in the SNA keyword.

3 DEST1 identifies a TCP/IP link with a tracker and the details of it are defined in the TCPIP keyword.

4 The controller sends work to a tracker using a submit/release data set. The ddname of the submit/release data set in the JCL procedure of the controller is SYSB, as specified in the DASD keyword.

5 The controller calls the operation-initiation exit, EQQUX009, when operations are ready to be started on workstations that specify the USRGRP1 destination.

6 ZCENT1 and ZCENT2 specify the links to two IBM Workload Scheduler for z/OS Agent workstations. ZCENT1 and ZCENT2 are also to be specified as the destination names in the workstation definitions of the IBM Workload Scheduler for z/OS Agents.

The status of ZCENT2 is checked every 5 minutes by the controller.

7 REMZ1 specifies the link to a z/OS remote engine workstation. REMZ1 is also to be specified as the destination name in the definition of the remote engine workstation.

SERVOPTS

Purpose

The SERVOPTS statement is for a server which handles transactions directed to the controller subsystem name on the same z/OS system as the server.

Format

```
►►─SERVOPTS──┬────────────────────────┬──┬────────────────────────────────────────┬──────────►
             │           ┌─NO──┐      │  │               ┌─IBM - 037────────────┐ │
             └─ARM─(─────┴─YES─┴──)───┘  └─CODEPAGE─(────┴─host system codepage─┴──)─┘

►─┬──────────────────────────────────┬─┬─────────────────────────────────────────┬──────────►
  │              ┌─DBOPT───────┐     │ │                  ┌─local hostname─┐     │
  └─DBOPTPRM─(───┴─member name─┴──)──┘ └─JSCHOSTNAME─(────┼─hostname───────┼──)──┘
                                                          └─IP address─────┘

►─┬────────────────────────────┬─┬────────────────────────────────────┬─────────────────────►
  │               ┌─425───┐    │ │                ┌─,──────────┐      │
  └─PORTNUMBER─(──┴─value─┴──)─┘ │                ▼  ┌─APPC─┐  │      │
                                 └─PROTOCOL─(─────────┼─E2E──┼──┴───)─┘
                                                      └─TCP──┘

►─┬──────────────────────────────────────┬─SUBSYS─(──Controller Subsystem name──)──────────►
  │               ┌─,──────────────┐     │
  │               ▼                │     │
  └─SCHEDULER─(─────Scheduler name─┴──)──┘

►─┬───────────────────────────┬─┬──────────────────────────────────────┬───────────────────►
  │             ┌─YES─┐       │ │               ┌─TPLGPARM────┐        │
  └─TASKUSR─(───┴─NO──┴───)───┘ └─TPLGYPRM─(────┴─member name─┴───)────┘

►─┬────────────────────────────────────────────┬───────────────────────────────────────────►◄
  └─USERMAP─(─parameters library member─)──────┘
```

Parameters

This section describes parameters that apply to all connection types. Parameters that are specific to a connection type are described separately in the sections that follow.

`ARM (YES|NO)`

The z/OS Automatic Restart Manager (ARM) can reduce the impact of an unexpected error to IBM Workload Scheduler for z/OS because z/OS can restart it automatically, without operator intervention.

Specify YES if automatic restart of the failed IBM Workload Scheduler for z/OS component should be activated. ARM recovery of the failed IBM Workload Scheduler for z/OS component is possible in the same image (restart-in-place). This feature allows the recovery of the tracker and a fast restart of the controller and the server. In addition, restart-in-place does not reduce the number of standby controllers when there is a controller failure. The number of restarts and the period of a restart are parameters that can be customized for each IBM Workload Scheduler for z/OS component in the z/OS ARM policy.

`CODEPAGE (`*`host system codepage`*`|IBM–037)`

The name of the host code page. You can provide the IBM–*nnn* value, where *nnn* is the EBCDIC code page. The default value, IBM–037, defines the EBCDIC code page for US English, Portuguese, and Canadian French. If you specify a codepage value different from the default value, a check has been implemented to use the default codepage if the first four characters of the codepage you specify are different from "IBM-". The following is a list of the EBCDIC code pages:

IBM–939	Japan Extended
IBM–937	Taiwan
IBM–935	China
IBM–933	Korea
IBM–975	Greece
IBM–971	Iceland
IBM–970	Latin 2
IBM–838	Thai
IBM–500	International
IBM–424	Israel
IBM–297	France
IBM–285	UK
IBM–284	Spain - Latin America
IBM–280	Italy

IBM–278
: Sweden - Finland

IBM–277
: Denmark - Norway

IBM–274
: Belgium

IBM–273
: Germany

IBM–1388
: China

IBM–1122
: Estonia

IBM–1112
: Baltic

IBM–1047
: Open Systems

IBM–1026
: Latin 5 (Turkey)

IBM–1025
: Cyrillic

The following is a list of the EBCDIC code pages for EURO support:

IBM–1140
: Finland, Sweden

IBM–1141
: Austria, Germany

IBM–1142
: Denmark, Norway

IBM–1143
: USA

IBM–1144
: Italy

IBM–1145
: Spain, spanish-speaking Latin America

IBM–1146
: UK

IBM–1147
: France

IBM–1148
: Belgium, Switzerland

IBM–1149
: Iceland

DBOPTPRM(*member name***|DBOPT)**
Indicates the member of the PARMLIB that contains the parameters to connect to the database and manage the historical data archiving process.

JSCHOSTNAME (*JSChostname***|***IP address***|** *local hostname***)**
Specifies the host name or IP address that is used by a remote application to connect to the server, when PROTOCOL=TCP. It can be up to 52 alphanumeric characters. The default is the host name returned by the operating system.

You can define a virtual IP address for each server of the active controller and the standby controllers. If you use a dynamic virtual IP address in a SYSPLEX environment, when the active controller fails and the standby controller takes over the communication, the remote application automatically switches the communication to the server of the standby controller.

If you specify the TCPOPTS statement for the server, the HOSTNAME parameter overrides the JSCHOSTNAME parameter in the SERVOPTS statement. It applies also if the HOSTNAME parameter is not explicitly defined: in this case, the default value overrides any different value specified in the SERVOPTS statement.

PORTNUMBER (*value*|425)
The port number used by the server when PROTOCOL=TCP. Valid values are from 0 to 65535. The default is 425. This port number is used by the server to connect to the remote application. Select different values for this parameter and the one specified in the TOPOLOGY statement.

If you specify the TCPOPTS statement for the server, the SRVPORTNUMBER parameter overrides the PORTNUMBER parameter in the SERVOPTS statement. It applies also if the SRVPORTNUMBER parameter is not explicitly defined: in this case, the default value overrides any different value specified in the SERVOPTS statement.

PROTOCOL (APPC,E2E,TCP)
Identifies the types of communication used by the server. You can specify any combination of the following values separated by a comma:

APPC For communication with ISPF dialog and PIF, through the APPC protocol.
TCP For communication with ISPF dialog, PIF, and Dynamic Workload Console, through the TCP/IP protocol.
E2E For communication with a distributed environment, through the TCP/IP protocol.

For example, **PROTOCOL(E2E,TCP)** activates all the communication with the server through the TCP/IP protocol.

If you do not specify this keyword, APPC is used as the default value.

Note: Although you can configure one server task to handle multiple protocols, for example PROTOCOL(E2E,APPC,TCP), consider having multiple server tasks, each one with one PROTOCOL function. By using separate server tasks, you can:

- Maximize the time that the server is up and running; you do not need to shut down the server to configure another PROTOCOL value.
- Minimize the occurrence of storage handling problems.

SCHEDULER (*scheduler name*)
Identifies the name of the server as an APPC scheduler. This parameter is used only if PROTOCOL is set to APPC. If you omit this parameter, the started task name is used as scheduler name.

SUBSYS (*controller subsystem name*)
Identifies the controller for which this server is started.

TASKUSR(NO|YES)
Specifies if a started task is to be run with the user ID associated with the task, instead of the user ID associated with the job name.

YES The task is run with the user ID associated with the started task name. This is the default.

NO The task is run with the user ID associated with the job name.

TPLGYPRM(*member name*|TPLGPARM)
Specify this parameter to activate the end-to-end scheduling with fault tolerance capabilities feature, when you set PROTOCOL to E2E.

The specified *member name* is a member of the PARMLIB in which the fault-tolerant end-to-end options are defined by the TOPOLOGY statement.

USERMAP (*parameters library member*)
Defines a member in the file identified by the EQQPARM DD statement in the server startup job. This member contains all the associations between a z/OS connector user and a RACF user ID. If the USERMAP exists, the TMEADMIN security class is ignored.

The syntax of a USERMAP is:

```
USER 'z/OS_connector_user_ID' RACFUSER(matching_RACF_user_ID)
     RACFGROUP(matching_RACF_group_ID)
USER 'z/OS_connector_user_ID' RACFUSER(matching_RACF_user_ID)
     RACFGROUP(matching_RACF_group_ID)
...
USER 'z/OS_connector_user_ID' RACFUSER(matching_RACF_user_ID)
     RACFGROUP(matching_RACF_group_ID)
```

where RACFGROUP can be omitted if *RACF_group_ID* is the default.

For example:

```
USER 'BMDLPS@ITSWB019'      RACFUSER(BMDLPS)
USER 'BMDLPS@ITSWB020'      RACFUSER(BMDLPS)
USER 'BMD1LPS@ITSWB019'     RACFUSER(BMDLPS)
USER 'BMD2LPS@ITSWB020'     RACFUSER(BMDLPS)
USER 'RSOGFK@ITSWB019'      RACFUSER(RSOGFK)
USER 'RSOGFK@ITSWB020'      RACFUSER(RSOGFK)
USER 'RSOLLM@ITSWB019'      RACFUSER(RSOLLM)
USER 'RSOLLM@ITSWB020'      RACFUSER(RSOLLM)
```

Parameters

USER '*z/OS_connector_user_ID*'
The ID of every Dynamic Workload Console user who is authorized to log on to IBM Workload Scheduler for z/OS using the IBM Workload Scheduler for z/OS connector. The ID format is *username@domain*. This parameter is mandatory.

Note that you must also include the ID of the user who installed the IBM Workload Scheduler for z/OS connector (administrator). For more information about associating the administrator to RACFUSER, see "User IDs involved with Dynamic Workload Console" on page 198.

RACFUSER(*RACF_user_ID*)
The RACF user ID defined for every *z/OS_connector_user_ID* specified with the USER keyword. This parameter can be up to 8 characters in length and is mandatory (see RACF user definitions).

RACFGROUP(*RACF_group*)
The RACF group related to the RACF user. This parameter can be up to 8 characters in length and is optional (see RACF group definitions). It is used to set a group different from the default one associated with the specified *matching_RACF_user_ID* of the RACFUSER.

To simplify the task of mapping users in the USERMAP member, you can use the following wildcard characters:

Asterisk (*) for USER
To filter 0 or more characters in *z/OS_connector_user_ID*. Only one asterisk per row is allowed.

Ampersand (&) for RACFUSER
To match in *matching_RACF_user_ID* the string wildcarded in *z/OS_connector_user_ID*.

For example, to add the following users:

```
USER 'BMDLPS@ITSWB019'      RACFUSER(BMDLPS)
USER 'BMD1LPS@ITSWB019'     RACFUSER(BMDLPS)
```

you can enter:

```
USER 'BMD*LPS@ITSWB019'      RACFUSER(BMDLPS)
```

or to add the following users:

```
USER 'RSOGFK@ITSWB020'      RACFUSER(RSOGFK)
USER 'RSOLLM@ITSWB020'      RACFUSER(RSOLLM)
```

you can enter:

```
USER 'RSO*@ITSWB20'      RACFUSER(RSO&)
```

The wildcarded rows are resolved at runtime.

Important:

- Updates to the USERMAP require a stop and restart of the TCP/IP server to become active. But if you use the wildcards, and you want to add a Dynamic Workload Console user that falls within an already wildcarded definition, this does not require to restart the server. For example, this would be the case if the USERMAP already contains the following entry:
  ```
  USER 'RSO*@ITSWB20'      RACFUSER(RSO&)
  ```
  In this example, the new user RSO115@ITSWB20 (who has already been defined to RACF as RSO115) can connect to IBM Workload Scheduler for z/OS without the need of a USERMAP update and of a server restart.
- The '*' and '&' characters respectively represent the '5C' and '50' hexadecimal characters of the 037 codepage. If you run on a different codepage, replace '*' and '&' with the characters that match '5C' and '50' HEX on that codepage.
- When the '&' character is processed at runtime to match the string wildcarded in *z/OS_connector_user_ID*, the resulting RACFUSER entry may exceed the 8-character limit. For example, to add user:
  ```
  john.smith@mydomain.com
  ```
  in the USERMAP, you enter:
  ```
  USER 'JOHN.*@MYDOMAIN.COM'    RACFUSER(RACF&)
  ```
  At runtime the RACFUSER entry is resolved as RACFSMITH, which is 9 characters.

  When this happens, the RACFUSER entry is automatically truncated to 8 characters and message EQQPH70W is issued to communicate that this action was taken. In this example RACFUSER becomes RACFSMIT.
- To activate auditing on the mapping tasks between USER and RACFUSER, edit member EQQPH6 to set MLOG=YES in the headers of messages EQQPH64, EQQPH65, and EQQPH66. Then run a RECYCLE of the TCP/IP server.

TCPOPTS

Purpose

The optional TCPOPTS statement defines local attributes for a product component acting as partner in a TCP/IP communication. Decide whether to specify it by considering each component according to a client-server model:

Client role
It is the role of:
- The tracker started task, in a tracker-to-controller communication.
- The data store started task, in a data store-to-controller communication.
- The remote interface (ISPF dialog or PIF program), in a remote interface-to-server communication.

Server role
It is the role of:
- The controller started task, in a tracker-to-controller or data store-to-controller communication.
- The server started task, in a remote interface-to-server communication.

TCPOPTS does not apply to connections with z-centric agents. Use ROUTOPTS to define options for connecting with these agents.

Most of the TCPOPTS parameters, depending on which component specifies them, can affect different functional areas: automatic connection restart after a standby controller takeover (exploiting Dynamic Virtual Internet Protocol Addressing - VIPA), firewall management, Secure Sockets Layer (SSL), connection timeout management. The following table groups the TCPOPTS parameters by functional area and interested component:

	Client role	Server role
Automatic restart through Dynamic VIPA		HOSTNAME valid for controller or server started task.
Firewall management		HOSTNAME valid for controller or server started task. TRKPORTNUMBER valid for controller started task. DSTPORTNUMBER valid for controller started task. SRVPORTNUMBER valid for server started task.
Connection timeout	CONNTIMEOUT	

	Client role	Server role
SSL	SSLLEVEL SSLKEYSTORE SSLKEYSTOREPSW SSLAUTHMODE SSLAUTHSTRING Specify the same values for all the communication partners.	SSLLEVEL SSLKEYSTORE SSLKEYSTOREPSW SSLAUTHMODE SSLAUTHSTRING Specify the same values for all the communication partners.
Tivoli Enterprise Portal interface		TCPIPJOBNAME valid for controller started task

You can define the TCPOPTS statement in the parameter file identified by the following DD statements:

- EQQPARM, in the controller procedure.
- EQQPARM, in the tracker procedure.
- EQQPARM, in the data store procedure.
- EQQPARM, in the server procedure.
- EQQYPARM, in the TSO logon procedure of the dialog user.
- EQQYPARM, in the JCL used to run the PIF application.

Format

```
►►──TCPOPTS──┬──────────────────────────────────────────────┬──────────────────────────────────►
             │                  ┌─60─────────────────────┐   │
             └─CONNTIMEOUT──(───┴─TCPIP timeout interval─┴──)─┘

►─┬──────────────────────────────────────────────┬─┬─────────────────────────────┬──────────────►
  │                  ┌─PortNumber─┐              │ │                ┌─NO──┐       │
  └─DSTPORTNUMBER──(─┴─TCPIP port─┴──)───────────┘ └─ENABLEFIPS──(──┴─YES─┴──)───┘

►─┬────────────────────────────────────────┬─┬───────────────────────────────────────┬──────────►
  │             ┌─local hostname─┐         │ │                   ┌─425────────┐      │
  └─HOSTNAME──(─┼─hostname───────┼──)──────┘ └─SRVPORTNUMBER──(──┴─TCPIP port─┴──)──┘
                └─IP address─────┘

►─┬──────────────────────────────────┬─┬────────────────────────────────────────┬───────────────►
  │                 ┌─CAONLY─┐       │ │                   ┌─tws────────┐       │
  └─SSLAUTHMODE──(──┴─STRING─┴──)────┘ └─SSLAUTHSTRING──(──┴─SSL string─┴──)───┘

►─┬───────────────────────────────────────────────┬─────────────────────────────────────────────►
  └─SSLKEYSTORE──(───SSL keystore db file name───)─┘

►─┬───────────────────────────────────────────────────┬─┬──────────────────────────────┬────────►
  │                                                   │ │                ┌─OFF───┐     │
  └─SSLKEYSTOREPSW──(───SSL keystore pw file name───)─┘ └─SSLLEVEL──(────┴─FORCE─┴──)──┘

►─┬──────────────────────────────────────────────┬─┬────────────────────────────────────────┬──►◄
  │                ┌─TCPIP──────────────────┐    │ │                  ┌─PortNumber─┐        │
  └─TCPIPJOBNAME──(┴─TCPIP started task─────┴──)─┘ └─TRKPORTNUMBER──(─┴─TCPIP port─┴──)────┘
```

Parameters

CONNTIMEOUT(*TCPIP timeout interval*|60)
It defines how many seconds a TCP/IP connection attempt waits before a timeout occurs. It is expressed in seconds. Valid values are from 1 to 10000. The default is 60.

DSTPORTNUMBER(*TCPIP port*|PortNumber)
The local TCP/IP port number used by the TCP/IP communication subtasks of the controller and data store. Valid values are from 0 to 65535. The default PortNumber value can be one of the following:

423 It applies to the controller only.

0 It applies to the data store, meaning that the process returns the actual value.

ENABLEFIPS(NO|YES)
Indicates whether the SSL communication must comply with FIPS standards. Specify YES to have a FIPS compliant SSL communication. This keyword is ignored if the SSL communication is not enabled.

For more information about how you activate the support for FIPS standard, see*IBM Workload Scheduler for z/OS: Planning and Installation*.

For more information about the FIPS compliance, see Step 23. Activating support for FIPS standard over SSL secured connections.

HOSTNAME(*hostname*|*IP address*| *local hostname*)
The local host name or IP address used by the scheduler component. The default is the IP address returned by TCP/IP. It can be up to 52 alphanumeric characters and specifies a host name or IP address in IPv4 or IPv6 format. Enclose this value in single quotation marks. If you specify this parameter for the server it overrides the JSCHOSTNAME specified in the SERVOPTS statement, if any.

Omitting this parameter might affect how long the server initialization process takes. TCP/IP must free resources used by previously opened connections. Before doing this, it waits the time specified in the TCP/IP profile, through the FINWait2time parameter of the TCPCONFIG statement. When this time limit is reached, the system waits a further 75 seconds before dropping the connection. The default is 600 seconds, but you can specify a lower value. For details about the TCPCONFIG statement refer to *z/OS Communication Server IP Configuration Reference*.

SRVPORTNUMBER(*TCPIP port*|425)
The local TCP/IP port number used by the server. It overrides the PORTNUMBER specified in the SERVOPTS statement. Valid values are from 0 to 65535. The default port number is 425. In a server-to-remote interface communication, this parameter applies to the server only, while the remote interface ignores it: in fact it always uses a port number assigned by TCP/IP as local port.

SSLAUTHMODE(STRING|CAONLY)
The SSL authentication type. Specify one of the following values:

CAONLY
The scheduler checks the certificate validity by verifying that a recognized Certification Authority has issued the peer certificate. Information contained in the certificate is not checked. This is the default value.

STRING
The scheduler checks the certificate validity as described in the CAONLY option. It also verifies that the Common Name (CN) of the Certificate Subject matches the string specified in the SSLAUTHSTRING parameter.

To avoid any communication error, specify the same SSLLEVEL value for the scheduler started tasks that are to communicate with each other.

SSLAUTHSTRING(SSL string|tws)
Defines a string used to verify the certificate validity when you set SSLAUTHMODE to STRING. The string is up to 64 characters. The default is tws.

SSLKEYSTORE(*SSL keystore db filename***)**
Identifies the database containing keys and certificates. It consists of an SSL working directory name and file name, in the format SSLworkdir/TWS.kbd. It is case sensitive. This field is required if the SSLLEVEL parameter is set to FORCE.

SSLKEYSTOREPSW(*SSL keystore pw filename***)**
Identifies the file containing the key password. It consists of an SSL working directory name and file name, in the format SSLworkdir/TWS.sth. It is case sensitive. This field is required if the SSLLEVEL parameter is set to FORCE.

SSLLEVEL(FORCE|OFF)
The SSL authentication type. Specify one of the following values:

OFF The scheduler component does not support SSL authentication for its connections. This is the default value.

FORCE
The scheduler component uses SSL authentication for all its connections. It refuses any incoming connection, if it is not SSL.

To avoid communication errors, specify the same SSLLEVEL value for the scheduler started tasks that are to communicate with each other.

TCPIPJOBNAME(*TCPIP started task***|TCPIP)**
The name of the TCP/IP started task running on the z/OS system where you run the scheduler component. Set this parameter when you have multiple TCP/IP stacks or a TCP/IP started task with a name different from TCPIP.

TRKPORTNUMBER(*TCPIP port***|PortNumber)**
The local TCP/IP port number used by the TCP/IP communication subtasks of the controller and tracker. Valid values are from 0 to 65535. The default PortNumber value can be one of the following:

424 It applies to the controller only.

0 It applies to the tracker, meaning that the process returns the actual value.

Examples

```
TCPOPTS TCPIPJOBNAME('TCPIP')          1
        HOSTNAME('1.111.111.111')      2
        TRKPORTNUMBER(4444)            3
```

In this example of a TCPOPTS statement:

1 The TCP/IP started task name is set to the default value.

2 The IP address 1.111.111.111 identifies the scheduler started task in the TCP/IP network.

3 4444 is the local port number in a tracker-to-controller communication.

TOPOLOGY

Purpose

The TOPOLOGY statement defines the passwords for the users who need to schedule jobs to run on Windows workstations. Omit it if your scheduling environment does not include these workstations.

For a detailed description of this statement, refer to the *IBM Workload Scheduler for z/OS: Scheduling End-to-end with Fault Tolerance Capabilities* manual.

TRGOPT

Purpose

Specify this statement for event-driven workload automation support. It is used by the Java program that creates configuration files for data set triggering.

Format

```
►►──TRGOPT──┬─────────────────────────────────────────────┬──────────────────►
            │                 ┌─IBM - 037────────────┐    │
            └─CODEPAGE──(─────┴─host system codepage─┴──)─┘

►─┬───────────────────────────────────┬─┬──────────────────────────────────┬─►◄
  │                ┌─0─────┐          │ └─WRKDIR──(──working directory──)─┘
  └─TRACELEVEL──(──┴─level─┴──)───────┘
```

Parameters

CODEPAGE (*host system codepage*|IBM–037)
The name of the host code page. You can provide the IBM–*nnn* value, where *nnn* is the EBCDIC code page. The default value, IBM–037, defines the EBCDIC code page for US English, Portuguese, and Canadian French. The following is a list of the EBCDIC code pages:

IBM–939	Japan Extended
IBM–937	Taiwan
IBM–935	China
IBM–933	Korea
IBM–975	Greece
IBM–971	Iceland
IBM–970	Latin 2

IBM–838
: Thai

IBM–500
: International

IBM–424
: Israel

IBM–297
: France

IBM–285
: UK

IBM–284
: Spain - Latin America

IBM–280
: Italy

IBM–278
: Sweden - Finland

IBM–277
: Denmark - Norway

IBM–274
: Belgium

IBM–273
: Germany

IBM–1388
: China

IBM–1122
: Estonia

IBM–1112
: Baltic

IBM–1047
: Open Systems

IBM–1026
: Latin 5 (Turkey)

IBM–1025
: Cyrillic

The following is a list of the EBCDIC code pages for EURO support:

IBM–1140
: Finland, Sweden

IBM–1141
: Austria, Germany

IBM–1142
: Denmark, Norway

IBM–1143
: USA

IBM–1144
: Italy

IBM–1145
: Spain, spanish-speaking Latin America

IBM–1146
: UK

IBM–1147
: France

IBM–1148
: Belgium, Switzerland

TRACELEVEL (*level*|0)

Trace level for internal logging and traces. Possible values are:

0 To receive error messages only.

1 To receive error and warning messages.

2 To receive error, warning, and informational messages.

3 Indicates the *fine* level, to receive the most important messages with the lowest volume.

4 Indicates the *finer* level, to activate entry and exit traces.

5 Indicates the *finest* level, to receive the most detailed tracing output.

The default value is 0.

You find the trace output in the same working directory as specified in the WRKDIR parameter.

WRKDIR (*working directory***)**

The complete path of the working directory for the configuration files build process. Each subsystem must have its own working directory. You can use the same working directory used for end-to-end scheduling with fault tolerance capabilities. This parameter is required and does not have a default value.

Run EQQPCS08 to customize the content of the working directory.

This parameter is required and does not have a default value.

TRROPTS

Purpose

The TRROPTS statement defines routing options from:

- A z/OS tracker that is connected to a primary or standby controller through shared DASD, SNA (VTAM), XCF, or TCP/IP.
- A z/OS tracker that is connected to a backup controller through TCP/IP. In this case, the only connections supported between the tracker and the primary controller are XCF and TCP/IP.

  If the tracker is connected to the primary controller through XCF, ensure that you also set `TCPOPTS HOSTNAME` and `TCPOPTS TRKPORTNUMBER` on the tracker.

Include TRROPTS in the statements for each z/OS tracker in your IBM Workload Scheduler for z/OS configuration, except where the tracker and controller are started in the same address space. Use TRROPTS where `OPCOPTS OPCHOST(NO)` is specified.

TRROPTS is defined in the member of the EQQPARM library as specified by the PARM parameter on the JCL EXEC statement.

Format

```
►►──TRROPTS──┬──────────────────────────────────────────────────────┬──────────────►
             └─BKPHOSTNAME──(──┬─backup controller hostname───┬─)─┘
                               └─backup controller IP address─┘
```

```
►─┬──────────────────────────────────┬─HOSTCON──(──┬─DASD─┬──)────────────────►
  │                   ┌─424───┐      │             ├─SNA──┤
  └─BKPPORTNUMBER──(──┴─value─┴──)───┘             ├─TCP──┤
                                                   └─XCF──┘

►─┬─────────────────────────────────┬─┬─────────────────────────────────────┬─►
  │             ┌─,───────────┐     │ └─TCPHOSTNAME──(──┬─hostname───┬──)───┘
  │             ▼             │     │                   └─IP address─┘
  └─SNAHOST──(───VTAM LU name─┴──)──┘

►─┬─────────────────────────────────────┬────────────────────────────────────►◄
  │                   ┌─424─────────┐   │
  └─TCPPORTNUMBER──(──┴─TCPIP port──┴──)┘
```

Parameters

BKPHOSTNAME (*backup controller hostname***|***backup controller IP address***)**
The host name, or IP address in IPv4 or IPv6 format, of the remote backup controller. Valid values are fully-qualified names up to 52 alphanumeric characters. This parameter is required only if you want to connect to a backup controller; in this case the HOSTCON parameter, which is used to connect to the primary controller, must be set to TCP or XCF.

BKPPORTNUMBER (*value***|424)**
The TCP/IP port number used to communicate with the remote backup controller. Valid values are from 0 to 65535. If not specified, the default value of 424 is used. This parameter is required only if you want to connect to a backup controller; in this case the HOSTCON parameter, which is used to connect to the primary controller, must be set to TCP or XCF.

HOSTCON(DASD|SNA|TCP|XCF)
The HOSTCON keyword identifies the connection that is used when transmitting events to the controller.

If you specify HOSTCON(DASD), you cannot specify EWSEQNO on the EWTROPTS statement.

If you specify HOSTCON(SNA), the SNAHOST keyword must contain the NCF LU name of the controller. This tracker must also have the NCFAPPL keyword specified in the OPCOPTS statement.

If you specify HOSTCON(XCF), the XCFOPTS statement must also be present.

If you specify HOSTCON(TCP), set also TCPHOSTNAME to identify the remote controller.

SNAHOST(*VTAM LU name***,...,***VTAM LU name***)**
The SNAHOST keyword is required for trackers connected to the controller through an SNA link. This keyword defines the VTAM LU name of the controller and any standby controllers. In a hot standby configuration, you can specify several LU names. At initialization, the tracker logs on to the LU at the SNAHOST that becomes active first. That is, the tracker attempts to communicate with the first IBM Workload Scheduler for z/OS started task that is identified as the controller. If you specify the SNAHOST keyword, the HOSTCON keyword must be SNA.

TCPHOSTNAME (*hostname***|***IP address***)**
The host name or IP address in IPV4 or IPV6 format of the remote controller. Valid values are fully-qualified names up to 52 alphanumeric characters. This parameter is required.

TCPPORTNUMBER (*value*|424)
The TCP/IP port number used to communicate with the remote controller. Valid values are from 0 to 65535. If not specified, the default value of 424 is used.

Examples

```
TRROPTS HOSTCON(SNA)       1
        SNAHOST(NCFAPPL1)  2
```

In this example:

1 The tracker is connected to the controller through a VTAM link.

2 The name of the NCF LU used by the controller is NCFAPPL1.

```
TRROPTS HOSTCON(XCF)                 1
        BKPHOSTNAME('9.168.119.53')  2
        BKPPORTNUMBER(924)           3
```

In this example:

1 The tracker is connected to the primarycontroller through an XCF link.

2 The host name of the backup controller to which the tracker is connected.

3 The TCP/IP port number used by the tracker to communicate with the remote backup controller.

USRREC

Purpose

This statement defines the passwords for the users who need to schedule jobs to run on Windows workstations. Omit it if your scheduling environment does not include these workstations or if you choose to define the Windows user ID and password locally on the workstations (in this latter case, you must set LOCALPSW(YES) in the TOPOLOGY statement).

USRREC is defined in the member of the EQQPARM library specified by the USRMEM keyword in the following statement:

For end-to-end scheduling with fault tolerance capabilities
TOPOLOGY. This statement is read at daily plan Symphony renew, replan, or extend phase.

For a detailed description of the TOPOLOGY statement, refer to the *IBM Workload Scheduler for z/OS: Scheduling End-to-end with Fault Tolerance Capabilities* manual.

For end-to-end scheduling with z-centric capabilities
HTTPOPTS. This statement is read at controller startup.

For a detailed description of the HTTPOPTS statement, refer to the *IBM Workload Scheduler for z/OS: Scheduling End-to-end with z-centric Capabilities* manual.

XCFOPTS

Purpose

The XCFOPTS statement defines run-time options for IBM Workload Scheduler for z/OS systems that use services of the cross-system coupling facility (XCF). Specify this statement for a tracker, controller, or standby controller that uses XCF for communication.

XCFOPTS is defined in the member of the EQQPARM library as specified by the PARM parameter on the JCL EXEC statement.

Format

```
►►──XCFOPTS──GROUP──(──XCF group name──)──MEMBER──(──XCF member name──)──────────►

►──┬──────────────────────────────────────┬─────────────────────────────────────►◄
   │              ┌─,─────────────┐        │
   │              ▼               │        │
   └─TAKEOVER──(────┬─HOSTFAIL─┬──┴──)──┘
                    └─SYSFAIL──┘
```

Parameters

GROUP(*XCF group name*)
The name of the XCF group that the IBM Workload Scheduler for z/OS system should join. This is an alphanumeric name consisting of 1 to 8 characters where the first character is alphabetic.

The name of this XCF group must be different from the one defined in the DSTOPS and FLOPTS groups.

MEMBER(*XCF member name*)
The XCF member name that identifies the IBM Workload Scheduler for z/OS system. This is an alphanumeric name consisting of 1 to 8 characters where the first character is alphabetic.

The member name must be unique within the group. If an IBM Workload Scheduler for z/OS system tries to join a group with the same name as an existing member, an error message is issued, and IBM Workload Scheduler for z/OS ends.

TAKEOVER(HOSTFAIL,SYSFAIL)
The TAKEOVER keyword applies to an IBM Workload Scheduler for z/OS system where you specify OPCHOST(STANDBY) on the OPCOPTS statement. It defines the situations when the standby system automatically takes over from the host IBM Workload Scheduler for z/OS system if the host fails. If you have not specified TAKEOVER, IBM Workload Scheduler for z/OS sends a WTO message to the operator console asking the operator to manually start takeover actions. You can specify either one or both of the takeover conditions.

HOSTFAIL
Automatic takeover occurs when the controller fails.

SYSFAIL
Automatic takeover occurs when the system that the IBM Workload Scheduler for z/OS controller is running on fails.

Examples

```
XCFOPTS MEMBER(GRP1STBY)           1
        GROUP(XCFGRP1)             2
        TAKEOVER(HOSTFAIL,SYSFAIL) 3
```

In this example of an XCFOPTS statement:

1 A standby controller has a member name of GRP1STBY.

2 GRP1STBY is a member of the XCF group XCFGRP1.

3 The standby controller automatically attempts to take over the functions of the controller if the controller fails, or if the z/OS system that the controller is running on fails.

Chapter 2. Identifying related initialization-statement parameters

This chapter describes related initialization statements and parameters. You can use this information to identify the parameters to consider when implementing particular functions, and to evaluate the effect on other processes that a function might have. These functions are described:

- “Configuration” on page 170
- “Security” on page 171
- “Generating audit information (JT log data)” on page 172
- “Determining the success or failure of a job” on page 172
- “Recovery” on page 174
  - “Restart and cleanup” on page 174
  - “Automatic job recovery” on page 175
  - “Workload restart” on page 175
- “Performance” on page 176
- “Reporting” on page 177
- “RODM monitoring” on page 177
- “Output processing” on page 178
- “End-to-end scheduling with fault tolerance capabilities” on page 178
- “WLM integration” on page 182
- “External monitoring” on page 182

Table 7 shows statements described in this chapter and functions that they relate to. Chapter 1, “Defining initialization statements,” on page 3 describes all statements in detail.

Table 7. Initialization statements and related functions

Statement	Information about related functions
ALERTS	“Performance” on page 176 and“External monitoring” on page 182
AROPTS	“Security” on page 171 and Chapter 2, “Identifying related initialization-statement parameters”
AUDIT	“Security” on page 171, “Generating audit information (JT log data)” on page 172, and “Performance” on page 176
AUTHDEF	“Security” on page 171, “Generating audit information (JT log data)” on page 172, and “Performance” on page 176
BATCHOPT	“Generating audit information (JT log data)” on page 172, “Reporting” on page 177 and Chapter 2, “Identifying related initialization-statement parameters”
CPUREC	“End-to-end scheduling with fault tolerance capabilities” on page 178
ERDROPTS	“Configuration” on page 170
EWTROPTS	“Configuration” on page 170, “Generating audit information (JT log data)” on page 172, “Determining the success or failure of a job” on page 172, Chapter 2, “Identifying related initialization-statement parameters,” and “Performance” on page 176
FLOPTS	Chapter 2, “Identifying related initialization-statement parameters” and, “Job log retrieval” on page 175
JCCOPTS	“Determining the success or failure of a job” on page 172
JOBREC	“End-to-end scheduling with fault tolerance capabilities” on page 178
JTOPTS	“Generating audit information (JT log data)” on page 172, “Determining the success or failure of a job” on page 172, Chapter 2, “Identifying related initialization-statement parameters,” “Performance” on page 176, and “Reporting” on page 177

Table 7. Initialization statements and related functions (continued)

Statement	Information about related functions
MONOPTS	"External monitoring" on page 182
MONPOL	"External monitoring" on page 182
NOERROR	"Determining the success or failure of a job" on page 172 and Chapter 2, "Identifying related initialization-statement parameters," on page 169
OPCOPTS	"Configuration," Chapter 2, "Identifying related initialization-statement parameters," on page 169, "Performance" on page 176, "RODM monitoring" on page 177, "Output processing" on page 178, "WLM integration" on page 182 and "External monitoring" on page 182
RCLOPTS	Chapter 2, "Identifying related initialization-statement parameters," on page 169
RECOVERY	"End-to-end scheduling with fault tolerance capabilities" on page 178
RESOURCE	"Reporting" on page 177
RODMOPTS	"RODM monitoring" on page 177
ROUTOPTS	"Configuration" and "RODM monitoring" on page 177
SERVOPTS	"Configuration"
TOPOLOGY	"End-to-end scheduling with fault tolerance capabilities" on page 178
TRROPTS	"Configuration"
USRREC	"End-to-end scheduling with fault tolerance capabilities" on page 178
VARSUB	"End-to-end scheduling with fault tolerance capabilities" on page 178
XCFOPTS	"Configuration"

Configuration

These statements and parameters specify your IBM Workload Scheduler for z/OS configuration. They identify IBM Workload Scheduler for z/OS subsystems and the connections between them.

Table 8. Configuration-related parameters

Statement	Keywords	Description
OPCOPTS	OPCHOST	Specifies the subsystem type (tracker, controller, or standby controller).
	NCFTASK	Starts NCF for communication through VTAM.
	EWTRTASK	Starts an event writer to collect events from the z/OS system.
	ERDRTASK	Starts an event reader to transfer events to the event.
	SERVERS	Starts one or more server queues at the controller.
	TPLGYSRV	Starts the IBM Workload Scheduler end-to-end enabler task.
EWTROPTS	EWSEQNO	Event writer also performs event reader function. Separate event reader is not required.
	SUREL	Specifies if the event writer reads a submit/release data set.
ERDROPTS	ERSEQNO	Specifies the event reader and defines the ddname of the input event data set.
	RELDDNAME	Specifies the submit/release data set that release commands are written to in a shared DASD environment.
ROUTOPTS		Identifies routes from the controller to tracker destinations.
TRROPTS		Identifies the route from a tracker to the controller.
XCFOPTS		Identifies XCF connections and specifies when a standby controller performs take over.

Table 8. Configuration-related parameters (continued)

Statement	Keywords	Description
SERVOPTS	SUBSYS	Identifies the controller with which the server communicates.

Security

You specify these parameters to protect IBM Workload Scheduler for z/OS functions and data, and to record access to IBM Workload Scheduler for z/OS data.

Table 9. Security-related parameters

Statement	Keywords	Description
AUTHDEF		Specifies how IBM Workload Scheduler for z/OS resources are defined to RACF
AROPTS	AUTHUSER	Specifies where IBM Workload Scheduler for z/OS retrieves a name for authority checking
	USERREQ	Specifies if a valid user ID is required
AUDIT		Specifies when access to IBM Workload Scheduler for z/OS data is recorded
JTOPTS	JOBCHECK	Specifies if the job name in JCL must match the operation job name
USRREC	USRNAM	Specifies the user name.
	USRPSW	Specifies the user password.
SERVOPTS	USERMAP	Defines a member that contains all the associations between Dynamic Workload Console users (via IBM Workload Scheduler for z/OS connector) and matching RACF user IDs.
TOPOLOGY	LOCALPSW	Specifies if the user ID and password to be used for Windows workstations are to be found locally, when missing from the Symphony file.

You set up the security environment when you install IBM Workload Scheduler for z/OS. You can then customize IBM Workload Scheduler for z/OS security by specifying particular levels of protection. If you use RACF, you perform these steps:
- Add IBM Workload Scheduler for z/OS to the started-procedure table, ICHRIN03. If you use RACF 2.1, you can instead add IBM Workload Scheduler for z/OS to the STARTED class. You need not perform this action if you run IBM Workload Scheduler for z/OS as a batch job.
- Add each IBM Workload Scheduler for z/OS subsystem name to the APPL class. This determines the level of access to the subsystem.
- Add a general resource class for IBM Workload Scheduler for z/OS to the class descriptor table. If you use RACF 2.1, you can use the general resource class supplied for IBM Workload Scheduler for z/OS, IBMOPC.
- Update the router table, ICHRFR01, to specify what action is taken for the resource class.

You can then specify levels of protection for particular IBM Workload Scheduler for z/OS functions and data. The *Planning and Installation* describes how you set up the security environment. Chapter 3, "Implementing security," on page 185 describes in detail how to protect IBM Workload Scheduler for z/OS.

You specify parameters on the AUDIT and AUTHDEF statements to determine when AUDIT information is produced. For more information, see "Generating audit information (JT log data)."

Generating audit information (JT log data)

These parameters determine the amount of auditable information that IBM Workload Scheduler for z/OS produces.

The information is written to the job-tracking log and can be copied at daily planning to the tracklog data set (EQQTROUT). You can invoke AUDIT directly from ISPF dialog, when appropriately customized (for details, see *IBM Workload Scheduler for z/OS: Planning and Installation*).

Table 10. Auditing-related parameters

Statement	Keywords	Description
AUDIT		Record access to IBM Workload Scheduler for z/OS data.
BATCHOPT	NCPTROUT	Specifies if track-log records are copied to EQQTROUT from the NCP at daily planning.
	OCPTROUT	Specifies if track-log records are copied to EQQTROUT from the old CP at daily planning.
	LOGID	Specifies the numeric identifier placed in all records on the track log (EQQTROUT).
EWTROPTS	STEPEVENTS	Specifies when IBM Workload Scheduler for z/OS creates events for ending job-steps.
	PRINTEVENTS	Specifies if IBM Workload Scheduler for z/OS creates events for print tasks (type 4).
JTOPTS	PRTCOMPLETE	Specifies when IBM Workload Scheduler for z/OS sets print operations to complete.
	JTLOGS	
AUTHDEF	LISTLOGGING	Specifies how much data RACF stores for accesses to IBM Workload Scheduler for z/OS data.

Generating extended-auditing information (DB log data)

These parameters determine the amount of auditable information that IBM Workload Scheduler for z/OS produces.

The information is written to the extended-auditing data log and can be copied at daily planning to the tracklog data set (EQQDBOUT).

Table 11. Extended auditing-related parameters

Statement	Keywords	Description
AUDIT		Record access to IBM Workload Scheduler for z/OS data.
JTOPTS	JTLOGS	Specifies the number of auditing logs that IBM Workload Scheduler for z/OS must open when it is started.

Determining the success or failure of a job

These parameters specify how IBM Workload Scheduler for z/OS determines the next status of an operation when the job or started-task ends.

When the failing job has been obtained by restarting an operation at Step or Job level (see Restart and Cleanup function) and the failure is determined by the EQQCLEAN step ending with RC>=8 (and causing all subsequent steps to FLUSH), then the operation status is always set to Error, overriding any completion checking logic implemented. For jobs using non-centralized scripts and running on fault-tolerant workstations, see the RCCONDSUC keyword of the JOBREC statement described in the *Scheduling End-to-end with Fault Tolerance Capabilities* manual.

Table 12. Completion-checking-related parameters

Statement	Keywords	Description
EWTROPTS	RETCODE	Create job-end (3P) event with highest or last return code
	STEPEVENTS	Specifies when IBM Workload Scheduler for z/OS creates events for ending job-steps
JCCOPTS		Job completion checker actions
NOERROR	LIST	Error codes that are not errors
JTOPTS	NOERROR	Error codes that are not errors
	HIGHRC	Highest return code that is not an error
	ERRRES	Reset operation status to A (arriving) for these error codes

These job options in operation details override statement values:
- ERROR TRACKING
- HIGHEST RETURNCODE

IBM Workload Scheduler for z/OS processes the options in this order when a job or started task ends:

1. EWTROPTS RETCODE - create job-end event.
2. JCC - the event is passed to JCC if it is active. The JCC can set a new value for the return code. After JCC processing, the event passes to the controller.

   The event reaches the event queue at the controller.
3. Return code 0 - Operation status set to C. Or continue checking.
4. ERROR TRACKING - If operation details specify no error tracking, the operation status is set to C. Or continue checking.
5. NOERROR - If the return code matches a NOERROR entry, the operation status is set to C. Or continue checking.

   IBM Workload Scheduler for z/OS checks all NOERROR statements and the NOERROR keyword of JTOPTS for a matching entry.
6. HIGHRC - If the return code is less than or equal to HIGHRC, the operation status is set to C. Or continue checking.

   IBM Workload Scheduler for z/OS first uses the HIGHRC value in the operation details. If blank, JTOPTS HIGHRC is used.
7. ERRRES - If the return code matches an ERRRES entry, the operation status is set to A.

If no match has occurred, the operation status is set to E. Recovery processing can then occur.

Recovery

IBM Workload Scheduler for z/OS can perform recovery actions for job and started-task failures and for system failures. You can use these recovery functions in IBM Workload Scheduler for z/OS:
- Restart and cleanup
- Automatic job recovery
- Workload restart.

For jobs using non-centralized scripts and running on fault-tolerant workstations, see the RCCONDSUC keyword of the JOBREC statement in the *Scheduling End-to-end with Fault Tolerance Capabilities* manual.

Restart and cleanup

Table 13. Restart and cleanup-related parameters

Statement	Keywords	Description
BATCHOPT	RCLEANUP	Enables maintenance cleanup of local data store
OPCOPTS	RECOVERY	No automatic recovery action will be performed until cleanup actions are complete or discarded (when cleanup type is Immediate)
	RCLEANUP	Starts restart and cleanup tasks
FLOPTS	SNADEST	Specifies the table of tracker and data store destinations used to locate the data store used by the job log when an SNA connection is being used
	XCFDEST	Specifies the table of tracker and data store destinations used to locate the data store used by the job log when an XCF connection is being used
RCLOPTS	CLNJOBPX	Specifies the job name prefix to be used for stand-alone cleanup
	CLNJOBCARD	Specifies the job account information used while creating stand alone cleanup jobs.
	DDALWAYS	Lists the DD names that make step always re-executable
	DDNEVER	Lists the DD names that make step never re-executable
	DDNOREST	Lists the DD names that make step not restartable
	DDPRMEM	Contains the name of the PDS member of the parameter library containing the list of protected DD names
	DDPROT	Lists the DD names that identify protected data sets
	DSNPRMEM	Contains the name of the PDS member of the parameter library containing the list of protected data set names
	DSNPROT	Lists the protected data set names
	DSTCLASS	Specifies a JES class when JCC is used
	DSTDEST	Specifies the destination to be added in the JCL to create a sysout copy for the data store
	DSTRMM	RMM is active and cleanup will use the RMM API
	STEPRESCHK	Specifies the possibility to select a step restart range overriding the product logic checks

Job log retrieval

Table 14. Data store job log retrieval-related parameters

Statement	Keywords	Description
OPCOPTS	RCLEANUP	Activates the FL task on the controller to connect to the data store
FLOPTS	SNADEST	Specifies the table of tracker and data store destinations used to locate the data store used by the job log when an SNA connection is used
	XCFDEST	Specifies the table of tracker and data store destinations used to locate the data store used by the job log when an XCF connection is used
	CTLLUNAM	Specifies SNA values to be used for SNA connection to the data store
	DSTGROUP, CTLMEM	Specifies XCF values to be used for XCF connection to the data store

Automatic job recovery

When an operation ends in error, IBM Workload Scheduler for z/OS can perform recovery actions automatically or on request from the ended-in-error list in the MCP dialog. Recovery waits for cleanup action, if needed.

Table 15. Automatic-job-recovery-related parameters

Statement	Keywords	Description
OPCOPTS	RECOVERY	Determines if JCL is checked for RECOVER statements when an operation ends in error
	RCLEANUP	IBM Workload Scheduler for z/OS performs cleanup before recovery starts if clean type is immediate
AROPTS		Specifies recovery options
EWTROPTS	STEPEVENTS	Specifies when IBM Workload Scheduler for z/OS creates events for ending job-steps
	RETCODE	Highest or last return code
JTOPTS	ERRRES	Operation status reset to A for these error codes. Recovery is not performed.
	HIGHRC	Perform recovery only if the return code is greater than HIGHRC
	NOERROR	Error codes that are not errors. Recovery is not performed.
NOERROR		Error codes that are not errors. Recovery is not performed.

These job options in operation details override statement values:
- ERROR TRACKING
- HIGHEST RETURNCODE
- RESTART AND CLEANUP

Workload restart

Table 16. Workload-restart-related parameters

Statement	Keywords	Description
JTOPTS	WSFAILURE	System-level actions for workstation failure
	WSOFFLINE	System-level actions for workstation offline
	OPRESTARTDEFAULT	Action if restartable field in operation details is blank
	OPREROUTEDEFAULT	Action if reroutable field in operation details is blank
	OFFDELAY	Elapsed time before offline actions are taken

Table 16. Workload-restart-related parameters (continued)

Statement	Keywords	Description
ROUTOPTS	PULSE	The time between handshakes for the controller and the trackers. If a tracker does not respond to two successive handshake requests, the controller forces the destination offline.

These job options in operation details override statement values:
- RESTARTABLE
- REROUTABLE

Performance

These statements and parameters can affect the performance of IBM Workload Scheduler for z/OS.

Table 17. Performance-related parameters

Statement	Keywords	Description
AUDIT		Specifies how much audit information is produced
ALERTS	LATEOPER	Use only when deadlines are accurate
AUTHDEF	SUBRESOURCES	Specifies subresources that you want to check
EWTROPTS	STEPEVENTS	Specifies when IBM Workload Scheduler for z/OS creates events for ending job-steps
	PRINTEVENTS	Specifies if IBM Workload Scheduler for z/OS creates events for print tasks (type 4)
	HOLDJOB	Specifies if IBM Workload Scheduler for z/OS holds and releases jobs on the JES queue
	EWWAIT	Specifies the time between reads of a submit/release data set
JTOPTS	BACKUP	Specifies if a CP backup is performed automatically and how many records are written to the JT log between backups
	EVELIM	Specifies how often statistics messages related to the STATMSG keyword are issued
	MAXJSFILE	Specifies if a JS backup is performed automatically and how large the JCL repository file grows before IBM Workload Scheduler for z/OS performs a backup
	QUEUELEN	Specifies the number of operations IBM Workload Scheduler for z/OS starts when the workstation analyzer subtask gets the CP lock
	STATIM	Specifies when statistics messages are issued
	STATMSG	Determines if IBM Workload Scheduler for z/OS issues performance statistics for the current plan, event manager, general service, and WSA task
	TRACK	Determines which jobs IBM Workload Scheduler for z/OS tracks
OPCOPTS	RCLEANUP	Specifies if the restart and cleanup function is active. Performance is affected when the user chooses to archive the user sysouts for the majority of the operations.
	VARSUB	Determines which jobs are scanned for JCL variables and directives
TOPOLOGY	LOGLINES	Specifies the maximum number of lines that the Job Log Retriever returns for a single Job Log.

The amount of data written to the job-tracking log affects how often IBM Workload Scheduler for z/OS performs a current plan backup. Keep this in mind when specifying a value for the BACKUP keyword of JTOPTS.

Reporting

These statements and parameters affect the reports that are produced by daily planning batch jobs.

Table 18. Reporting-related parameters

Statement	Keywords	Description
BATCHOPT	DATEFORM	Date format in reports
	DPROUT	The ddname of the file that reports are written to
	HDRS	Character strings used as report headers
	PAGESIZE	Number of lines per page
	PLANHOUR	Start of a plan period for reporting purposes
	PREVRES	Previous period results (the 24 hours before PLANHOUR)
JTOPTS	PLANSTART	Start of a plan period for reporting purposes
RESOURCE	FILTER	Specifies which special resources should be reported on

You can also specify in a workstation description the ddname of a file that daily planning writes reports to for that workstation. This value overrides DPROUT only for reports for the workstation.

You select which report types IBM Workload Scheduler for z/OS produces when you run a daily planning job.

RODM monitoring

IBM Workload Scheduler for z/OS support for RODM lets you use established resource monitoring. Through subscriptions to RODM, you can monitor the status of real resources used by IBM Workload Scheduler for z/OS operations.

Table 19. RODM-related parameters

Statement	Keywords	Description
OPCOPTS	RODMTASK	Starts the RODM subtask
	RODMPARM	Specifies where RODMOPTS statements are stored
	EWTRTASK	Starts an event writer to collect resource events
RODMOPTS		Specifies subscription parameters
ROUTOPTS		Specifies routes to tracker destinations

You specify RODMOPTS statements only for the controller. A separate RODMOPTS is required for each subscription. You specify RODMTASK(YES) for an IBM Workload Scheduler for z/OS address space that communicates with RODM, which must be started on the same z/OS image as the RODM subsystem. An event writer must be started in the same address space.

If communication with RODM is through a tracker, you specify the tracker destination on RODMOPTS. The destination must be defined on ROUTOPTS.

Output processing

These statements and parameters determine how IBM Workload Scheduler for z/OS processes print output.

Table 20. Output-related parameters

Statement	Keywords	Description
EWTROPTS	PRINTEVENTS	Specifies if IBM Workload Scheduler for z/OS creates events for print tasks (type 4)
JCCOPTS	CHKCLASS	SYSOUT classes that IBM Workload Scheduler for z/OS checks
	JCWAIT	Time JCC waits before rechecking the SYSOUT queue for a job
	MAXDELAY	Maximum time the JCC tries to find SYSOUT
	SYSOUTDISP	Specifies if SYSOUT is held, deleted, or requeued after processing
	UMAXLINE	Specifies how many lines to check in user SYSOUT
	USYSOUT	Specifies if user SYSOUT is scanned
JTOPTS	OUTPUTNODE	Specifies which NJE node that SYSOUT is spooled to is used to create JES2 job-termination events

End-to-end scheduling with fault tolerance capabilities

These statements and parameters specify network configuration and job definitions in the end-to-end with fault tolerance capabilities environment.

Network configuration

Table 21. Network configuration-related parameters

Statement	Keywords	Description
CPUREC	CPUNAME	Specifies the name of the IBM Workload Scheduler workstation.
	CPUOS	Specifies the host CPU operating system related to the IBM Workload Scheduler workstation.
	CPUNODE	Specifies the node name or the IP address of the CPU.
	CPUTCPIP	Specifies the TCP port number of NETMAN on the current CPU.
	CPUDOMAIN	Specifies the name of the IBM Workload Scheduler domain of the CPU.
	CPUHOST	Specifies the name of the host CPU of the agent.
	CPUACCESS	Specifies the name of the access method.
	CPUTYPE	Specifies the CPU type.
	CPUAUTOLNK	Specifies the autolink between the agent and the domain manager.
	CPUFULLSTAT	Specifies that the link from the domain manager operates in Full Status mode.
	CPURESDEP	Specifies that the agent's Production Control process operates in Resolve All Dependencies mode.
	CPUSERVER	Identifies a server (Mailman) process on the domain manager that sends messages to the agent.
	CPULIMIT	Specifies the number of jobs that can run at the same time in a CPU.
	CPUTZ	Specifies the local timezone of the fault-tolerant workstation.
	CPUUSER	Specifies the default user for the workstation.
	SSLLEVEL	Specifies if the workstation uses SSL authentication when it connects with its domain manager.
	SSLPORT	Defines the port used to listen for incoming SSL connections.
	FIREWALL	Specifies if the communication between a workstation and its domain manager must crosses a firewall.
SERVOPTS	TPLGYPRM	Defines a member in the file identified by the EQQPARM DD statement in the server startup job. The member contains the fault-tolerant end-to-end options defined by the TOPOLOGY statement. It is used to activate the end-to-end scheduling with fault tolerance capabilities in the server.
	PROTOCOL	Identifies the types of communication used by the server.
	SUBSYS	Identifies the controller for which this server is started.

Table 21. Network configuration-related parameters (continued)

Statement	Keywords	Description
TOPOLOGY	BINDIR	Specifies the base file system directory where binaries, catalogs, and other files are installed and shared among subsystems.
	CODEPAGE	Specifies the name of the host codepage.
	HOSTNAME	Specifies the hostname or the IP address that will be used by the server in the end-to-end with fault tolerance capabilities environment.
	LOCALPSW	Specifies if the user ID and password to be used for Windows workstations are to be found locally, when missing from the Symphony file.
	LOGLINES	Specifies the maximum number of lines that the job log retriever returns for a single job log.
	PLANAUDITLEVEL	Enables or disables plan auditing for distributed agents.
	PORTNUMBER	Defines the TCP/IP port number used by the server to communicate with the distributed agents.
	SSLLEVEL	Specifies if the server uses SSL authentication.
	SSLPORT	Defines the port used to listen for incoming SSL connections on the server.
	TCPIPJOBNAME	Specifies the TCP/IP started-task name used by the server.
	TIMEZONE	Local time zone in the z/OS system where the controller runs.
	TPLGYMEM	Specifies the PARMLIB member where the domain and CPU definition are.
	TRCDAYS	Specifies the number of days the trace files are kept before being deleted.
	USRMEM	Specifies the PARMLIB member where the user definitions are.
	WRKDIR	Specifies the location of the files of a subsystem.
USRREC	USRCPU	Identifies the workstation on which the user can launch jobs. It is valid only on Windows workstations.
	USRNAM	Specifies the user name. It is valid only on Windows workstations.
	USRPWD	Specifies the user password. It is valid only on Windows workstations.

Job definitions

Table 22. Job definition-related parameters

Statement	Keywords	Description
JOBREC	JOBSCR	Specifies the name of the shell script or executable file to run
	JOBCMD	Specifies the name of the shell command to run
	JOBUSR	Specifies the name of the user submitting the script or command
	INTRACTV	Specifies that a Windows job runs interactively on the Windows desktop
	RCCONDSUC	Specifies a Boolean expression which determines the return code (RC) required to consider a job successful.

Table 22. Job definition-related parameters (continued)

Statement	Keywords	Description
RECOVERY	OPTION	Specifies the action that IBM Workload Scheduler for z/OS must take when a job ends unexpectedly
	MESSAGE	Specifies the text of a recovery prompt
	JOBCMD	Specifies the name of the shell command to run if the job abends
	JOBSCR	Specifies the name of the shell script or executable file to be run if the job abends
	JOBUSR	Specifies the name of the user submitting the recovery job action
	JOBWS	Specifies the name of the workstation
	INTRACTV	Specifies that the recovery job runs interactively on a Windows desktop
	RCCONDSUC	Specifies a Boolean expression which determines the return code required to consider a recovery job successful.
VARSUB	TABLES	Identifies the variable tables that must be searched, and the search order
	PREFIX	Specifies a non-alphanumeric character that precedes a variable
	BACKPREF	Specifies a non-alphanumeric character that delimits a variable to form simple and compound variables
	VARFAIL	Specifies whether IBM Workload Scheduler for z/OS is to issue an error message when a variable substitution error occurs
	TRUNCATE	Specifies if keywords are to be truncated

Regional settings

Missing or incorrect code page and time zone settings might cause unexpected results, such as garbage in retrieved job logs or incorrect job run times. The following sections provide a simple checklist to prevent this kind of problem.

Code page

At host side, you can set the code page by specifying TOPOLOGY(CODEPAGE()) in the server and batch initialization statements. The input translator and the fault tolerant agent (FTA) use the specified value to convert received data, from UTF-8 format to EBCIDIC format and conversely.

At distributed side, to verify the active code page you can use operating system specific commands, for example the **chcp** command for Windows and **locale** for UNIX. Moreover:

- Verify that the *TWS_TISDIR* environment variable is set to the name of the IBM Workload Scheduler home directory.
- To handle jobs containing some national characters on a Windows workstation, add the `chcp` *`code_page`* command to the *`TWS_home`*`\jobmanrc` file, where *`code_page`* is the code page including those national characters.

Time zone

In an end-to-end with fault tolerance capabilities environment, the work is scheduled in terms of controller local time.

When adding time-dependent jobs to the Symphony file, the scheduler converts that time to the local time of the FTA. The conversion is first from controller local time to GMT, then from GMT to FTA local time. The conversion succeeds only if the following conditions occur:

- At host side:

- You correctly set the local and GMT clocks.
- The USS `$BINDIR/zoneinfo` directory contains correct time zone definitions.
- For each workstation referenced through the TPLGYMEM statement, server and batch initialization statements contain a CPUREC(CPUTZ()) value matching the time zone value of the operating system where the agent runs.

- At distributed side, you correctly set clock and time zone values for the operating system hosting each agent.

WLM integration

This statement and these keywords determine how IBM Workload Scheduler for z/OS integrates with Workload Manager (WLM) to run operations.

Table 23. WLM integration-related parameters.

Statement	Keywords	Description
OPCOPTS	WLM	Provides information about the WLM service class integration function (class name, policy name).
	SECHECK	Specifies if and how integration with the WLM scheduling environment is to be activated.
	SERESETJS	Specifies if the JCL must be replaced in the JS file, when the WLM scheduling environment becomes available; in this way, any variables contained in the JCL are updated with the most up-to-date values.
	JESPLEX	Specifies the list of systems comprising the JESplex to which the tracker belongs.
	SYSPLEXID	Specifies the number identifying the sysplex to which the tracker belongs.
	SUPPRESSENF	Specifies if activation of the ENF 57 and ENF 41 listener exits is to be suppressed.

External monitoring

These statements and these keywords specify the configuration options for IBM Workload Scheduler for z/OS to work with Tivoli Business Systems Manager and IBM Tivoli Monitoring through the Tivoli Enterprise Portal component.

Table 24. Parameters related to integration with external monitors

Statement	Keywords	Description
ALERTS	MONALERT	Defines the conditions under which a generic alert will be sent to IBM Tivoli Monitoring.
	MONOPER	This parameter determines whether the error conditions specified by the MONALERT keyword will be in effect for monitored jobs only or for all jobs. It is used with IBM Tivoli Monitoring.
OPCOPTS	EXTMON	Specifies if Integration with Tivoli Business Systems Manager is enabled.
	CODEPAGE	Specifies the host code page to be used for the data collected by the monitoring task

Table 24. Parameters related to integration with external monitors (continued)

Statement	Keywords	Description
MONOPTS	MONHOSTNAME	Identifies the host name or IP address of the remote monitoring application. This parameter is used for the integration with the IBM Tivoli Monitoring product.
	MONPORT	Specifies the port number of the remote monitoring application. It is used for the integration with IBM Tivoli Monitoring.
	LOCHOSTNAME	Specifies the local host name or IP address that will be used to communicate with IBM Tivoli Monitoring.
	LOCPORT	Specifies the local port number used by the controller to communicate with IBM Tivoli Monitoring.
	BULKDISC	Defines if and how the bulk discovery is to be performed. This keyword is used for the integration with IBM Tivoli Monitoring.
	CONNTIMEOUT	Defines the connection establishment timeout to be used when communicating with IBM Tivoli Monitoring.
MONPOL	OPERATION	Specifies the types of operations that will be automatically selected for monitoring by IBM Tivoli Monitoring and Tivoli Business Systems Manager.

Chapter 3. Implementing security

This chapter explains how you use IBM Workload Scheduler for z/OS security features to protect IBM Workload Scheduler for z/OS functions and data.

Before you use IBM Workload Scheduler for z/OS security features, you must define and enable the security environment. For details, see *Planning and Installation*.

Planning security implementation

Consider the tasks in this section when determining your security requirements.

About this task

Table 25. Security planning

Task	**Reference**
Topic	
How IBM Workload Scheduler for z/OS verifies access.	"How IBM Workload Scheduler for z/OS verifies access authority" on page 186
Determine which user IDs require access to IBM Workload Scheduler for z/OS.	"Identifying users" on page 187
Establish naming conventions for IBM Workload Scheduler for z/OS resources.	"Establishing naming conventions for IBM Workload Scheduler for z/OS resources" on page 188
Group RACF users and resources.	"Grouping RACF users and resources" on page 188
Review general security considerations.	"General security considerations" on page 189
Determine if you use a centralized or decentralized strategy. Your strategy determines to some extent the levels of protection you need: • Subsystem - Who can access IBM Workload Scheduler for z/OS. • Fixed resources - Which functions can a user access, for example, the AD dialog, the MCP dialog, or the REFRESH function. • Subresources - What data can a user access within a function. For example, you might permit a user access to the AD dialog but only to certain applications.	"Examples of security strategies" on page 210
	"Controlling access to the IBM Workload Scheduler for z/OS subsystem" on page 190
	"Controlling access to IBM Workload Scheduler for z/OS fixed resources" on page 190
	"Controlling access to IBM Workload Scheduler for z/OS subresources" on page 191
Review API security and access requirements if you use the API from your own TP or through the GUI.	"Controlling access to IBM Workload Scheduler for z/OS from APPC" on page 193
Review security and access requirements if you use Dynamic Workload Console.	"Controlling access to IBM Workload Scheduler for z/OS using Dynamic Workload Console" on page 195

Table 25. Security planning (continued)

Task	Reference
Topic	
Review access requirements for IBM Workload Scheduler for z/OS TSO commands.	"Controlling access through TSO commands" on page 198

When you have determined your security requirements, implement security access:

Table 26. Security implementation

Task	Reference
Topic	
Verify that the environment is set up. Ensure that you have: • Defined the user ID of the IBM Workload Scheduler for z/OS in the STARTED class. • Defined the IBM Workload Scheduler for z/OS subsystem name as a resource in the APPL class. • Used the resource class reserved for IBM Workload Scheduler for z/OS, IBMOPC.	Refer to *IBM Workload Scheduler for z/OS Planning and Installation*
Specify access to the subsystem.	"Controlling access to the IBM Workload Scheduler for z/OS subsystem" on page 190
Specify fixed resources.	"Controlling access to IBM Workload Scheduler for z/OS fixed resources" on page 190
Specify subresources.	"Controlling access to IBM Workload Scheduler for z/OS subresources" on page 191
Implement security access through the IBM Workload Scheduler for z/OS API, if you use this function.	"Controlling access to IBM Workload Scheduler for z/OS from APPC" on page 193
Implement security access through the IBM Workload Scheduler for z/OS server, if you use this function.	"Controlling access to IBM Workload Scheduler for z/OS from APPC" on page 193
Specify subresources on the AUTHDEF statement.	"AUTHDEF" on page 17
Specify resource names on the AUDIT statement, if you need audit information.	"AUDIT" on page 14

How IBM Workload Scheduler for z/OS verifies access authority

To verify access authority, IBM Workload Scheduler for z/OS uses the RACROUTE macro. This macro has a general-purpose interface to a security product through the system authorization facility (SAF). The security product can be RACF or any other product that works with SAF. In this chapter, RACF commands show how IBM Workload Scheduler for z/OS interfaces with a security product.

To verify a user's authority, IBM Workload Scheduler for z/OS uses the RACROUTE macro to invoke the SAF z/OS router. This conditionally directs control to RACF, if present.

The RACROUTE options that IBM Workload Scheduler for z/OS uses invoke these RACF functions:

RACINIT
Provides RACF user identification and verification when IBM Workload Scheduler for z/OS services are requested. (IBM Workload Scheduler for z/OS does not have its own logon panel or user IDs.)

RACLIST
Builds in-storage profiles for resources defined by RACF, which improve performance for resource authorization checking.

Note: Some security products do not support this function. If you are using such a product, RACLIST is effectively a *no* operation.

RACHECK
Provides authorization checking when you request access to a RACF-protected resource, for example, when you access:
- Data (such as the current plan)
- A function (such as REFRESH)

For more information about resources that you can protect, see "Functions and data that you can protect" on page 199.

FRACHECK
Provides authorization checking in the IBM Workload Scheduler for z/OS subsystem.

Note: Security products that do not support RACLIST convert FRACHECK requests to the corresponding RACHECK request. This could have a severe impact on the performance of some IBM Workload Scheduler for z/OS dialog functions.

Identifying users

RACF controls the interaction between users and resources. You define resources and the level of access allowed by users to these resources in RACF profiles. A user is an alphanumeric user ID that RACF associates with the user.

IBM Workload Scheduler for z/OS needs access to non-IBM Workload Scheduler for z/OS resources for the work it schedules. The user ID associated with IBM Workload Scheduler for z/OS can be obtained from:
- The IBM Workload Scheduler for z/OS address space that accesses data sets used by the work it schedules, and that submits work and issues JES and MVS commands.
- The *user*= parameter on the JOB card of a batch job.
- The IBM Workload Scheduler for z/OS job-submit exit, EQQUX001, which is called when IBM Workload Scheduler for z/OS is about to submit a job or start a started task, and which can pass back a user ID.
- The USRREC statement, which specifies the name and password of the user on a supported Windows workstation.
- The LOCALPSW statement, which specifies whether the name and password of a user on a Windows workstation is defined either on z/OS using the USRREC statement (LOCALPSW set to NO) or on the Windows workstation using a local

file (LOCALPSW set to YES). If you set LOCALPSW to YES, the scheduler looks for the USRREC statement first, then for the local file.
- The USERMAP keyword of the SERVOPTS statement.

User IDs that access IBM Workload Scheduler for z/OS resources can be:
- A TSO user ID that accesses the IBM Workload Scheduler for z/OS dialogs, submits batch jobs that access IBM Workload Scheduler for z/OS resources, and issues IBM Workload Scheduler for z/OS TSO commands.
- An IBM Workload Scheduler for z/OS address space, which must be permitted access to IBM Workload Scheduler for z/OS resources.
- Other started-task address spaces that pass requests to an IBM Workload Scheduler for z/OS address space.
- A user ID supplied by a transaction program (TP) that uses the IBM Workload Scheduler for z/OS API to communicate with the controller.
- A user ID defined by the USERMAP keyword of the SERVOPTS statement to work with Dynamic Workload Console.

Establishing naming conventions for IBM Workload Scheduler for z/OS resources

The use of consistent naming conventions lets you group users and helps to reduce the number of RACF resource names you need. Name standards are particularly important if you restrict access to IBM Workload Scheduler for z/OS data by specifying subresources. Consider these resources when establishing naming conventions:
- Application ID
- Group definition ID
- Owner ID
- Authority Group ID
- Calendar ID
- Period name
- Operation name
- Workstation name
- JCL-variable table ID
- Special-resource name

Also consider which resources you use to restrict access. For example, you can protect application descriptions through the application ID, the owner ID, and the authority group ID.

Note: Some IBM Workload Scheduler for z/OS fields that can be used for security verification permit characters not acceptable to RACF. For example, the characters for semicolon, comma, and blank cannot be specified in a RACF resource name regardless of the FIRST and OTHER specifications in the ICHERCDE MACRO, but are acceptable as part of an IBM Workload Scheduler for z/OS owner ID.

Grouping RACF users and resources

It is strongly recommended that you do not grant access to individual users, but try to group users into different categories. You can then define a RACF user group for each category of users.

With RACF user groups, you need not change access lists of different profiles as often. When you must make a change, you add or remove a user ID in the group, or move the user ID to another group.

These categories are used at many IBM Workload Scheduler for z/OS installations:
- Schedulers
- Workstation operators
- IBM Workload Scheduler for z/OS shift leaders
- Machine room operators
- IBM Workload Scheduler for z/OS system support

Also consider using generic profiles when specifying RACF resource names. Resources protected by generic profiles have similar names and identical security requirements.

General security considerations

IBM Workload Scheduler for z/OS submits jobs for users and starts started tasks. Users communicate with IBM Workload Scheduler for z/OS through ISPF dialogs running under TSO or through batch jobs. These dialogs and batch jobs use the IBM Workload Scheduler for z/OS subsystem.

Some users might need to allocate, delete, or reorganize IBM Workload Scheduler for z/OS data sets. RACF and IBM Workload Scheduler for z/OS facilities let you give individual users the level of access they need while protecting your data from accidental or malicious damage.

IBM Workload Scheduler for z/OS needs update access to catalogs and alter access to data sets for all work that it tracks, which uses the restart and cleanup function. But if you permit IBM Workload Scheduler for z/OS access to all your systems, a user might gain unauthorized access through IBM Workload Scheduler for z/OS, because any job submitted by IBM Workload Scheduler for z/OS can access the data. So if you use RACF 1.9 or later, consider *surrogate job submission* to authorize jobs submitted by IBM Workload Scheduler for z/OS. By specifying IBM Workload Scheduler for z/OS as a surrogate user for each of your systems, you can avoid violations from other users. For more information, refer to *Planning and Installation* and *RACF Administrator's Guide*

If you use the IBM Workload Scheduler for z/OS hot standby facilities, consider the security environment on any potential standby system. If the standby is invoked, you must access IBM Workload Scheduler for z/OS data sets, dialogs, resources, and subresources from the standby system.

If you use the workload restart function, ensure that rerouted work can access the required resources on the system where the work is performed. IBM Workload Scheduler for z/OS work that is submitted at a particular destination has the authority of IBM Workload Scheduler for z/OS at that destination or, if the EQQUX001 exit is used, the authority of the submitting user.

You can track access to IBM Workload Scheduler for z/OS resources by specifying parameters on the AUDIT initialization statement. When a user accesses a nominated resource, a record is written to the current job-tracking-log data set. The AUDIT statement is described in "AUDIT" on page 14.

Controlling access to IBM Workload Scheduler for z/OS

IBM Workload Scheduler for z/OS security involves three levels of protection:
- The IBM Workload Scheduler for z/OS subsystem determines if a user can establish communication with IBM Workload Scheduler for z/OS.

- Fixed resources protect functions in IBM Workload Scheduler for z/OS; for example, modifying the current plan or requesting a backup of a resource data set.
- Subresources protect IBM Workload Scheduler for z/OS data.

The level of access given to a user at one level determines the default access that the user has at remaining levels. For example, if a user has update access to the IBM Workload Scheduler for z/OS subsystem, then that user has, by default, update access to all fixed resources. A user's access to fixed resources in turn determines the default access to subresources. The default value is used when a resource has not been specifically protected.

If you use the IBM Workload Scheduler for z/OS API, you can also use the security functions of APPC/MVS to protect access to the controller. See "Controlling access to IBM Workload Scheduler for z/OS from APPC" on page 193, for more information.

Carefully review your security requirements, and specify the levels of protection that you require. Do not specify extra levels of protection if they are not needed.

Controlling access to the IBM Workload Scheduler for z/OS subsystem

Specify the name of your host IBM Workload Scheduler for z/OS subsystem as a resource in the APPL class with default access NONE. You can effectively control access to IBM Workload Scheduler for z/OS dialog functions by allowing or denying users access to the subsystem resource. If the user runs any batch jobs that use the subsystem, these batch jobs are similarly restricted. This restriction does not apply to the EQQEVPGM or TSO commands. For example, to permit only user group OPCUGRP access to subsystem OPCC, and to grant update authority, you enter:

```
RDEFINE APPL OPCC  UACC(NONE)
PERMIT OPCC ID(OPCUGRP) ACCESS(UPDATE) CLASS(APPL)
```

When a dialog user tries to access a subsystem (for example, OPCC), RACF looks in the APPL class to see if this resource is defined. If the resource is defined and the access authority is read or update, the user can continue. If the resource is not defined, the dialog user has update access to all IBM Workload Scheduler for z/OS fixed resources.

Any TSO user with either read or update access to the subsystem resource in the RACF APPL class can enter the IBM Workload Scheduler for z/OS dialogs. By default, the user has the same access (read or update) to IBM Workload Scheduler for z/OS fixed resources.

Controlling access to IBM Workload Scheduler for z/OS fixed resources

You will probably want to place more restrictions on access to resources. For example, one user might need update access to the JCL and job library file but need only read access to calendar data. You achieve this level of control by specifying IBM Workload Scheduler for z/OS fixed resources in a general resource class used by IBM Workload Scheduler for z/OS. RACF provides an IBM reserved resource class, IBMOPC. For a checklist about using RACF classes, refer to CLASS parameter description in "AUTHDEF" on page 17.

Note: Preventing a user from accessing a data set might not prevent the user from updating the data within the data set. When using IBM Workload Scheduler for z/OS dialogs, users access IBM Workload Scheduler for z/OS data through the IBM Workload Scheduler for z/OS subsystem with the subsystem level of access.

Table 29 on page 200 shows the fixed resources that you can protect.

When you define the resource names of the IBM Workload Scheduler for z/OS fixed resources you want to protect, you grant a level of access to users. These access levels are meaningful:

ACCESS(NONE)
ACCESS(READ)
ACCESS(UPDATE)

ACCESS(ALTER) has no code support in IBM Workload Scheduler for z/OS for either fixed resources or subresources. ALTER gives the same level of access as UPDATE.

If you change a user's access level or remove the user's profile entirely, the change does not take effect until the user exits the IBM Workload Scheduler for z/OS dialog and tries to enter it again. Remember that the default access to IBM Workload Scheduler for z/OS fixed resources is determined by the user's level of access to the IBM Workload Scheduler for z/OS subsystem.

RACF does not check for a RACF class until that class is activated. You can activate a class by using the ACTIVATE parameter of the SETROPTS command.

Controlling access to IBM Workload Scheduler for z/OS subresources

You can restrict access to IBM Workload Scheduler for z/OS data by specifying subresources. This level of protection is useful if you want to permit different users access to a particular IBM Workload Scheduler for z/OS function, while allowing the users' access only to their own IBM Workload Scheduler for z/OS data. For example, a user might not need access to all applications, only to payroll applications.

If you do not specify subresources, access to IBM Workload Scheduler for z/OS data is determined by a user's access to fixed resources or, if fixed resources are not defined, by the user's access to the IBM Workload Scheduler for z/OS subsystem. To implement specific protection of IBM Workload Scheduler for z/OS data, you must:

- Give users access to the fixed resource that owns the subresource.
- Add the subresource to the list of SUBRESOURCES on the AUTHDEF statement.
- Define to RACF the RACF resource name that corresponds to the IBM Workload Scheduler for z/OS subresource name.
- Set up the required RACF profile for the resource.

Table 29 on page 200 shows the subresources that you can protect.

If you change a profile for a subresource while IBM Workload Scheduler for z/OS is active, the change does not take effect immediately. Three ways to effect the change are:

- Stop and restart IBM Workload Scheduler for z/OS.

- Use the modify command (F xxxx,P=GEN followed by F xxxx,S=GEN) to stop and restart the general service subtask.
- Select Service Functions (item 9 on the IBM Workload Scheduler for z/OS dialog main menu), and then select RACF Resources.

RACF does not check for a RACF class until that class is activated. You can activate a class by using the CLASSACT parameter of the SETROPTS command. To enable generic profile checking, define the class in the GENERIC parameter of SETROPTS.

IBM Workload Scheduler for z/OS subresources and RACF resources

About this task

The AUTHDEF statement uses the IBM Workload Scheduler for z/OS subresource name to activate RACF checking for an IBM Workload Scheduler for z/OS subresource. For example, if you want IBM Workload Scheduler for z/OS to verify authorization for application descriptions by checking the application name, you specify the value AD.ADNAME on the SUBRESOURCES keyword of the AUTHDEF statement. The resource name that RACF then checks consists of a 3-character code identifying the subresource, followed by a name specifying the particular data to be protected. For example, to protect application descriptions whose application name is PAYROLL, you define a RACF resource, ADA.PAYROLL, in the RACF resource class that is specified on the AUTHDEF statement.

Here is an example that shows how you could protect the JCL and job library file (JS). This example assumes that:

- The file is protected against update but can be read by any user.
- The owner ID is used to protect access. Table 29 on page 200 shows the other names that you can select to protect the JS fixed resource.
- User CASHIER has update access to data with owner PAYROLL but has only read access to other data.
- OPCCLASS is the RACF resource class used to protect IBM Workload Scheduler for z/OS resources. This name is specified on the AUTHDEF statement.
- The required resources are not defined.

To implement protection:

1. Define the fixed resource that owns the subresource and give universal read access to it:
   ```
   RDEFINE (OPCCLASS) JS UACC(READ)
   ```
2. Give user CASHIER update access to the JS fixed resource:
   ```
   PERMIT JS ID(CASHIER) ACCESS(UPDATE) CLASS(OPCCLASS)
   ```
3. Define a RACF resource, JSO.PAYROLL, to RACF and give universal read access to JSO.PAYROLL:
   ```
   RDEFINE (OPCCLASS) JSO.PAYROLL UACC(READ)
   ```
   JSO is the 3-character code that RACF uses for JS.OWNER.
4. Give user CASHIER update access to JSO.PAYROLL:
   ```
   PERMIT JSO.PAYROLL ID(CASHIER) ACCESS(UPDATE) CLASS(OPCCLASS)
   ```
5. Define a subresource JSO.* to RACF and give universal read access to this subresource:
   ```
   RDEFINE (OPCCLASS) JSO.* UACC(READ)
   ```

This rule prevents the user CASHIER from updating JCL in occurrences that do not have the PAYROLL owner ID.

6. Start checking for the JS.OWNER subresource by specifying JS.OWNER on the SUBRESOURCES keyword of the AUTHDEF statement.

A user's default access to IBM Workload Scheduler for z/OS subresources is determined by the user's access to IBM Workload Scheduler for z/OS fixed resources.

If you base your subresources on application names or owner IDs and these do not have consistent naming standards, you might need hundreds or even thousands of RACF profiles. This would make your subresources difficult to maintain. It would also slow IBM Workload Scheduler for z/OS processing, particularly at startup time when all profiles are read in. You can dramatically reduce the number of profiles you need by using consistent naming standards, or RACF generic profiles, or both.

Note:

1. If you define only fixed resources, a user who asks for a list of occurrences sees the names of *all* occurrences. If you define subresources, only the occurrences that the user has access to are listed. So two IBM Workload Scheduler for z/OS users asking for the same list on the same panel could see different lists.
2. If you use subresource protection, you can control the number of access violations that are logged for list requests through the LISTLOGGING keyword of AUTHDEF.
3. The check for a subresource authority does not depend on LISTLOGGING or the order of the subresources in the AUTHDEF statement. When more than one subresource is specified, a check for each one is issued. The check process stops at the first failure and no check is performed for the other subresources.

Controlling access to IBM Workload Scheduler for z/OS from APPC

Read the following information if you use any of the IBM Workload Scheduler for z/OS user interfaces to access IBM Workload Scheduler for z/OS using APPC. The interfaces include:

- The IBM Workload Scheduler for z/OS API, either from your own transaction program or through the GUI
- The IBM Workload Scheduler for z/OS server, from your own PIF programs or ISPF dialogs, or the GUI for Application Description.

Access to IBM Workload Scheduler for z/OS through the API or the server can be controlled in two distinct ways: through security provided by:

- APPC/MVS and RACF
- IBM Workload Scheduler for z/OS and RACF

APPC/MVS and RACF

The APPC/MVS component builds a security environment that is passed to the IBM Workload Scheduler for z/OS scheduler for the user ID that allocates the conversation. APPC/MVS requires that a conversation be either trusted or non-trusted. If a conversation is defined as trusted, then the security environment is assumed to have been verified and the allocation is accepted. If a conversation is non-trusted, then the security environment must be passed as part of the allocation.

Access to IBM Workload Scheduler for z/OS through PIF or the ISPF dialogs uses a trusted allocation.

Access to IBM Workload Scheduler for z/OS through the API, or GUI uses a non-trusted allocation. As an extra level of security, these accesses use a transaction program that must have a security type of PGM or SAME. For GUI, these definitions are made in the Communications Manager.

You must also ensure that a RACF user profile exists in the RACF database for the user ID that is passed in an allocate request.

You can use APPC/MVS security functions to control:
- Access to LUs
- Access for LU to LU communication
- Access to transaction programs
- Security within the network

IBM Workload Scheduler for z/OS recognizes the following transaction program names:

Name	**Supplied by**
EQQTRK	Trackers that communicate with the controller through APPC
EQQAPI	User programs (ATPs) that communicate with IBM Workload Scheduler for z/OS through the API
EQQDIA	The IBM Workload Scheduler for z/OS ISPF dialogs

Refer to *APPC Management* for detailed information about protecting your APPC/MVS environment. *ICSF/MVS Programmer's Guide* describes how you can protect information that crosses a network.

IBM Workload Scheduler for z/OS API and RACF

IBM Workload Scheduler for z/OS performs security checking at the controller for all transaction programs (TP) that use the API.

To establish a conversation, the outbound TP must supply a `user_id` and password, and optionally a profile that indicates the RACF user group. The `user_id` must have access to the IBM Workload Scheduler for z/OS subsystem resource, which is defined in the APPL class.

A user needs this access to fixed resources for API requests:

GET
CP read. SR read is also required to retrieve special resource information.

PUT
CP update. RL update is required to change the status of operations or issue ready list commands such as MH or NOP. EXEC update is required to use the EXEC command.

DEL
CP update.

CREATE
IBM Workload Scheduler for z/OS does not support security checking for CREATE requests because a request could be directed to more than one IBM Workload Scheduler for z/OS subsystem where security rules differ. You can prevent unauthorized use of CREATE requests through APPC/MVS security mechanisms by protecting the LU and the TP name.

If you protect IBM Workload Scheduler for z/OS data by specifying subresources, users must have the appropriate access to subresources. Table 29 on page 200 shows the subresources that you can specify for each fixed resource.

Controlling access to IBM Workload Scheduler for z/OS using Dynamic Workload Console

Read the following information if you are using Dynamic Workload Console to access IBM Workload Scheduler for z/OS using TCP/IP.

The z/OS connector performs a security check when a user tries to use Dynamic Workload Console of IBM Workload Scheduler for z/OS, by checking the user ID and password. The z/OS connector associates each user ID and password with an administrator.

IBM Workload Scheduler for z/OS resources are currently protected by RACF.

The Dynamic Workload Console user should have to enter only a single user ID and a password combination, and not provide two levels of security checking (at z/OS connector level and again at IBM Workload Scheduler for z/OS level).

The security model is based on z/OS connector security handling initial user verification, and, at the same time, obtains a valid corresponding RACF user ID. This makes it possible for the user to work with the security environment in z/OS.

z/OS security is based on a table that maps the administrator to a RACF user ID. When a z/OS connector connects to IBM Workload Scheduler for z/OS, the z/OS connector administrator is mapped to the corresponding user ID. Therefore, ensure that the administrator ID is associated with a RACF user ID in the USERMAP parameter of the SERVOPTS initialization statement. The RACF user ID does not need to have particular permissions to the IBM Workload Scheduler for z/OS resources. For details about how to map a user ID to a RACF user ID, see "Using a server initialization parameter to associate a RACF user ID" on page 197.

For any operations performed through Dynamic Workload Console, ensure that the Dynamic Workload Console user ID is associated with a corresponding RACF user ID. The RACF user ID must have the permissions required to access the IBM Workload Scheduler for z/OS resources.

IBM Workload Scheduler for z/OS server uses the RACF user ID to build the RACF environment to enable the user to access IBM Workload Scheduler for z/OS services.

You can obtain the RACF user ID in either of the following ways:

- Using the RACF Tivoli-supplied and predefined resource class TMEADMIN. For details, see "Creating the TMEADMIN class to associate a RACF user ID" on page 196.
- Using a server initialization parameter (SERVOPTS USERMAP) to define a member in the file identified by the EQQPARM DD statement in the server startup job. If the USERMAP parameter is specified for the SERVOPTS statement, the TMEADMIN RACF class is ignored. For details, see "Using a server initialization parameter to associate a RACF user ID" on page 197.

This example shows the authentication process performed by the z/OS connector when you connect as a Dynamic Workload Console user. Suppose that:

- The name of the host in which the z/OS connector runs is ROME1.
- The z/OS connector user is named ZCONN1.
- The Dynamic Workload Console user ID with which you connect to the z/OS connector is GRAPHUSR.

When GRAPHUSR connects to the z/OS connector, this user ID is authenticated on ROME1. Also, ZCONN1 is authenticated on the z/OS engine by providing the following credentials:

```
USER 'ZCONN1@domain' RACFUSER (TSOuser)
```

where *TSOuser* is the TSO user ID with which the IBM Workload Scheduler for z/OS dialogs are run.

When GRAPHUSR performs an operation, the z/OS connector uses these credentials, therefore it is required that both GRAPHUSR and ZCONN1 are associated with a RACF user ID. The RACF user ID associated with the z/OS connector user does not need to have particular permissions to the IBM Workload Scheduler for z/OS resources, while the RACF user ID associated with the console user needs the permissions to perform the required operations.

The following table shows the relationship between security products and the security selections.

Table 27. Relationship between security products and security selections

Security Product used	Solution	Prerequisite
Security Server (RACF)	TMEADMIN	None (TMEADMIN class provided in z/OS base)
Other SAF-compliant	TMEADMIN	Manually define TMEADMIN class (using EQQ9RFDE and EQQ9RF01 samples)
All security products	ID mapping table	

Creating the TMEADMIN class to associate a RACF user ID

About this task

1. Make sure your operating system has the Security Server feature.
2. Create the TMEADMIN class for mapping the administrator ID and host name to the RACF user ID.

   Note: If RACF is your security product and your operating system does not have the Security Server feature, you can use the supplied samples to create the following:

   - RACF TMEADMIN class EQQ9RFDE. Use the following macro, which you can access in the EQQ9RFDE member of SEQQSAMP library:

     ```
     TMEADMIN ICHERCDE CLASS=TMEADMIN,
                                   ID=129,
                                   MAXLNTH=246,
                                   FIRST=ALPHANUM,
                                   OTHER=ANY,
                                   POSIT= 26,
                                   OPER=NO,
                                   DFTUACC=NONE,
                                   DFTRETC=8,
                                   RACLIST=ALLOWED,
                                   GENLIST=ALLOWED
     ```

- RCAF Router Table EQQ9RF01. Use the following macro, which you can access in the EQQ9RF01 member of SEQQSAMP library:

  ```
  TAB18           ICHRFRTB  CLASS=TMEADMIN,ACTION=RACF
  ```

3. Using RCAF TMEADMIN class, map the administrator ID to the RACF user ID. The RACF user ID is associated with the administrator defined at the Tivoli workstation. Any administrative action is thereby traceable to the user issuing the request.
4. Define a profile in the Tivoli-supplied resource class TMEADMIN for each administrator who is able to access Dynamic Workload Console.

   Note: In the following tasks, which are for mapping the administrator to RACF user IDs, it is recommended that each administrator maps to a unique RACF user ID.
5. Activate the TMEADMIN class by typing the following command: SETROPTS CLASSACT (TMEADMIN).
6. In the TMEADMIN class, use the following string to define a unique RACF user ID for each Tivoli administrator who will perform Dynamic Workload Console operations: userid@hostname. For example, for a user with the identifier SCOT at the host pelican you would use SCOT@pelican.
7. Enter the following command to define a general resource profile in the TMEADMIN class to associate the administrator with a RACF user ID (in this example, SCOT):

   ```
   RDEFINE TMEADMIN SCOT@hostname APPLDATA('SCOT')
   ```

   Note: The string SCOT@hostname is not case sensitive.
8. Refresh the TMEADMIN class with the following command:

   ```
   SETROPTS RACLIST(TMEADMIN) REFRESH
   ```

   If you experience problems using special characters to define a profile in the TMEADMIN class, use the following command instead:

   ```
   SETROPTS GENERIC(TMEADMIN) REFRESH
   ```

Also, use the % sign instead of the special character. For example, for the Italian code page, the character @ (hex'B5') is not accepted by RACF. So, for SCOT@pelican, you should type SCOT%pelican.

When searching a list of TMEADMIN files for a match, RACF looks for the most similar generic profile.

Using a server initialization parameter to associate a RACF user ID

About this task

Define a member in the file identified by the EQQPARM DD statement in the server startup job. This member contains all the associations between a z/OS controller user and a RACF user ID.

1. Set the USERMAP parameter in the SERVOPTS server initialization parameter to define the user name, as follows:

   ```
   SERVOPTS
     SUBSYS(xxxx)
     USERMAP(USERS)
     PROTOCOL(E2E)
     PORTNUMBER(425)
   ```

2. Using the same approach as for the TMEADMIN class, check that the member USERS of the initialization parameter data set contain the following:

   ```
   USER 'SCOT@PELICAN' RACFUSER(SCOT) RACFGROUP(GROUP1)
   USER 'PAOLO@PELICAN' RACFUSER(FALSI) RACFGROUP(GROUP1)
   USER 'MOSSOTT@PELICAN2' RACFUSER(FMOSSOTT) RACFGROUP(GROUP1)
   ```

Permitting access to the controller through Dynamic Workload Console

If you use Dynamic Workload Console, you can control access to the controller through the security functions of both the z/OS connector and IBM Workload Scheduler for z/OS. Ensure that you consider both these environments when you update RACF.

User IDs involved with Dynamic Workload Console

The following example explains how the Tivoli administrator ID and the local user ID are related and used through Dynamic Workload Console:

- The name of the host on which you run Dynamic Workload Console is `Rome1`
- An administrator named `ADMN1` is defined on `Rome1`
- A local user `LOCUSR1` is defined on `Rome1`
- The administrator `ADMN1` is associated with local user `LOCUSR1`
- The controller identifies Dynamic Workload Console by means of the following user name:

  ```
  USER ADMN1-AT-ROME1-domain RACFUSER (TSOuser)
  ```

  where *`TSOuser`* is the TSO user ID with which the IBM Workload Scheduler for z/OS dialog runs.

The WebSphere Application Server user identity:

When the z/OS connector opens a connection with the IBM Workload Scheduler for z/OS server, the WebSphere® Application Server authenticates the communication by means of the WebSphere Application Server user identity.

Depending on your configuration, the server user identity can be one of the following:

- An automatically generated server identity that is not stored in a user repository (for example `SERVER:TIPCELL_TIPNODE_SERVER1`).
- A server identity that is stored in the repository.

In either case you need to associate the server user identity to a RACF ID. You can modify your configuration to use the administrator ID as the server user identity by editing the `changeSecurityproperty` WebSphere Application Server tool (*TWA_home*/wastools or *TWA_home*\wastools) where you set `UseRegistryServerId=true` and you specify the administrator user ID and password in the `ServerID` and `ServerPassword` keys.

Controlling access through TSO commands

You can use IBM Workload Scheduler for z/OS TSO commands to perform several functions.

Table 28 on page 199 shows the commands and the resources that they need access to.

Table 28. Access requirements for IBM Workload Scheduler for z/OS TSO commands

Command	Fixed resource	Subresource	Description
BACKUP	BKP		Initiate backup of an IBM Workload Scheduler for z/OS data set
BULKDISC	BUL		Initiate® bulk discovery for the monitoring agent
JSUACT	JSUB		Activate or deactivate job submission
OPINFO	CP		Update the user field of an operation
OPSTAT	RL	RL.WSNAME	Change the status of an operation
SRSTAT	SR	SR.SRNAME	Create or update a special resource
WSSTAT	RL	RL.WSSTAT	Change the status of a workstation

The authority of the requester is verified by the subsystem name identified in the command, if an AUTHDEF statement is defined for that subsystem. If the request is broadcast, each IBM Workload Scheduler for z/OS subsystem on the z/OS system that the command is directed to will attempt to verify the authority of the requester, before it generates an event. It could be rejected by one subsystem and accepted by another. A user must have update authority to the required resource to use TSO commands. *Managing the Workload* describes TSO commands in detail.

Functions and data that you can protect

You can use fixed resources and subresources to protect IBM Workload Scheduler for z/OS functions and data.

Fixed resources are always checked as part of the IBM Workload Scheduler for z/OS dialog. Subresources are checked only if they are defined in the “AUTHDEF” on page 17statement.

Table 29 on page 200 describes all fixed resources and subresources. Use the table to determine which resources you should define to RACF. You use Table 31 on page 206 to determine what access is required to the defined resources for each user.

Note: The subresource name and the RACF resource name are not the same. You specify the subresource name shown in column 2 on the SUBRESOURCES keyword of AUTHDEF to start subresource verification. The corresponding RACF resource name shown in column 3 must be defined in the general resource class used by IBM Workload Scheduler for z/OS, which is specified on the CLASS keyword of “AUTHDEF” on page 17.

Table 29. Protected fixed resources and subresources

Fixed resource	Subresource	RACF resource name	Description
AD		AD	Application-description file
	AD.ADNAME	ADA.name	Application name
	AD.ADGDDEF	ADD.name	Group-definition-ID name
	AD.NAME	ADN.name	Operation extended name in application-description
	AD.OWNER	ADO.name	Owner ID
	AD.GROUP	ADG.name	Authority group ID
	AD.JOBNAME	ADJ.name	Operation job name in application description
	AD.RESNAME	ADR.resname	Special resource name
	AD.SECELEM	ADM.name	Security element name
	AD.UFVAL	ADU.field_name.field_value	User field name and value.
ADEP		ADEP	Selecting all dependencies in the QCP dialog
CL		CL	Calendar data
	CL.CALNAME	CLC.name	Calendar name
CP		CP	Current-plan file
	CP.ADD	CP.ADD	Add workload
	CP.DELETE	CP.DELETE	Delete workload
	CP.MODIFY	CP.MODIFY	Modify workload
	CP.ADDOPER	CP.ADDOPER	Add operation
	CP.DELOPER	CP.DELOPER	Delete operation
	CP.MODOPER	CP.MODOPER	Modify operation
	CP.MODDEP	CP.MODDEP	Modify dependencies
	CP.MODOPSTAT	CP.MODOPSTAT	Modify operation status
	CP.COMMAND*n*	CP.COMMAND*n*	List of commands
	CP.ADNAME	CPA.name	Occurrence name
	CP.CPGDDEF	CPD.name	Occurrence group-definition-ID
	CP.NAME	CPN.name	Operation extended name
	CP.OWNER	CPO.name	Occurrence owner ID
	CP.GROUP	CPG.name	Occurrence authority-group ID
	CP.JOBNAME	CPJ.name	Occurrence operation name
	CP.WSNAME	CPW.name	Current plan workstation name
	CP.ZWSOPER	CPZ.name	Workstation name used by an operation
	CP.SECELEM	CPM name	Security element name
	CP.UFVAL	CPU.field_name.field_value	Operation user field name and value
	CP.RESNAME	CPR.resname	Special resource name
ETT		ETT	ETT dialog
	ET.ETNAME	ETE.name	Name of triggering event
	ET.ADNAME	ETA.name	Name of application to be added
HIST		HIST	Retrieving history data with HIST command
JL		JL	Job library data sets
	JL.DSNAME	JLD.name	Job library data set name
	JL.MEMBER	JLM.name	JCL member name
JS		JS	JCL and job-library file
	JS.ADNAME	JSA.name	Occurrence name
	JS.OWNER	JSO.name	Occurrence owner ID
	JS.GROUP	JSG.name	Occurrence authority group ID
	JS.JOBNAME	JSJ.name	Occurrence operation name
	JS.WSNAME	JSW.name	Current plan workstation name

Table 29. Protected fixed resources and subresources (continued)

Fixed resource	Subresource	RACF resource name	Description
JV		JV	JCL variable-definition file
	JV.OWNER	JVO.name	Owner ID of JCL-variable-definition table
	JV.TABNAME	JVT.name	Name of JCL-variable table
LT		LT	Long-term-plan file
	LT.ADNAME	LTA.name	Occurrence name
	LT.LTGDDEF	LTD.name	Occurrence group-definition ID
	LT.OWNER	LTO.name	Occurrence owner ID
OI		OI	Operator-instruction file
	OI.ADNAME	OIA.name	Application name
PR		PR	Period data
	PR.PERNAME	PRP.name	Period name
RD		RD	Special resources file
	RD.RDNAME	RDR.name	Special resource name
RG		RG	Run cycle group
	RG.RGNAME	RGY.name	Run cycle group name
	RG.OWNER	RGO.name	Run cycle group owner
RL		RL	Ready list data
	RL.ADNAME	RLA.name	Occurrence name
	RL.OWNER	RLO.name	Occurrence owner ID
	RL.GROUP	RLG.name	Occurrence authority-group ID
	RL.WSNAME	RLW.name	Current-plan workstation name
	RL.WSSTAT	RLX.name	Current-plan workstation changed by WSSTAT
RP		RP	Dynamic Workload Console reports
	RP.REPTYPE	RPT.*reptype*	Report type depending on the report you request: **RUNHIST** For job run history reports. **RUNSTATS** For job run statistics. **WWR** For workstation workload runtimes reports. **WWS** For workstation workload summary. **SQL** For reports obtained by customized SQL queries.
SR		SR	Special resources in the current plan
	SR.SRNAME	SRS.name	Special resource name
WS		WS	Workstation data
	WS.WSNAME	WSW.name	Workstation name in workstation database
ARC		ARC	Activate/deactivate automatic recovery
BKP		BKP	Request backup of a resource data set
BUL		BUL	Initiate bulk discovery for the monitoring agent

Table 29. Protected fixed resources and subresources (continued)

Fixed resource	Subresource	RACF resource name	Description
CMAC		CMAC	Data set and Catalog Cleanup used by the Restart and Cleanup function.
CONT		CONT	Refresh RACF subresources
ETAC		ETAC	Activate/deactivate event-triggered tracking
EXEC		EXEC	EX (execute) row command
JSUB		JSUB	Activate/deactivate job submit
REFR		REFR	Refresh LTP and delete CP
WSCL		WSCL	All-workstations-closed data

As shown in Table 29 on page 200, these items exist only as fixed resources:

Name	Protects
ADEP	The use of **ALL DEP** inquiry from EQQSOPGD panel in the Query Current Plan (QCP) dialog. To use this function, you need read or update authority to the ADEP fixed resource.
ARC	The **ACTIVATE/DEACTIVATE automatic recovery** function in the IBM Workload Scheduler for z/OS Service Functions dialog. To use this function, you need update authority to the ARC fixed resource.
BKP	The use of the BACKUP command. BACKUP lets you request a backup of the current plan data set or JCL repository data set. To use this command, you need update access to the BKP fixed resource on the system where the command is issued.
BUL	The use of the BULKDISC command. BULKDISC allows you to initiate a bulk discovery. To use this command you need update access to the BUL fixed resource on the system where the command is issued.
CMAC	The Restart and Cleanup function in the IBM Workload Scheduler for z/OS panels. To use Step Restart, Job Restart and Start Cleanup update authority is needed to the CMAC fixed resource. No authority is required to CMAC for use of Display Cleanup.
CONT	The **RACF RESOURCES** function in the IBM Workload Scheduler for z/OS Service Functions dialog. This lets you activate subresources that are defined after IBM Workload Scheduler for z/OS started. To use this function, you need update authority to the CONT fixed resource.
ETAC	The **ACTIVATE/DEACTIVATE ETT** function in the Service Functions dialog. To use this function, you need update authority to the ETAC fixed resource.
EXEC	The use of the EX (execute) row command. You can issue this command

from the Modify Current® Plan dialog and workstation ready lists, if you have update access to the EXEC fixed resource.

JSUB The **ACTIVATE/DEACTIVATE job submission** function in the IBM Workload Scheduler for z/OS Service Functions dialog or TSO JSUACT command. To use this function, you need update authority to the JSUB fixed resource.

REFR The **REFRESH** function (Delete current plan and reset long-term plan) in the IBM Workload Scheduler for z/OS Service Functions dialog. To use this function, you need update authority to the REFR fixed resource.

WSCL The **All Workstations Closed** function of the Workstation Description dialog. To browse the list of time intervals when all workstations are closed, you need read authority to the WSCL fixed resource. To update the list, you need update authority to the WSCL fixed resource.

Note: Ensure that you restrict access to these fixed resources to users who require them. REFR is particularly important because this function deletes the current plan.

When working with the subresources CP.ADD, CP.DELETE, CP.MODIFY, CP.ADDOPER, CP.DELOPER, CP.MODOPER, CP.MODDEP, CP.MODOPSTAT, and CP.COMMAND*n*, which control actions, consider that:

- The subresources control the actions without filtering the objects.
- The CP.ADD subresource gives the user authority to add new occurrences and operations to existing occurrences. If you want to keep these authorizations separated, use the CP.ADDOPER subresource to give the user authority to add only operations to existing occurrences.
- The CP.DELETE subresource gives the user authority to delete occurrences and operations from the occurrences. If you want to keep these authorizations separated, use the CP.DELOPER subresource to give the user authority to delete only operations from existing occurrences.
- The CP.MODIFY subresource gives the user authority to modify occurrences' attributes and operations in the occurrences. If you want to keep these authorizations separated, use the CP.MODOPER subresource to give the user authority to modify only operations in existing occurrences.
- The CP.MODDEP subresource gives the user authority to add, delete, and modify dependencies.
- When rerunning an occurrence:
  - You can perform a restart and cleanup (JR, SR) only if you are authorized to submit the rerun, JR, and SR commands.
  - If you issue the SC command without having the appropriate authorization, the rerun is performed nevertheless.
- The CP.MODOPSTAT subresource gives the user authority to modify the operation status.

  The CP.MODOPSTAT subresource includes the following commands:

  N Set next logical status

  N-x Set specific logical status

  R Reset Status

-
- Table 30 on page 204 shows the actions that are affected by the subresources that are set in “AUTHDEF” on page 17.

Table 30. Relationships between actions and subresources

Subresources set in AUTHDEF	Impacted actions
CP.ADD	Add occurrence, Add operation, Add group
CP.COMMAND*x*, when the list of commands includes C and CG	Complete group
CP.DELETE	Delete occurrence, Delete operation, Delete group
CP.DELETE, CP.COMMAND*x* when the list of commands includes DG	Delete group
CP.MODIFY	Modify occurrence, Complete occurrence, Modify operation, Remove group, Complete group
CP.ADDOPER, CP.MODDEP, CP.MODOPER, CP.ADD	Add operation
CP.DELOPER, CP.MODDEP, CP.MODOPER, CP.DELETE	Delete operation
CP.MODIFY, CP.MODOPER	Modify operation
CP.MODOPSTAT, CP.MODOPER, CP.MODIFY	Change status (from Modify Occurrence or Modify Operation)
CP.MODDEP, CP.MODOPER, CP.MODIFY	Add, delete, modify dependencies
CP.MODIFY, CP.COMMAND*x* when the list of commands includes RG	Remove group
CP.MODIFY, CP.COMMAND*x* when the list of commands includes C	Complete an occurrence
CP.MODIFY, CP.COMMAND*x* when the list of commands includes W	Set waiting

There are some things to consider when working with fixed resources and subresources that control objects:

- The AD.JOBNAME and CP.JOBNAME subresources protect *only* the JOBNAME field within an application or occurrence. You use these subresources to limit the job names to which the user has access during job setup and similar tasks. If you do not use these subresources, a dialog user might obtain greater authority by using IBM Workload Scheduler for z/OS to perform certain functions. For example, a user could submit an unauthorized job by adding an application to the current plan, changing the job name, and then letting IBM Workload Scheduler for z/OS submit the job.

  For these subresources, only the ACCESS(UPDATE) level is meaningful.
- The subresources AD.GROUP, CP.GROUP, JS.GROUP, and RL.GROUP are used to protect access to IBM Workload Scheduler for z/OS data based on the authority group ID and not application description groups.
- The subresource data is passed to SAF without modifications. Your security product might have restrictions on which characters it allows. For example, RACF resource names cannot contain asterisks, embedded blanks, or DBCS characters.
- The EQQ9RFDE member in the sample library updates the class-descriptor tables with an IBM Workload Scheduler for z/OS-specific class called OPCCLASS.
- Use the CP.ZWSOPER subresource if you want to protect an operation based on the name of the workstation where the operation will be started. You must have update access to this subresource if you want to modify an operation. If you

want to specify dependencies between operations, you must have update authority to both the predecessor and successor operations.

You can use the CP.ZWSOPER subresource to protect against updates to an operation in an occurrence or the unauthorized deletion or addition of an operation in an occurrence. This subresource is not used to protect the addition of an occurrence to the current plan or to protect an occurrence in the current plan that a user attempts to delete, set to waiting, or set to complete. When an occurrence is rerun, access authority is checked only for the particular operation that the rerun is started from.

The subresource CP.ZWSOPER is unlike the subresource CP.WSNAME, which protects workstations but does not protect against updates to operations.

- When no current plan occurrence information is available, subresource protection for job setup and JCL editing tasks is based on information from the application description. For example, if you are adding an occurrence to the CP and you request JCL edit for an operation, subresource requests using owner ID or authority group ID are issued using the owner ID or authority group ID defined in the AD, because the CP occurrence does not yet exist. Similarly, when editing JCL in the LTP dialog, subresources are based on CP occurrence information, if the occurrence is in the CP. If the occurrence is not in the CP, subresource requests are issued using information from the AD.
- The use the HIST (history) command from the IBM Workload Scheduler for z/OS panels, you need at least READ access to the HIST fixed resource.
- Security checks are not performed on user fields for which there is no value specified.
- AD.UFVAL and CP.UFVAL subresources:
  - The AD.UFVAL and CP.UFVAL subresources are used to protect user field names and values. If you specify these subresources in an AUTHDEF statement using the predefined class, IBMOPC, note that the IBMOPC profile supports user fields not longer than 54 characters. The 54 characters is the sum of the characters that comprise the following string:
    - For the AD.UFVAL subresource: ADU.<*field_name*>.<*field_value*>
    - For the CP.UFVAL subresource: CPU.<*field_name*>.<*field_value*>

    Therefore, if you require protection for user fields longer than 54 characters, then you must manually create a new RACF profile, or use an existing profile you have defined, that supports user fields with values longer than 54 characters. For example, the profile could specify MAXLNTH=80 to ensure longer user field names and values are supported.
  - The characters permitted in the ADU.<*field_name*>.<*field_value*> and CPU.<*field_name*>.<*field_value*> strings depend on the security product you use through the system authorization facility (SAF). The security product can be RACF or any other product that works with SAF. No checks are performed to validate the characters used, so you must be careful not to use characters than can cause unexpected results. For example, avoid using characters that are considered wildcard characters for the security product you are using. In the case of RACF, this means avoid using the following wildcard characters: [*, %].

Access requirements to fixed resources for dialog users

To use an IBM Workload Scheduler for z/OS dialog, you need the authority to access the required resources.

About this task

Table 31 shows the resource access that you need for each dialog function.

For example, to permit user CASHIER to add applications to the current plan, you need to:

1. Find the Modify Current Plan (MCP) dialog in the table.
2. Find the add function.
3. Give user CASHIER access to the fixed resources shown for MCP add: read access to the AD and JS profiles and update access to the CP profile.

   Access is not required to the other resources shown for MCP add, unless CASHIER also needs to use the functions that they protect when adding an application. A numeric suffix identifies the note that describes the function. The notes are listed after the table.

If you use subresources, the user must also have access to the subresource owned by the fixed resource.

Table 31. Access requirements to fixed resources for dialog users

Dialog	Function	Fixed resource	Access type
Work Station	Browse workstation	WS	Read
	Update workstation	WS WSCL	Update Read [1]
	Browse workstation closed	WSCL	Read
	Update workstation closed	WSCL	Update
	Print	None	None
Calendar	Browse	CL	Read
	Update	CL	Update
	Print	None	None
Period	Browse	PR	Read
	Update	PR JV	Update Read [2]
	Print	CL	Read

Table 31. Access requirements to fixed resources for dialog users (continued)

Dialog	Function	Fixed resource	Access type
Application Description	Browse	AD CL WS OI RD	Read Read Read Read [3] Read [13]
	Update	AD CL PR WS OI JV RD	Update Read Read Read Update [4] Read [2] Read [14]
	Print	WS	Read [5]
	Mass update	AD CL PR WS JV RD	Update Read Read Read Read Read [14]
Operator Instructions	Browse	OI	Read
	Update	OI	Update
	Print	None	None
	Mass update	None	None
Special resource	Browse	RD WS	Read Read
	Update	RD WS	Update Read
Event Triggered Tracking	Browse	ETT	Read
	Update	ETT	Update
Job Descriptions	Browse	AD WS OI RD	Read Read Read [3] Read [13]
	Update	AD CL PR WS OI JV RD	Update Read Read Read Update [4] Read [2] Read [14]
	Print	WS	Read [5]

Table 31. Access requirements to fixed resources for dialog users (continued)

Dialog	**Function**	**Fixed resource**	**Access type**
JCL Variable Tables	Browse	JV	Read
	Update	JV	Update
	Print	JV	Read
JCL in job library	Browse	JL	Read
	Update	JL	Update
Long-Term Plan	Browse	LT AD CL PR WS	Read Read Read Read Read
	Update (delete or modify) or add	LT AD CL PR WS JV	Update Read Read Read Read Read [2]
	Job setup	LT AD CL PR WS JS	Read Read Read Read Read Update
	Batch	LT	Read
	Display Status	LT	Read
	Set defaults	None	None
Daily Planning	Batch	CP	Read
Work Station Communication	Using ready lists	RL CP JS OI JV EXEC	Update [6] Read [7] Update [8] Read [9] Read [10] Update [12]
	Waiting list	CP JS OI	Read Update [8] Read [9]
	Job setup	CP JS OI	Read Update Read [9]
	Review workstation status	CP	Read
	Define ready lists	None	None

Table 31. Access requirements to fixed resources for dialog users (continued)

Dialog	Function	Fixed resource	Access type
Modify Current Plan (MCP)	Add	AD CP JS JV SR	Read Update Read Read [2] Update [15]
	Update (delete or modify), change status of workstations	CP JS JV SR	Update Update [8] Read [2] Update [15]
	Change status, rerun, error handling	CP JS OI EXEC	Update Update [8] Read [9] Update [12]
	Restart and cleanup	CP JS CMAC	Update Update Update
	Browse	CP JS OI SR	Read Read [11] Read [9] Read [13]
	Job setup	CP JS	Read Update [8]
	Define error lists	None	None
	History data request	HIST	read[16]
Query Current Plan (QCP)	All	CP JS OI SR	Read Read [11] Read [9] Read [13]
Service Functions	Activate/deactivate job submission	JSUB CP	Update Update
	Activate/deactivate automatic recovery	ARC CP	Update Update
	Refresh (delete current plan and reset long-term plan)	REFR LT	Update Update
	Activate RACF resources	CONT	Update
	Activate/deactivate event-triggered tracking	ETAC	Update
	Produce APAR tape	None	None

Note:

1. If you are modifying open intervals for one day
2. If you specify or update a JCL variable table name
3. If you are browsing operator instructions
4. If you are modifying operator instructions
5. If sorted in workstation order
6. If you want to change status
7. If you request a review of details
8. If you want to modify JCL
9. If you want to browse operator instructions
10. If you perform job setup using JCL variable substitution
11. If you want to browse JCL
12. If you want to issue the EX (execute) command
13. If you want to browse special resources
14. If you want to specify special resource allocations
15. If you want to add or update special resources
16. To issue the HIST (history) command, you need at least READ access

Examples of security strategies

You can implement security in IBM Workload Scheduler for z/OS in many ways. Two extreme examples are shown below:

- A centralized strategy in which all security functions are controlled at a central site using fixed resources
- A decentralized strategy in which most security functions are delegated to users, and many subresources are defined.

Your security strategy probably falls somewhere between these two extremes.

A centralized security strategy

All IBM Workload Scheduler for z/OS users are gathered in one area. They have at least a working knowledge of all the major functions of IBM Workload Scheduler for z/OS. Because they share the same tasks, there is little need to divide authority.

The only outside IBM Workload Scheduler for z/OS users are at the printer pool, where operators report progress on a print ready list. Machine room operators do not have IBM Workload Scheduler for z/OS tasks; people in the IBM Workload Scheduler for z/OS area perform reruns and JCL corrections themselves.

These RACF groups are defined:

Group	**Contains**
OPCGROUP	Most of the IBM Workload Scheduler for z/OS users.
OPCSPEC	The manager, the two group leaders, the system programmers responsible for IBM Workload Scheduler for z/OS, and their backups.
OPCPRINT	The users of the ready list at the printer pool.

External access to IBM Workload Scheduler for z/OS

Update access to the IBM Workload Scheduler for z/OS data sets is given to OPCSPEC. This provides access outside IBM Workload Scheduler for z/OS so that the manager and group leaders can submit batch jobs that cause updates, such as daily-plan-extend. The system programmers can use non-IBM Workload Scheduler for z/OS programs to extract diagnostic information.

Access through the IBM Workload Scheduler for z/OS subsystem

About this task

These authorization layers are defined:

1. Subsystem access: OPCSPEC and OPCGROUP are given update access in the APPL class, which lets them use all functions (fixed resources) in the IBM Workload Scheduler for z/OS dialog that are not specifically protected. OPCPRINT is given read access to the IBM Workload Scheduler for z/OS subsystem in the APPL class.
2. Critical functions: Some fixed resources, such as JSUB and REFR, represent functions that have a serious impact on IBM Workload Scheduler for z/OS operation, and can be turned on or off with a single keystroke. Access to these functions is restricted to OPCSPEC to reduce the risk of accidental errors:
   ```
   RDEFINE (OPCCLASS) ARC UACC(NONE)
   PERMIT ARC ID(OPCSPEC) ACCESS(UPDATE) CLASS(OPCCLASS)
   ```
   These steps are repeated for ETAC, JSUB, and REFR.
3. Data updated infrequently: Some IBM Workload Scheduler for z/OS data is updated infrequently, for example, the calendar database is typically updated only once each year, and workstation data even less often. These databases are used by most IBM Workload Scheduler for z/OS functions, so it is a good idea to restrict update access to them:
   ```
   RDEFINE (OPCCLASS) CL UACC(READ)
   PERMIT CL ID(OPCSPEC) ACCESS(UPDATE) CLASS(OPCCLASS)
   ```
   These steps are repeated for PR and WS.
4. Subresource protection: The only subresources are defined for the printer workstation. The OPCPRINT group already has read access to the resources in the APPL class. This lets printer-pool operators enter the functions of IBM Workload Scheduler for z/OS and browse the data. They must also be able to update the ready list at a printer workstation (but not at other workstations):
   a. The fixed resource RL is defined, and OPCPRINT, OPCGROUP, and OPCSPEC are given update access to it:
      ```
      RDEFINE (OPCCLASS) RL UACC(NONE)
      PERMIT RL ID(OPCSPEC) ACCESS(UPDATE) CLASS(OPCCLASS)
      PERMIT RL ID(OPCGROUP) ACCESS(UPDATE) CLASS(OPCCLASS)
      PERMIT RL ID(OPCPRINT) ACCESS(UPDATE) CLASS(OPCCLASS)
      ```
      This lets the printer-pool operators enter the Workstation Communication dialog without authority violations.
   b. The subresource RLW.* is defined. Both OPCGROUP and OPCSPEC are given update access; OPCPRINT is given only read access:
      ```
      RDEFINE (OPCCLASS) RLW.* UACC(NONE)
      PERMIT RLW.* ID(OPCSPEC) ACCESS(UPDATE) CLASS(OPCCLASS)
      PERMIT RLW.* ID(OPCGROUP) ACCESS(UPDATE) CLASS(OPCCLASS)
      PERMIT RLW.* ID(OPCPRINT) ACCESS(READ) CLASS(OPCCLASS)
      ```
      This becomes the default access for all workstations that are not explicitly defined with further subresource definitions.
   c. Finally, the subresource RLW.PRT is defined; PRT is the IBM Workload Scheduler for z/OS name of the workstation. OPCPRINT is given update access:

```
RDEFINE (OPCCLASS) RLW.PRT UACC(NONE)
PERMIT RLW.PRT ID(OPCSPEC) ACCESS(UPDATE) CLASS(OPCCLASS)
PERMIT RLW.PRT ID(OPCGROUP) ACCESS(UPDATE) CLASS(OPCCLASS)
PERMIT RLW.PRT ID(OPCPRINT) ACCESS(UPDATE) CLASS(OPCCLASS)
```

OPCPRINT group members can now browse data in IBM Workload Scheduler for z/OS and update the ready list for the printer pool.

A decentralized security strategy

Most IBM Workload Scheduler for z/OS work is delegated to representatives of the user departments. They perform all IBM Workload Scheduler for z/OS functions (including reruns and JCL corrections during the day) but only for the applications in their own department.

During the second and third shifts, machine-room operators take care of reruns and JCL corrections for all departments.

These RACF groups are defined:

Group	Contains
OPCGROUP	Most of the IBM Workload Scheduler for z/OS users. Unlike the users in a centralized installation, these users work alone and perform all IBM Workload Scheduler for z/OS functions on a limited number of applications.
OPCSPEC	The schedulers and system support people who keep IBM Workload Scheduler for z/OS running continuously.
OPER	The people who correct jobs that end in error overnight.

The major difference from a centralized installation is that authority is divided by department, not by IBM Workload Scheduler for z/OS function. The decentralized location relies mainly on IBM Workload Scheduler for z/OS subresources for security. This makes name standards more important if the number of profiles is to be kept to a minimum. The administrator must decide which subresource names to use. For example, access to applications could be restricted by job name, owner ID, or authority group ID. Critical functions such as REFR (refresh) should not be decentralized.

External access to IBM Workload Scheduler for z/OS

As in the centralized installation, update access to IBM Workload Scheduler for z/OS data sets is given to members in the OPCSPEC group. In a decentralized installation, though, all other groups must have ACCESS(NONE). This prevents members of OPCGROUP or OPER from reading data that belongs to other users. With read access to the IBM Workload Scheduler for z/OS data sets, a user could run a utility program outside the IBM Workload Scheduler for z/OS subsystem to look at data that belongs to another user.

Access through the IBM Workload Scheduler for z/OS subsystem

About this task

These authorization layers are defined:

1. Subsystem access: All groups are given update access to the IBM Workload Scheduler for z/OS subsystem in the APPL class. This lets all users update

most IBM Workload Scheduler for z/OS functions (fixed resources) for their own department. The APPL class specification is the default if a fixed resource is defined.

Another way to handle fixed resources is to define them individually and give update access to OPCGROUP and OPCSPEC. But the OPER group needs update access only to CP, JS, and RL (for JCL corrections and reruns). They could have ACCESS(NONE) to the rest of the fixed resources. This would prevent them from entering any IBM Workload Scheduler for z/OS dialog that they do not need for their work.

2. Critical functions: Some fixed resources, such as JSUB and REFR, represent functions that have a serious impact on IBM Workload Scheduler for z/OS operation, and can be turned on or off with a single keystroke. Access to these functions should not be decentralized. Access is restricted to OPCSPEC to reduce the risk of accidental errors:

   ```
   RDEFINE (OPCCLASS) ARC UACC(NONE)
   PERMIT ARC ID(OPCSPEC) ACCESS(UPDATE) CLASS(OPCCLASS)
   ```

   These steps are repeated for ETAC, JSUB, and REFR.

3. Subresource protection: The installation protects access to applications and JCL using subresources. But the installation does not have consistent naming conventions for applications. So subresource protection is implemented through the owner ID and job name, which have consistent naming conventions.

   a. These subresources are specified on the AUTHDEF statement:

Table 32. Subresources specified on the AUTHDEF statement

subresources	
AD.JOBNAME	LT.OWNER
AD.OWNER	JS.JOBNAME
CP.JOBNAME	JS.OWNER
CP.OWNER	RL.OWNER.

   The subresources AD.JOBNAME, CP.JOBNAME, and JS.JOBNAME are used to prevent users from specifying unauthorized job names when they create an application. Otherwise, IBM Workload Scheduler for z/OS could be used to submit a job with a job name that the users do not normally have access to.

   b. The RACF resource names are defined with ACCESS(NONE), so the default access for all users to these subresources is NONE:

   ```
   RDEFINE (OPCCLASS) ADJ.* UACC(NONE)
   ```

   This is repeated for ADO.*, CPJ.*, CPO.*, LTO.*, JSJ.*, JSO.*, and RLO.* resource names.

   c. When profiles are created, OPCGROUP members receive the authority to decide the access list to their own subresources.

   OPCSPEC is given update access in case this is needed for support:

   ```
   PERMIT ADO.* ID(OPCSPEC) ACCESS(UPDATE) CLASS(OPCCLASS)
   ```

   This is repeated for each subresource.

   d. OPER is given access to the CP.OWNER, CP.JOBNAME, JS.JOBNAME, JS.OWNER, and RL.OWNER subresources so that operators can work during night shifts:

   ```
   PERMIT CPO.* ID(OPER) ACCESS(UPDATE) CLASS(OPCCLASS)
   PERMIT CPJ.* ID(OPER) ACCESS(UPDATE) CLASS(OPCCLASS)
   PERMIT JSJ.* ID(OPER) ACCESS(UPDATE) CLASS(OPCCLASS)
   PERMIT JSO.* ID(OPER) ACCESS(UPDATE) CLASS(OPCCLASS)
   PERMIT RLO.* ID(OPER) ACCESS(UPDATE) CLASS(OPCCLASS)
   ```

If many resources with similar names have the same access list, the resources can be grouped under generic profiles with the percent sign (%). For example, the ADO, CPO, JSO, LTO, and RLO profiles could be specified as one profile, %%O.*. Note that *O.* is an invalid RACF entity.

Many RACF resource names must be defined in the OPCCLASS resource class to protect the data of every owner. Each subresource has its own profile, unless some subresources can be grouped under generic profiles. For example, the owner IDs PAYROLL, PAYROLL-A, and PAYROLL-02, can be grouped as PAYROLL*.

Defining profiles might seem like a lot of work, but the number of owners is usually limited, and you can often use generic profiles. Because you can have many more job names than owner IDs, generic definitions of job names are even more important. Most jobs can be handled with a small number of generic profiles.

Chapter 4. IBM Workload Scheduler for z/OS exits

Read the following information to understand General-use Programming Interface and Associated Guidance Information. It describes exits that are called by IBM Workload Scheduler for z/OS. Your own programs can use the information passed by the exits to perform a variety of functions.

Exits with name prefix EQQUX*nnn* (where *nnn* is a number) are called when IBM Workload Scheduler for z/OS is started. Exits with name prefix EQQUX*xxx* (where *xxx* is not a number) are not called by the scheduler. For example, EQQUXCAT is called (if present) by the EQQDELDS sample or by the EQQCLEAN program, while EQQUXPIF is called by a PIF program during the application description update (INSERT AD or REPLACE AD).

Each exit is loaded if the exit module exists, if the exit has not been disabled, and if the exit has not been replaced by another exit name on the EXITS statement. See "EXITS" on page 57 for a detailed description of the EXITS statement.

The JCL-variable-substitution exit is called at either job setup or job submission, when a JCL variable has retrieved its value. The JCL-imbed and automatic-recovery exits are called via IBM Workload Scheduler for z/OS JCL statements. None of these exits is affected by the EXITS statement.

The EQQDPUE1 exit is used by daily planning batch jobs and is not affected by the EXITS statement. The batch jobs use the exit if it is available. If the exit is not found, a message is written to the message log of the batch job.

Exits are invoked using standard linkage conventions. When the exit is entered, register 1 points to a parameter list. Each address in this list points to a parameter that is passed to the exit. Table 33 contains information about the IBM Workload Scheduler for z/OS exits.

Table 33. IBM Workload Scheduler for z/OS exits

Exit name	Loaded by	AMODE at entry	RMODE when linking	Exit description
EQQUX000	tracker and controller	31	ANY	IBM Workload Scheduler for z/OS start/stop
EQQUX001	controller	31	24	Job submit
EQQUX002	controller	31	24	Job library read
EQQUX003	controller	31	ANY	Application description feedback
EQQUX004	tracker	24	24	Event filtering
EQQUX005	tracker	24	24	JCC SYSOUT archiving
EQQUX006	tracker	24	24	JCC incident-record create
EQQUX007	controller	31	ANY	Operation status change
EQQUX009	controller	31	ANY	Operation initiation
EQQUX011	controller	31	ANY	Job-tracking log write
EQQUX013	controller	31	ANY	Job-tailoring prevention
EQQUX014	controller	24	24	Scheduler time adjustment

Table 33. IBM Workload Scheduler for z/OS exits (continued)

Exit name	Loaded by	AMODE at entry	RMODE when linking	Exit description
EQQUXCAT	EQQDELDS sample or EQQCLEAN program	31	ANY	EQQDELDS/EQQCLEAN catalog exit
user-defined	Fetch directive on //%OPC JCL statement	24/31	24	JCL imbed
user-defined	JCL variable	24/31	24	Variable substitution
user-defined	Rebuild parameter of AJR control statement	31	ANY	Automatic job recovery
EQQDPUE1	Daily planning job	31	ANY	Daily planning report
EQQUXPIF	PIF	31	ANY	Validation of changes done in AD (INSERT or REPLACE AD)
EQQUXGDG	EQQCLEAN program	31	ANY	EQQCLEAN GDG resolution
EQQUXSAZ	controller	31	24	System Automation command

Note:

1. All exits are entered with the RACF authority of the IBM Workload Scheduler for z/OS subsystem.
2. Calling the program-interface module, EQQYCOM, from exits that are taken by the controller address space is not recommended and will cause unpredictable results if attempted.

See Appendix B, "Sample library (SEQQSAMP)," on page 391 and *Diagnosis Guide and Reference* for more information about the running environment of the exits.

The following pages describe the IBM Workload Scheduler for z/OS exits and include information about:
- Conditions that cause the exit to be called
- The mapping of each parameter (described in assembler-language format).

IBM Workload Scheduler for z/OS user exits invocation

Start/stop exit (EQQUX000)

Exit EQQUX000 is invoked in the Tracker Address Space and the Controller Address Space (Standby and Active). The invoker is the Subsystem Dispatcher task EQQZMAIN, when initializing the address space and when terminating the processing in the address space.

Job-submit exit (EQQUX001)

Exit EQQUX001 is invoked in the Controller Address Space (Active). The invoker is the Workstation Analyser task, when a ready operation is selected for processing and the JCL for the operation is available. During the running of the exit other tasks in the address space are prevented from reading and updating the Current Plan.

When exit EQQUX001 is invoked to add a new step, it scans the entire JCL stream to locate the jobcard. If it does not find the jobcard, it does not add the new line in the JCL.

See also the section *Limitation on the number of job steps* in the *IBM Workload Scheduler for z/OS: Managing the Workload* manual.

Job-library-read exit (EQQUX002)

Exit EQQUX002 is invoked in the Controller Address Space (Active). Invokers are Workstation Analyser and General service tasks, when a ready operation is selected for processing. During the running of the exit other tasks in the address space are prevented from reading and updating the Current Plan.

Application-description-feedback exit (EQQUX003)

Exit EQQUX003 is invoked in the Controller Address Space (Active). Invokers are the Event Manager, the Workstation Analyser, the General Service tasks, Automatic Recovery and the Normal Mode manager. During the running of the exit other tasks in the address space are prevented from reading and updating the Current Plan.

Event-filtering exit (EQQUX004)

Exit EQQUX004 is invoked in the tracker Address space and in the controller address space if the controller has an Event Writer task (EWTRTASK(YES) specified in the OPCOPTS initialization statement). Invoker is the Event Writer task.

SYSOUT archiving exit (EQQUX005)

Exit EQQUX005 is invoked in the tracker Address space and in the controller address space if the Controller has JCC active (JCCTASK(YES) specified in the OPCOPTS initialization statement. Invoker is the Job Completion checker.

Incident-record-create exit (EQQUX006)

Exit EQQUX006 is invoked in Tracker address space and in the controller address space if the controller has JCC task active (JCCTRTASK(YES) specified in the OPCOPTS initialization statement). Invoker is the Job Completion checker.

Operation-status-change exit (EQQUX007)

Exit EQQUX007 is invoked in the Controller Address Space (Active). Invokers are the Event Manager, the Workstation Analyzer, the General Service tasks, Automatic Recovery, and the Normal Mode manager. During the running of the exit other tasks in the address space are prevented from reading and updating the Current Plan.

Operation-initiation exit (EQQUX009)

Exit EQQUX009 is invoked in the Controller Address Space (Active). Invoker is the Data Router task.

Job-tracking log write exit (EQQUX011)

Exit EQQUX011 is invoked in the Controller Address Space (Active). Invokers are the Event Manager, the Workstation Analyzer, the General Service tasks, Automatic Recovery and the Normal Mode manager. During the running of the exit other tasks in the address space may be prevented from reading and updating the Current Plan.

Job-tailoring prevention exit (EQQUX013)

EQQUX013 is invoked in the Controller Address Space. Invoker is the Workstation Analyzer task, when a ready operation is selected for processing and the JCL for the operation is available. During the running of the exit, other tasks in the address space are prevented from reading and updating the current plan.

Time-dependent-operation exit (EQQUX014)

Exit EQQUX014 is invoked in the Controller Address Space (Active). Invoker can be any Normal Mode Manager, General Server, Workstation Analyzer, or Event Manager tasks, when a time dependent operation is set to ready status in a z/OS environment. During the running of the exit other tasks in the address space are prevented from reading and updating the current plan.

JCL-imbed exit (on FETCH directive)

A JCL-Imbed exit is invoked in the Controller Address Space (Active). Invokers are the Workstation Analyser subtask and the General Service tasks. During the running of the exit other tasks in the address space are prevented from reading and updating the Current Plan.

Variable-substitution exit (on JCL or job definition variable)

A Variable-Substitution exit is invoked in the Controller Address Space (Active) or during the daily planning process. Invokers are:

- The Workstation Analyser subtask and the General Service tasks, if you are submitting a job contained in the job library.
- The job definition parsing process, if you are submitting a job definition contained in the SCRPTLIB library, using the MCP dialog or running a Daily Planning batch job.

During the running of the exit other tasks in the address space are prevented from reading and updating the Current Plan.

Automatic-job-recovery exit (on RECOVER statement)

An Automatic-Job-Recovery Exit is invoked in the Controller Address Space (Active). Invoker is the Automatic Recovery task. During the running of the exit other tasks in the address space are prevented from reading and updating the Current Plan.

Daily-planning-report exit (EQQDPUE1)

Invoked by the batch program EQQDPRPT.

EQQDELDS/EQQCLEAN catalog exit (EQQUXCAT)

EQQUXCAT is invoked by the sample EQQDELDS or the program EQQCLEAN just before executing any single cleanup action.

EQQCLEAN GDG resolution exit (EQQUXGDG)

EQQUXGDG is invoked by the EQQCLEAN program just before executing any single GDG overwrite action into JES control blocks.

Application-description-validation (EQQUXPIF)

EQQUXPIF is invoked by a PIF program during the application description update (INSERT AD or REPLACE AD).

Daily-planning-scheduling environment exit EQQDPX01

The daily-planning-scheduling environment exit (EQQDPX01) is called by the IBM Workload Scheduler for z/OS daily-planning batch jobs and is used to set or modify the scheduling environment names associated to the operations in the plan.

This exit can be invoked for all new operations that do not run on a fault-tolerant workstation. The exit is optional: the daily plan batch attempts to load it, and when it finds it, it puts it to use. The exit should be used with care as it could affect system performance.

EQQDPX01 parameters

```
FUNC      DS   CL6       (Function type)
ADID      DS   CL 16     (Name of current application)
OPNUM     DS   F         (Operation number)
JOBNAME   DS   CL8       (Job name)
WSNAME    DS   CL4       (Operation workstation name)
SPECNR    DS   H         (Number of special resources)
SPECBUF   DS   A         (Special resource buffer)
WSCHENV   DS   CL 16     (Scheduling Environment Name)
MCAUSERF  DS   A         (User field)
```

FUNC Function type:
- 'INIT ' first call
- 'TERM ' last call
- 'CHECK ' check SCHENV call

The exit is called at DP batch start with function type INIT and at DP batch end with function type TERM. This is so that the open and the close only at the beginning and at the end of DP batch (to minimize impact on performance). The exit is called with function type CHECK each time an operation must be checked.

FUNC Is present only for compatibility reasons.

ADID
Is the name of the application that the job belongs to.

OPNUM
Is the operation number of the operation representing this job.

JOBNAME
Is the name of the job associated to the operation.

WSNAME
Is the name of the workstation where the operation is to run.

SPECNR
Is the number of special resource names in SPECBUF.

SPECBUF
Is an address to a buffer that contains a number of 64-byte fields. The number of 64-byte fields in the buffer is indicated by SPECNR. The first 44 bytes of each field contain the name of the special resource. The last 20 bytes of each field are reserved for future use.

WSCHENV
Is the scheduling environment name currently stored in the CP operation record. This value can be modified by the exit.

MCAUSERF
This field is reserved for users. IBM Workload Scheduler for z/OS does not use or update it.

Installing the exit

The load module implementing this exit must be link-edited into an APF-authorized library in the LNKLST concatenation or defined by the STEPLIB DD statement in the IBM Workload Scheduler for z/OS JCL procedure.

If the load module performs any input or output, it must be link-edited with RMODE(24) according to normal z/OS restrictions. Otherwise, it can be link-edited with RMODE(ANY). IBM Workload Scheduler for z/OS invokes the exit in AMODE 31. The AMODE parameter specified at link-edit time has no effect.

Interface to the exit

This exit is invoked in task mode, problem state, and key 8 and the job-step task is APF-authorized. The active task runs with the same access authority as the job-step task. The exit must restore this state before returning to its caller. Control is passed to the exit using the BAL instruction. The exit must return to its caller using the address and addressing mode passed to it, usually register 14.

The exit is entered in AMODE 31 but must switch to AMODE 24 before performing any input or output operations, and then switch back to AMODE 31 before returning to the caller.

When the exit is entered, register 1 contains the address of the parameter list. Each address in this list is used to locate the parameter value. These parameters are passed to the exit:

IBM Workload Scheduler for z/OS Start/Stop exit (EQQUX000)

EQQUX000 is called when IBM Workload Scheduler for z/OS is starting and when it is ending normally. You can use this exit to allocate resources when IBM Workload Scheduler for z/OS is started and to release them when IBM Workload Scheduler for z/OS is stopped. This avoids the extra overheads involved in allocating and then releasing resources each time they are used.

The sample library SEQQSAMP that was created during installation contains the EQQUX0N exit, which is a sample of start/stop exits. For information about this library and its samples, see Appendix B, “Sample library (SEQQSAMP),” on page 391.

Installing the exit

The load module implementing the start/stop exit must be link-edited into an APF-authorized library in the LNKLST concatenation or defined by the STEPLIB DD statement in the IBM Workload Scheduler for z/OS JCL procedure. If the load module performs any input or output operations it must be link-edited with RMODE(24) according to normal z/OS restrictions. Or it can be link-edited with RMODE(ANY).

IBM Workload Scheduler for z/OS invokes the exit in AMODE 31; the AMODE parameter specified at link-edit time has no effect.

Interface to the exit

The start/stop exit is invoked in task mode, problem state, and key 8 and the job-step task is APF-authorized. The active task runs with the same access authority as the job-step task. The exit must restore this state before returning to its caller.

Control is passed to the exit using the BAL instruction. The exit must return to its caller using the address and addressing mode passed to it in general register 14.

The exit is entered in AMODE 31 but must switch to AMODE 24 before performing any input or output operations, and then switch back to AMODE 31 before returning to the caller.

When the exit is entered, register 1 contains the address of the parameter list. Each address in this list is used to locate the parameter value. These parameters are passed to the exit:

EQQUX000 parameters

```
ACTION    DS   CL8    (Start/stop action)
MCAUSERF  DS   A      (User field)
```

ACTION has the value START when the exit is called during IBM Workload Scheduler for z/OS start. MCAUSERF is zero for this initial call. Normally, this exit will perform exit initialization functions for the start call when you start IBM Workload Scheduler for z/OS. If the exit needs to allocate storage that is used while IBM Workload Scheduler for z/OS is active, you should update MCAUSERF to address this storage. ACTION has the value STOP when the exit is called during IBM Workload Scheduler for z/OS termination. Normally, this exit performs exit termination functions for the stop call when you stop IBM Workload Scheduler for z/OS. If MCAUSERF is updated by the start call, the same value is passed to the exit for the stop call.

Job-submit exit (EQQUX001)

EQQUX001 is called when IBM Workload Scheduler for z/OS is about to submit a batch job or start a started task. You can use this exit to:

- Modify the JCL stream, but you cannot increase the number of JCL records, unless you use the OPCOPTS EXIT01SZ keyword.
- Modify the user that submits jobs with centralized scripts on fault-tolerant workstations.
- Make the same user own both the stand-alone cleanup job and the original job.

The sample library SEQQSAMP that was created during installation contains the EQQUX001 exit, which is a sample of job-submit exits. For information about this library and its samples, see Appendix B, “Sample library (SEQQSAMP),” on page 391.

Installing the exit

The load module implementing the job-submit exit must be link-edited into an APF-authorized library in the LNKLST concatenation or to a library defined by the STEPLIB DD statement in the IBM Workload Scheduler for z/OS JCL procedure. The load module should be link-edited with RMODE(24) and AMODE(31) attributes.

AMODE(24) is also supported but is not recommended.

Interface to the exit

The job-submit exit is invoked in task mode, problem state, and key 8 and the job-step task is APF-authorized. The active task runs with the same access authority as the job-step task. The exit must restore this state before returning to its caller.

IBM Workload Scheduler for z/OS invokes the exit in the addressing mode defined by the load module's AMODE attribute. When the exit is called in bit addressing mode, the job stream passed to the exit resides above the 16M line. When the exit is called in 24 bit addressing mode, the job stream passed to the exit resides below the 16M line.

Control is passed to the exit using the BASSM instruction. The exit must return to its caller using the address passed to it in general register 14. The exit can return in any addressing mode.

When the exit is entered, register 1 contains the address of the parameter list. Each address in this list is used to locate the parameter value. These parameters are passed to the exit:

EQQUX001 parameters

```
JOBNAME   DS   CL8       (Job name)
JCLLEN    DS   F         (Size, in bytes, of current job stream)
JCLAREA   DS   n * CL80  (All records in job stream)
LATEOUT   DS   CL10      (Latest start, format YYMMDDHHMM)
ESTDUR    DS   CL4       (Estimated duration, format HHMM)
NUMPS     DS   H         (Number of parallel servers required)
NUMR1     DS   H         (Amount required of workstation resource 1)
NUMR2     DS   H         (Amount required of workstation resource 2)
SPECRES   DS   CL8       (First 8 characters of first special resource name)
ADID      DS   CL16      (Name of current application)
MCAUSERF  DS   A         (User field)
GROUP     DS   CL8       (Name of current authority group)
RUSER     DS   CL8       (Name of RACF user submitting the job)
OPERTYPE  DS   CL1       (J for JCL job, S for started task, F for centralized end-to-end
                          job and for jobs that run on zcentric workstations)
UPDAT     DS   C         (Y or N, defines job origin)
JCLUSER   DS   CL8       (Last user updating this job)
JCLUTIME  DS   CL10      (Time of last update, format YYMMDDHHMM)
OPNUM     DS   F         (Operation number)
IATIME    DS   CL10      (Occurrence input-arrival time, YYMMDDHHMM)
OWNER     DS   CL16      (Application owner name)
SPECNR    DS   H         (Number of special resources)
SPECBUF   DS   A         (Special resource buffer)
WSNAME    DS   CL4       (Operation workstation name)
RETCO     DS   CL4       (Operation error code)
NEWREC    DS   F         (Number of JCL lines in NEWJCL)
NEWJCL    DS   n*CL80    (New jclarea)
USDREC    DS   F         (Number of JCL lines used in NEWJCL)
XINFO     DS   A         (Extended information address)
XJNAMLEN  DS   F         (Extended job name length)
CALTYP    DS   C         (Type of call: N if caller by WASUB, R if caller by WASUJ)
NOREEX    DS   C         (Type of call: N if first call, Y if second or subsequent calls
WSCHENV   DS   CL 16     (Scheduling Environment Name)
OCCPTR    DS   A         (Address of occurrence data)
OPRPTR    DS   A         (Address of operation data)
USRFNR    DS   F         (Number of user fields)
USRFAREA  DS   A         (User fields area address)
```

Note: EQQUX001 is also called when a job is resubmitted to perform a Restart and Cleanup. In this case, JCL changes are ignored. The JCL area is passed to the exit but no changes are applied. Only the exit's user ID and return code information is processed.

JOBNAME
The name of the job that is about to be submitted.

JCLLEN
The size, in bytes, of the job.

JCLAREA
The JCL records in the job.

LATEOUT
The latest-start-time value that IBM Workload Scheduler for z/OS has calculated for the job.

ESTDUR
The estimated duration of this job.

NUMPS
The number of parallel servers required.

NUMR1
The amount of workstation resource 1 required.

NUMR2
The amount of workstation resource 2 required.

SPECRES
The first 8 characters of the special resource name.

Note: When an operation is defined with more than one special resource, the SPECRES parameter contains the resource which is physically first in the ADR record in the EQQADDS data set. This is initially the special resource defined first for this operation, but is subject to change any time that a user adds or removes a special resource from the operation.

ADID The name of the application that the job is part of.

MCAUSERF
A user field that is also passed to the EQQUX000 exit. IBM Workload Scheduler for z/OS does not use or update the MCAUSERF field.

GROUP
The name of the authority group that the current operation belongs to.

RUSER
The name of the RACF user that owns the job. This parameter contains 8 blanks when the exit is called. The exit can update this parameter, if required, to cause the job to be submitted under the specified user ID.

The returned RUSER value is stored in the CP file. It guarantees that, whenever a stand-alone cleanup job is submitted, it has the same owner as the original job.

Different ways to set it are suggested in the EQQUX001 sample job contained in the SEQQSAMP library. A code sample extracting the value of the USER parameter from the JOB card is included as well. Refer to the comments in the sample for more details.

You can also use this parameter to modify the user submitting a job with a centralized script. If you use this keyword to specify the name of the user who submits the specified script or command on a Windows fault-tolerant workstation, you can:

- Associate this user name to the Windows workstation in the USRREC initialization statement.
- Set LOCALPSW(YES) in the TOPOLOGY statement, then use the user utility to define the name and password of the user in a local file on the Windows workstation.

This parameter is supported even if the job is sent to another system via a submit/release data set. So, there is a possibility that the SUBMIT SUBTASK of the controller or of the tracker which is submitting a given job may abend while executing under that RUSER-supplied user ID rather than under the user ID associated with the IBM Workload Scheduler for z/OS started task. If this should occur, DUMPTASK may fail with an ABEND913 if the user ID in control does not have WRITE access to the SYSMDUMP data set. For this reason, SYSMDUMP data sets should be defined with a UACC of UPDATE, that is they should be WRITE-ENABLED to all user IDs under which an IBM Workload Scheduler for z/OS-scheduled job might possibly be submitted.

If RUSER is blank and the job card does not specify the USER keyword, the job is submitted with the authority of the IBM Workload Scheduler for z/OS started task.

OPERTYPE
: Has the value J for a JCL job, S for a started task, or F for a centralized end-to-end job and a job that runs on z-centric and dynamic workstations.

UPDAT
: Has the value Y if the current job was retrieved from the EQQJS*n*DS data set. In all other cases, UPDAT has the value N.

JCLUSER
: The name of the last TSO user to update the current job. This parameter is meaningful only if UPDAT has a value Y.

JCLUTIME
: The date and time of the last update to the current job. This parameter is meaningful only if UPDAT has a value Y.

OPNUM
: The operation number of the operation representing this job.

IATIME
: The input-arrival time of the application occurrence that this job belongs to.

OWNER
: The name of the owner of the current application.

SPECNR
: The number of special resource names in SPECBUF.

SPECBUF
: An address to a buffer that contains a number of 64-byte fields. The number of 64-byte fields in the buffer is indicated by SPECNR. The first 44 bytes of each field contain the name of the special resource. The last 20 bytes of each field are reserved for future use.

WSNAME
: The name of the operation workstation.

RETCO
: The name of a field that can be used by the User Exit to stop submission and set the related error code. Refer to the description of the **SUBFAILACTION** keyword in Chapter 1, "Defining initialization statements," on page 3 for more details.

NEWREC
: The size of the new JCLAREA. It is expressed in number of JCL lines, each of which has a length of 80 bytes. The JCLAREA value is provided to the exit by the scheduler and cannot be changed. If the OPCOPTS EXIT01SZ keyword is used and the NEWREC value provided by IBM Workload Scheduler for z/OS to the exit is equal to zero, the scheduler does not have enough storage to build the new JCLAREA.

NEWJCL
: The area where the enlarged JCL is copied. It is provided by IBM Workload Scheduler for z/OS to the exit to allow the JCL size increase.

USDREC
: The number of lines that belong to the enlarged JCL that is copied into the new jclarea.

XINFO
: The address of the data specified in the Extended Info field of the Current

Plan for the corresponding operation. If its value is 0, no extended information is available in the current plan.

XJNAMLEN
The length that you specify in the Operation Extended Name field of the Current Plan for the corresponding operation. It is a sub-field of the extended information available in the current plan.

WSCHENV
The scheduling environment name currently stored in the CP operation record. This value can be modified by the exit.

OCCPTR
The address of the common data of record CPLREC3C.

OPRPTR
The address of the common data of record CPLREC3P.

USRFNR
The number of user field records in USRFAREA.

USRFAREA
The address of the user field area. It is laid out as follows:

USRFAREA

```
USRFNAME DS   CL16         (User field name)
USRFVAL  DS   CL54         (User field value)
```

Job-library-read exit (EQQUX002)

The job-library-read exit (EQQUX002) is called when IBM Workload Scheduler for z/OS is about to retrieve a batch job that does not exist in the EQQJSnDS data set. The exit is used to create a job stream to be submitted to JES by IBM Workload Scheduler for z/OS.

The sample library SEQQSAMP that was created during installation contains the EQQUX002 exit, which is a sample of job-library-read exits. For information about this library and its samples, see Appendix B, “Sample library (SEQQSAMP),” on page 391.

Installing the exit

The load module implementing the job-library-read exit must be link-edited into an APF-authorized library in the LNKLST concatenation or defined by the STEPLIB DD statement in the IBM Workload Scheduler for z/OS JCL procedure. The load module should be link-edited with RMODE(24) according to normal z/OS restrictions.

Note: EQQUX002 must be coded REENTRANT if the value of the GSTASK keyword on the OPCOPTS statement is greater than 1.

Interface to the exit

The job-library-read exit is invoked in task mode, problem state, and key 8 and the job-step task is APF-authorized. The active task runs with the same access authority as the job-step task. The exit must restore this state before returning to its caller.

Control is passed to the exit using the BAL instruction. The exit must return to its caller using the address and addressing mode passed to it in general register 14.

If the exit abends, it is flagged as *not executable*; IBM Workload Scheduler for z/OS does not try to call the exit again.

When the exit is entered, register 1 contains the address of the parameter list. Each address in this list is used to locate the parameter value. These parameters are passed to the exit:

EQQUX002 parameters

```
TYPE      DS   CL1   (Constant = J)
FUNC      DS   CL1   (Constant = G)
JOBNAME   DS   CL8   (Job name)
IOAREA    DS   A     (Address of I/O area)
IOAREAL   DS   F     (Size of I/O area)
RETCODE   DS   X     (Return code)
DATAL     DS   F     (Amount of data returned)
ERRDATA   DS   CL78  (Error message returned)
ADID      DS   CL16  (Name of current application)
USRAREA   DS   A     (User field, 0 at first call)
JCLUSER   DS   CL8   (Last user updating this job)
OPNUM     DS   F     (Operation number)
IATIME    DS   CL10  (Occurrence input-arrival time, YYMMDDHHMM)
VAROCCP   DS   A     (Address of occurrence data if operation is in CP)
VAROPRP   DS   A     (Address of operation data if operation is in CP)
VARWSP    DS   A     (Address of workstation data if operation is in CP)
MCAUSERF  DS   A     (Address set by the user in the EQQUX000 exit)
OCCPTR    DS   A     (Address of occurrence data)
OPRPTR    DS   A     (Address of operation data)
WSPTR     DS   A     (Address of workstation data)
AUTHGROU  DS   CL8   (Authority group)
MEMPRO    DS   CL1   (Indicator of memory problems)
TASKPTR   DS   A     (Address of TCB of caller task)
XINFO     DS   A     (Extended information address)
XJNAMLEN  DS   F     (Extended job name length)
USRFNR    DS   F         (Number of user fields)
USRFAREA  DS   A         (User fields area address)
```

TYPE Is present only for compatibility reasons.

FUNC Is present only for compatibility reasons.

JOBNAME
The name of the job that is to be submitted.

IOAREA
The address of a buffer that is allocated by IBM Workload Scheduler for z/OS, where JCL records for the current job must be placed.

IOAREAL
The amount of space, in bytes, in the IOAREA buffer

RETCODE
Is set by the exit. These values are valid:

0 Normal return.

4 End of data reached for the current job.

16 The job could not be found in any input data set.

20 There is no JCL to be returned by the exit. IBM Workload Scheduler for z/OS attempts to retrieve the JCL from EQQJBLIB.

44 Not enough space. The amount of free space in the IOAREA buffer (as determined by IOAREAL) is not enough to contain the next block of data.

241 I/O error has occurred.

242 An open error has occurred. One or more input data sets could not be opened.

The exit is called again to continue processing the same job when a return code 0 or 44 is returned. All other return codes end processing of the current job.

DATAL
The amount of data returned by the exit when the return code is 0 or 4.

ERRDATA
A user message area where you can describe a problem found in the exit. The text is issued in message EQQJ576 if return code 242 is set by the exit or in message EQQJ580 if return code 241 is set.

Note: If you modify the message library entry for EQQJ576 or EQQJ580 to generate a WTO, you must ensure that no more than 70 characters of message text are defined for each line. Reorganize the text, if required. Also, ensure that ERRDATA itself does not exceed 70 characters.

ADID The name of the current application.

USRAREA
Is zero the first time the exit is called to retrieve a job. The exit can set this parameter to any value. IBM Workload Scheduler for z/OS does not use or update this parameter.

The exit should use the USRAREA parameter whenever it returns a return code 0 or 44. Normally, the USRAREA parameter is used to contain the address of a work area that the exit has performed a GETMAIN on. This work area should contain enough information to enable the exit to continue processing the same job.

JCLUSER
Is zero the first time the exit is called. The exit should set this parameter to the name of the TSO user that is used for authority checking when the JCL contains automatic recovery statements.

OPNUM
The operation number of the operation representing this job.

IATIME
The input-arrival time of the application occurrence that this job belongs to.

VAROCCP
The address of occurrence data. The storage at this address is mapped by the segment CPOC of the program interface (PIF).

VAROPRP
The address of operation data. The storage at this address is mapped by the PIF segment CPOP.

VARWSP
The address of workstation data. The storage at this address is mapped by the PIF segment CPWS.

Note: VAROCCP, VAROPRP, and VARWSP contain valid addresses only if the exit is invoked for operations that are present in the current plan. The addresses contain zero when you edit JCL from the long-term plan dialog or when adding an occurrence via the MCP dialog.

MCAUSERF
A user field that lets you allocate resources in the start/stop exit, EQQUX000, that this exit can later use. For example, it is possible to open files for JCL retrieval in the start type call to EQQUX000 instead of opening them each time EQQUX002 is called. IBM Workload Scheduler for z/OS does not use or update this field. The MCAUSERF field is valid when the controller is active.

OCCPTR
The address of occurrence data. The storage at this address is mapped by the PIF segment CPOC.

OPRPTR
The address of operation data. The storage at this address is mapped by the PIF segment CPOP.

WSPTR
The address of workstation data. The storage at this address is mapped by the PIF segment CPWS.

Note: OCCPTR, OPRPTR, and WSPTR always contain valid addresses. The data in these areas is read from the application description and workstation description databases, if the operation associated with the JCL does not exist in the current plan.

AUTHGROU
The name of the authority group.

MEMPRO
The indicator of memory problems.

TASKPTR
The address of the task control block of the caller task.

If the exit needs to access its own files, these files must be opened on the first call for a job (USRAREA value=0) and closed in either of the following ways:

- Before returning control to the scheduler for the last time (before return code 4 is set)
- When an error occurs that does not allow IBM Workload Scheduler for z/OS to acquire further memory, IBM Workload Scheduler for z/OS informs the exit by setting mempro to:

  X'04' If the limit of 608 000 bytes is reached (EQQJ582 issued)

  X'08' If there is not enough storage available (EQQJ577 issued)

  XINFO
  Is the address of the data specified in the Extended Info field of the Current Plan for the corresponding operation. If its value is 0, no extended information is available in the current plan.

  XJNAMLEN
  Is the length that you specify in the Operation Extended Name field of the Current Plan for the corresponding operation. It is a sub-field of the extended information available in the current plan.

USRFNR
The number of user fields records in USRFAREA.

USRFAREA
The address of the user fields area. It is laid out as follows:

USRFAREA

```
USRFNAME DS   CL16         (User field name)
USRFVAL  DS   CL54         (User field value)
```

When the EQQUX002 exit is called to retrieve a job for the first time, the I/O area is 32 000 bytes. If the exit has retrieved the entire job and it fits in the buffer space available, the exit can update the I/O area, return the amount of data in the job, and set a return code 4.

If the exit has not retrieved the entire job, it can update the I/O area, return the amount of data in the job, and set a return code 0 to indicate that there is more data to be returned. The next time the exit is called, the address and the size of the I/O area will be updated because the I/O area is partly used by data from an earlier call. The exit should continue this process until there is no more data to return and then set a return code 4 to indicate that the entire job has been retrieved.

Because the available space in the buffer is reduced for each call, it is possible that the exit must set a return code 44 to indicate that the amount of free space is not enough. When return code 44 is returned, the exit is called again with a job name of eight equal signs (========). This is a reset call. The exit then prepares to process the job from the beginning.

No data can be returned on the reset call. When the exit is called again after the reset call, the I/O area is 32 000 bytes larger than before. This process of returning a “not-enough-space” condition can be repeated up to 19 times for a job. This means that the maximum buffer size that can be requested by the EQQUX002 exit is 608 000 bytes. This corresponds to a job of 7599 card images. When the 608 000 byte limit is reached, IBM Workload Scheduler for z/OS issues message EQQJ582, and the exit is called a 20th time if MEMPRO is set to 4.

The exit can also get more buffer space by using all available space in the current buffer. When this happens and return code 0 is set, the exit is called again with 32 000 bytes free in the buffer. The reset call is not used in this case; the exit should continue processing the current job normally. Extending the buffer in this manner can be continued to a maximum buffer size of 608 000 bytes.

Application-description-feedback exit (EQQUX003)

The application-description-feedback exit (EQQUX003) is called when IBM Workload Scheduler for z/OS is about to update the application-description data set with a new value for the estimated duration of an operation. The exit can change the estimated duration value that has been calculated by IBM Workload Scheduler for z/OS.

The sample library SEQQSAMP that was created during installation contains the EQQUX003 exit, which is a sample of application-description-feedback exits. For information about this library and its samples, see Appendix B, “Sample library (SEQQSAMP),” on page 391.

Installing the exit

The load module implementing the application-description-feedback exit must be link-edited into an APF-authorized library in the LNKLST defined by the DD statement of the current PIF program.

IBM Workload Scheduler for z/OS invokes the exit in AMODE 31; the AMODE parameter specified at link-edit time has no effect.

Interface to the exit

The application-description-feedback exit is invoked in task mode, problem state, and key 8 and the job-step task is APF-authorized. The active task runs with the same access authority as the job-step task. The exit must restore this state before returning to its caller.

Control is passed to the exit using the BAL instruction. The exit must return to its caller using the address and addressing mode passed to it in general register 14.

The exit is entered in AMODE 31 but must switch to AMODE 24 before performing any input or output operations, and then switch back to AMODE 31 before returning to the caller.

When the exit is entered, register 1 contains the address of the parameter list. Each address in this list is used to locate the parameter value. These parameters are passed to the exit:

EQQUX003 parameters

```
ADID      DS   CL16   (Name of current application)
OPIA      DS   CL10   (Input arrival of current operation, YYMMDDHHMM)
OPID      DS   CL6    (Workstation name, operation number (binary))
OLDDUR    DS   F      (Old estimated duration, in minutes)
CURDUR    DS   F      (Duration of current operation, in minutes)
NEWDUR    DS   F      (New estimated duration, in minutes)
```

ADID The name of the application that will be updated.

OPIA The input-arrival date and time of an operation that is described by the current application-description record.

OPID Identifies the current operation further by giving the name of the workstation and the internal operation number of the application.

OLDDUR
The current estimated duration (in minutes), as given by the application description.

CURDUR
The measured duration that the current operation was active at the workstation.

NEWDUR
The new estimated duration that IBM Workload Scheduler for z/OS will save in the application-description record. If this parameter is set, the minimum value is 1, and the maximum value is 5999.

Event-filtering exit (EQQUX004)

EQQUX004 is called when an IBM Workload Scheduler for z/OS event writer is about to write an event to the event data set or, where EWSEQNO is used, add the event to an XCF or NCF queue. In this exit, you can choose to discard events created by JES and SMF exits, or you can indicate that an event that would normally be queued to JCC is not processed by the JCC.

This exit is commonly used to filter the events created by nonproduction work. If you run a significant number of test jobs and other work, and your job naming standards let you do so, consider using EQQUX004 to filter the nonproduction work. The sample library SEQQSAMP that was created during installation contains the EQQUX004 exit, which is a sample of event-filtering exits.

Installing the exit

The load module implementing the event-filtering exit must be link-edited into an APF-authorized library in the LNKLST concatenation or defined by the STEPLIB DD statement in the IBM Workload Scheduler for z/OS JCL procedure. The load module should be link-edited with RMODE(24) according to normal z/OS restrictions.

IBM Workload Scheduler for z/OS invokes the exit in AMODE 24; the AMODE parameter specified at link-edit time has no effect.

Interface to the exit

The event-filtering exit is invoked in task mode, problem state, and key 8 and the job-step task is APF-authorized. The active task runs with the same access authority as the job-step task. The exit must restore this state before returning to its caller.

Control is passed to the exit using the BAL instruction. The exit must return to its caller using the address and addressing mode passed to it in general register 14.

If the exit abends, it is flagged as *not executable*; IBM Workload Scheduler for z/OS does not try to call the exit again.

When the exit is entered, register 1 contains the address of the parameter list. Each address in this list is used to locate the parameter value. These parameters are passed to the exit:

EQQUX004 parameters

```
JOBNAME   DS   CL8    (Name of current job)
RETCODE   DS   F      (Return code)
EXR       DS   CL80   (Exit record)
CDSP      DS   A      (Address of data set information)
CCMACT    DS   CL1    (Catalog management action, blank or D)
```

JOBNAME
: The name of the job for which a job-tracking event has been recognized and for which an event record is about to be written to the event data set.

RETCODE
: Is set by the exit. These values are recognized by the job completion checker:

0 Normal return. The event writer continues normal processing; the event is written to the event data set. If the event is a job-termination event (type 3P), it is passed to the JCC if the JCC is active.

4 No JCC processing for this event. The event is written to the event data set but is not passed to the JCC for processing.

8 This is not a scheduler event. The event is not written to the event data set and is not passed to the JCC for processing. However, if the event is a reader event (type 1) and the job was held by a scheduler job-tracking exit, the job is released from hold by the event writer.

EXR The exit record describing the job-tracking event. This record is built by the SMF or JES exit that recognized the event. The job number offset, EXRJOBID, in the exit record contains JOB as the first three characters if the event is created for a job. EXRJOBID contains STC as the first three characters if the event is created for a started task.

CDSP In earlier releases, the address pointing to a list of data sets that might be eligible for catalog-management actions.

The catalog-management function has been replaced by a new function called restart and cleanup. The CDSP parameter is retained only for compatibility with the exits already written.

CCMACT
The catalog-management-operation action.

This old catalog-management has been completely restructured and substituted with a new function called restart and cleanup, where this parameter is no more applicable. It is kept only for compatibility with the exits already written.

SYSOUT archiving exit (EQQUX005)

EQQUX005 is called by the job completion checker (JCC) during processing of SYSOUT data sets for a job. The exit can be called several times for the same job as the JCC progresses through the various SYSOUT data sets.

This exit is commonly used to change the defined SYSOUT disposition, depending on the success or failure of the job. EQQUX005 is a tracker exit, although the success or failure of an operation is ultimately determined by the controller. The sample library SEQQSAMP that was created during installation contains the EQQX5ASM exit, which is a sample of SYSOUT archiving exits. For more information about this sample, see "SYSOUT archiving exit" on page 396.

Installing the exit

The load module implementing the SYSOUT archiving exit must be link-edited into an APF-authorized library in the LNKLST concatenation or defined by the STEPLIB DD statement in the IBM Workload Scheduler for z/OS JCL procedure. The load module should be link-edited with RMODE(24) according to normal z/OS restrictions.

IBM Workload Scheduler for z/OS invokes the exit in AMODE 24; the AMODE parameter specified at link-edit time has no effect.

Interface to the exit

The SYSOUT archiving exit is invoked in task mode, problem state, and key 8 and the job-step task is APF-authorized. The active task runs with the same access authority as the job-step task. The exit must restore this state before returning to its caller.

Control is passed to the exit using the BAL instruction. The exit must return to its caller using the address and addressing mode passed to it in general register 14.

If the exit abends, it is flagged as *not executable*; IBM Workload Scheduler for z/OS does not try to call the exit again.

When the exit is entered, register 1 contains the address of the parameter list. Each address in this list is used to locate the parameter value. These parameters are passed to the exit:

EQQUX005 parameters

```
CALLTYPE  DS   CL1    (Call type)
RETCODE   DS   F      (Return code)
JOBNAME   DS   CL8    (Name of current job)
JOBNUM    DS   CL8    (Current job number)
CLASS     DS   CL1    (Current SYSOUT class)
SIZE      DS   F      (Size of current SYSOUT record)
RECFM     DS   X      (Current record format)
USERAREA  DS   A      (Address of job related info)
RECORD    DS   X      (Current SYSOUT or event record)
ERRCODE   DS   H      (Current error code)
SYSDISP   DS   CL2    (Job SYSOUT disposition)
```

CALLTYPE
Defines the circumstances under which the exit is called:

B The exit is called to process a SYSOUT record that has not been processed by the JCC because the JCC is skipping forward in the current SYSOUT data set.

C The JCC is ending. The exit is called for the last time.

E There is no more output for the current job. This is the last call for the current job.

F The exit is called because the JCC is starting to process a job that has failed.

I The JCC is starting. The exit is called for the first time.

J The JCC is starting to process a new job. This is the first call for the current job.

N A normal SYSOUT record is passed to the exit.

S The exit is called because the JCC has found an error while processing the current SYSOUT data set.

T The exit is called to process a SYSOUT record that has caused an incident record to be created in the JCC incident log data set.

RETCODE
Is set by the exit. These values are recognized by the JCC:

0 Normal return. The JCC continues normal processing.

4 Stop scanning the current SYSOUT data set. The JCC continues reading the current data set and calls the archiving exit for each record but does not perform other processing for the current data set.

8 Stop calling the archiving exit for this data set. The JCC continues reading the current data set but does not call the archiving exit again. The exit will be called if there are more SYSOUT data sets in the current job.

12 Stop calling the archiving exit. The JCC continues normal processing but does not call the archiving exit again. You must stop the JCC and start it again to reactivate the exit.

JOBNAME
The name of the job that the archiving exit is called for.

JOBNUM
The current job number in the format JOB*nnnnn*, where *nnnnn* is a number.

CLASS
The current SYSOUT class.

SIZE The size of the current SYSOUT record.

RECFM
The current record format defined in the same way as the DCBRECFM field in the mapping of a DCB.

USERAREA
The address containing information about the current job and the current SYSOUT data set in an area mapped as follows:

USERAREA map

```
JOBNAME   DS   CL8    (Job name)
JOBSTART  DS   CL8    (Job start time, format HH.MM.SS)
JOBEND    DS   CL8    (Job end time, format HH.MM.SS)
DATE      DS   CL8    (Current date, format YY/MM/DD)
SYSTEM    DS   CL5    (Name of z/OS system)
TRKID     DS   CL8    (Tracking identification)
STEPNAME  DS   CL8    (Step name from IEF450I)
DSNAME    DS   CL44   (SYSOUT data set name)
SYSCODE   DS   CL4    (System abend code from IEF450I)
USERCODE  DS   CL5    (User abend code from IEF450I)
JOBNUM    DS   CL8    (Job number)
```

RECORD
The current SYSOUT record. SIZE should be used to determine the size of the current record.

ERRCODE
The error code for the current job. In JES2, the error code is nonzero only when the exit is called for a SYSOUT record that has set a nonzero EID after matching a JCC table entry. Subsequent calls for the same job will have ERRCODE set to zero, unless a new EID is set by a subsequent match. When the exit is called is because JCC has started to process a new job.

SYSDISP
The SYSOUT disposition information in effect for the current job. See the description of SYSOUTDISP in the list of JCCOPTS “Parameters” on page 74 for more information about this parameter. The exit can change the SYSOUT disposition for a job at any time, but the normal disposition will be restored when the JCC starts to process the next job.

Incident-record-create exit (EQQUX006)

EQQUX006 is called by the job completion checker to build an incident record when the JCC message table specifies that an incident record should be created for the current SYSOUT record. This exit updates the incident file with the error conditions determined by the JCC incident-logging function.

The sample library SEQQSAMP that was created during installation contains a sample of incident-record-create exits. The sample consists of two members in the sample library:

EQQX6ASM
: Sample EQQUX006

EQQX6JOB
: Sample batch-job JCL skeleton to be used by the sample EQQX6ASM.

Refer to the sample library members themselves for further information.

This sample creates records in the incident data set like these:

Sample incident record

```
86/10/04 * JOBNAMEJ * JOB 5788 * 20.29 * 20.30 * MVS1 * 0001 OK-RC8
  IEF142I JOBNAMEJ AMS8 - STEP WAS EXECUTED - COND CODE 0008
```

The default incident-record-create exit normally generates 2 records for each incident found by the JCC. The first record identifies the job and its start and end times. The second record contains the first 72 characters from the SYSOUT record that caused the incident to be created. You can use the incident-record-create exit if you need to produce incident records with more information or with a different mapping. EQQUX006 is a tracker exit, and the success or failure of an operation is ultimately determined by the controller.

Installing the exit

The load module implementing the incident-record-create exit must be link-edited into an APF-authorized library in the LNKLST concatenation or defined by the STEPLIB DD statement in the IBM Workload Scheduler for z/OS JCL procedure. The load module should be link-edited with RMODE(24) according to normal z/OS restrictions.

IBM Workload Scheduler for z/OS invokes the exit in AMODE 24; the AMODE parameter specified at link-edit time has no effect.

Interface to the exit

The incident-record-create exit is invoked in task mode, problem state, and key 8 and the job-step task is APF-authorized. The active task runs with the same access authority as the job-step task. The exit must restore this state before returning to its caller.

Control is passed to the exit using the BAL instruction. The exit must return to its caller using the address and addressing mode passed to it in general register 14.

If the exit abends, it is flagged as *not executable*; IBM Workload Scheduler for z/OS does not try to call the exit again.

When the exit is entered, register 1 contains the address of the parameter list. Each address in this list is used to locate the parameter value. These parameters are passed to the exit:

EQQUX006 parameters

```
AREA      DS   CL1024 (Record build area)
NRECS     DS   F      (Number of records created)
RSIZE     DS   F      (Incident data set record size)
DATE      DS   CL8    (Current date, format YY/MM/DD)
JOBNAME   DS   CL8    (Job name)
JOBNUM    DS   CL8    (Job number)
JOBSTART  DS   CL8    (Job start time, format HH.MM.SS)
JOBEND    DS   CL8    (Job end time, format HH.MM.SS)
SYSTEM    DS   CL5    (Name of z/OS system)
EID       DS   CL8    (Error code)
TID       DS   CL8    (Tracking identification)
STEPNAME  DS   CL8    (Step name)
RFLAG     DS   CL1    (Record flag)
RECORD    DS   CL72   (Start of current SYSOUT record)
RETCODE   DS   F      (Return code)
```

AREA The record build area to be updated by the exit. This area is blank when the exit is called.

NRECS
Is set by the exit to tell the JCC how many records have been built in the record build area.

RSIZE Gives the size of each record built. The exit must build these records in the build area, each record must be contiguous to the preceding one, and the exit must not update storage outside the build area.

DATE The date of the job that caused the incident-record-create exit to be called.

JOBNAME
The name of the job that caused the incident-record-create exit to be called.

JOBNUM
The job number of the job that caused the incident-record-create exit to be called.

JOBSTART
The start time of the job that caused the incident-record-create exit to be called.

JOBEND
The end time of the job that caused the incident-record-create exit to be called.

SYSTEM
Identifies the system that the job was run on.

EID Contains the error code that is set for this job.

TID Contains tracking information from the current message-table entry.

STEPNAME
Identifies the name of the step in the current job that the exit is called for.

RFLAG
Has the value T if the incident-record-create exit is called because a match has been found in a message table for a SYSOUT record. If RFLAG has any other value, the exit is called for an incident that does not relate to a particular SYSOUT record.

RECORD
Contains the first 72 characters from the current SYSOUT record when RFLAG has the value T.

RETCODE
Is set by the exit. The JCC updates the incident log data set only if this parameter has the value zero.

Operation-status-change exit (EQQUX007)

The operation-status-change exit (EQQUX007) is called whenever an operation in the current plan changes status. The exit is also called when a new operation is added to the current plan by a function other than by daily planning jobs; for example, by PIF or the MCP dialog. The exit is called when the operation is added either to an existing occurrence or as a result of a new occurrence added to the current plan. EQQUX007 is not called for operations that are added at daily planning. The exit can be used to modify the USERDAT field of the OPERAREA parameter. The exit cannot modify other parameters passed to it or other IBM Workload Scheduler for z/OS data or resources. It can examine the parameters and take some action external to IBM Workload Scheduler for z/OS based on the parameter information.

You can use EQQUX007 to:

- Report errors to a problem-management system, such as Information Management
- Generate a write-to-operator message. Such a message could be handled by NetView or by a similar message-processing program to generate alerts or to trigger other processing.

The IBM Workload Scheduler for z/OS sample library that was created during installation contains a sample EQQUX007 exit written in assembler language. This sample exit reports batch-job errors to the Information Management product. The sample consists of two members in the sample library:

EQQX7ASM
Sample EQQUX007 assembler language program and JCL to assemble and link it

EQQX7JOB
Sample batch-job JCL skeleton to be used by the sample EQQUX007.

Refer to the sample library members themselves for further information.

Note: Because the same event might be processed a second time in a recovery situation and during current plan turnover, this must be considered when coding the exit. If invocation of the exit is NOT wanted during IBM Workload Scheduler for z/OS recovery or turnover, code the exit to check if the CALLER is NMM. The only circumstance when NMM calls EQQUX007 is during current-plan recovery and daily-plan turnover.

Installing the exit

The load module implementing the operation-status-change exit must be link-edited into an APF-authorized library in the LNKLST concatenation or defined by the STEPLIB DD statement in the IBM Workload Scheduler for z/OS JCL procedure. If the load module performs any input or output operations it must be link-edited with RMODE(24) according to normal z/OS restrictions. Or it can be link-edited with RMODE(ANY).

IBM Workload Scheduler for z/OS invokes the exit in AMODE 31; the AMODE parameter specified at link-edit time has no effect.

Interface to the exit

The operation-status-change exit is invoked in task mode, problem state, and key 8 and the job-step task is APF-authorized. The active task runs with the same access authority as the job-step task. The exit must restore this state before returning to its caller.

Control is passed to the exit using the BAL instruction. The exit must return to its caller using the address and addressing mode passed to it in general register 14.

The exit is entered in AMODE 31 but must switch to AMODE 24 before performing any input or output operations, and then switch back to AMODE 31 before returning to the caller.

When the exit is entered, register 1 contains the address of the parameter list. Each address in this list is used to locate the parameter value. These parameters are passed to the exit:

EQQUX007 parameters

```
NEWSTAT   DS   CL1   (New operation status)
OLDSTAT   DS   CL1   (Old operation status)
OPNUM     DS   H     (Operation number)
CALLER    DS   CL4   (Caller identification)
ERRCODE   DS   CL4   (Error code)
WSNAME    DS   CL4   (Workstation name)
ADNAME    DS   CL16  (Application name)
OWNER     DS   CL16  (Application owner name)
GROUP     DS   CL8   (Authority group name)
JOBAREA   DS   A     (Address of job-related data)
OPERAREA  DS   A     (Address of operation-related data)
USRAREA   DS   A     (User-defined field)
EXSTAT    DS   CL1   (Operation extended status)
OCCPTR    DS   A     (Address of occurrence data)
OPRPTR    DS   A     (Address of operation data)
UFNUM     DS   F     (Number of user fields)
UFPTR     DS   A     (User fields area address)
```

NEWSTAT
Defines the new status for the current operation. These values are possible:

A	Arrived at the workstation
C	Complete
E	Ended with errors
I	Interrupted
R	Ready for processing
*****	Ready for processing (predecessor at nonreporting workstation complete)
S	Active (started)
W	Waiting.

OLDSTAT
: Defines the previous status for the current operation. The same values are possible as for the new status, plus blank. Blank means that the operation has been added to the current plan by a function other than by daily-planning jobs. No call is made to EQQUX007 when operations are added by daily-planning jobs.

OPNUM
: Is the operation number of the current operation.

CALLER
: Identifies the function within IBM Workload Scheduler for z/OS that called the exit. These values are possible:

AR	Automatic-recovery task
EM	Event-manager task
GS	General-service task, but not modify-current-plan
MCP	Modify-current-plan function in the general-service task
NMM	Normal-mode-manager task
WSA	Workstation-analyzer task.

ERRCODE
: The error code for the current operation if the new status is E.

WSNAME
: The name of the workstation where the current operation is active or will become active.

ADNAME
: The application name for the current operation.

OWNER
: The name of the owner of the current application.

GROUP
: The name of the authority group that the current operation belongs to.

JOBAREA
: The address of an area that contains information about status changes that are related to a specific job. The job-related area is laid out as follows:

JOBAREA

```
JOBNAME  DS  CL8  (Job name)
JOBNUM   DS  CL8  (Job number)
DATE     DS  CL8  (Current date, format YY/MM/DD)
JOBSTART DS  CL8  (Job start time, format HH.MM.SS)
JOBEND   DS  CL8  (Job end time, format HH.MM.SS)
STEPNAME DS  CL8  (Step name)
ABCODE   DS  CL4  (System abend code, format Shhh)
USRCODE  DS  CL5  (User abend code, format Uhhh)
ORIGNJE  DS  CL8  (Name of origin NJE node)
PSTEPNAM DS  CL8  (Procedure step name)
```

The JOBNAME and DATE fields are always present for status changes at automatically reporting computer and printer workstations. JOBNUM is present when an operation at one of these workstations changes its status from **S** to **C**. When the job is submitted by a fault-tolerant workstation, the first three characters of the JOBNUM value are set to **UNX** by IBM Workload Scheduler for z/OS.

The status is changed to S (started). The JOBNUM value can change only if the operation is rerun. JOBSTART is present when the new status at automatically reporting computer and printer workstations is S (started), C (complete), E (ended-in-error), or I (interrupted). JOBEND is present when the new status at one of these workstations is C or E.

STEPNAME, PSTEPNAM, ABCODE, and USRCODE are present only for jobs that have abended during running. ORIGNJE is present for processing operations when the new status is C or E.

When the status of an operation is changed from C or E to S, C, E, or I, certain JOBAREA fields are set to their previous values for that operation.

Fields in the job-related area are blank if information is not available when the exit is called.

OPERAREA
The address of an area that contains information about the operation whose status is being changed. This area is mapped as follows:

OPERAREA

```
OPERTEXT DS CL24 (Operation description)
APPLIA   DS CL10 (Application input arrival,format YYMMDDHHMM)
OPERIA   DS CL10 (Operation input arrival, format YYMMDDHHMM)
PSTART   DS CL10 (Planned start, format YYMMDDHHMM)
PLEND    DS CL10 (Planned end, format YYMMDDHHMM)
DEADLINE DS CL10 (Operation deadline, format YYMMDDHHMM)
USERDAT  DS CL16 (User data field)
RSPRES   DS CL1  (Y=there is reason data, N=no reason data)
RSERR    DS CL4  (The error code set by the dialog user)
RSUSER   DS CL16 (The user data from the dialog: this is
*                 the same as USERDAT at entry to exit)
RSREASN  DS CL300(The dialog 'reason for restart' data)
RSPANEL  DS CL8  (The panel where the restart was initiated)
XJNAME   DS CL54 (Job extended name)
SAWS     DS CL1  (New automation work station)
```

USRAREA
A user field that is also passed to the EQQUX000 exit. It contains valid data only if you have an EQQUX000 exit that places some data in it. IBM Workload Scheduler for z/OS does not use or update this field.

EXSTAT
Defines the extended status code of the operation. Refer to *IBM Workload Scheduler for z/OS Managing the Workload* for a description of the valid codes.

For performance reasons, user exit EQQUX007 does not currently provide the extended status 'X' (waiting for resource). A new value, 'Z', valid for all current status codes and types of workstation, has been added to signal that an error has occurred in the DOA updating process. In this case, message EQQE106I is issued in EQQMLOG.

OCCPTR
The address of the common data of record CPLREC3C.

OPRPTR
The address of the common data of record CPLREC3P.

USRFNR
The number of user field records in USRFAREA.

USRFAREA
The address of the user field area. It is laid out as follows:

USRFAREA

```
USRFNAME DS   CL16         (User field name)
USRFVAL  DS   CL54         (User field value)
```

Operation-initiation exit (EQQUX009)

The operation-initiation exit (EQQUX009) is called when an operation is ready to be started on a workstation which specifies a user-defined destination ID.

You can use EQQUX009 to communicate with operating environments that do not support a Tivoli OPC tracker. The IBM Workload Scheduler for z/OS sample library that was created during installation contains these sample programs to demonstrate how you can use EQQUX009:

EQQUX9N
Sample EQQUX009 using NJE to communicate with VM

EQQX9OS2
Sample EQQUX009 using TCP/IP to communicate with OS/2

EQQX9AIX
Sample EQQUX009 using TCP/IP to communicate with AIX®.

See Appendix B, "Sample library (SEQQSAMP)," on page 391 for more information.

Installing the exit

The load module implementing the operation-initiation exit must be link-edited into an APF-authorized library in the LNKLST concatenation or defined by the STEPLIB DD statement in the IBM Workload Scheduler for z/OS JCL procedure. If the load module performs any input or output operations it must be link-edited with RMODE(24) according to normal z/OS restrictions. Or it can be link-edited with RMODE(ANY).

IBM Workload Scheduler for z/OS invokes the exit in AMODE 31; the AMODE parameter specified at link-edit time has no effect.

Interface to the exit

The operation-initiation exit is invoked in task mode, problem state, and key 8 and the job-step task is APF-authorized. The active task runs with the same access authority as the job-step task. The exit must restore this state before returning to its caller.

Control is passed to the exit using the BAL instruction. The exit must return to its caller using the address and addressing mode passed to it in general register 14.

The exit is entered in AMODE 31 but must switch to AMODE 24 before performing any input or output operations, and then switch back to AMODE 31 before returning to the caller.

If the exit abends, it is flagged as *not executable*; IBM Workload Scheduler for z/OS does not try to call the exit again.

When the exit is entered, register 1 contains the address of the parameter list. Each address in this list is used to locate the parameter value. These parameters are passed to the exit:

EQQUX009 parameters

```
DEST      DS  CL8  (The user-defined destination ID)
MCAUSERF  DS  A    (Address set by the user in the EQQUX000 exit)
OPCTOKEN  DS  F    (Token used to uniquely identify the operation)
WSNAME    DS  CL4  (Workstation name)
ADID      DS  CL16 (Application name)
IA        DS  CL10 (Input arrival date and time, format YYMMDDHHMM)
OPNUM     DS  CL3  (Operation number)
JOBNAME   DS  CL8  (Job name)
AREALEN   DS  F    (Length of the data area passed, zero if no data)
DATAP     DS  A    (Address of the data passed from JBLIB/JSFILE)
RETC      DS  F    (Return code set by the exit)
```

DEST Defines the user-defined destination ID from the workstation of the current operation.

MCAUSERF
Is a user field that lets you allocate resources in the start/stop exit, EQQUX000, that this exit can later use. This field contains the value set by the EQQUX000 exit. IBM Workload Scheduler for z/OS does not use or update this field.

OPCTOKEN
Contains the token assigned for the current operation. This should be stored by you and used as input when you invoke OPSTAT to uniquely identify an operation.

WSNAME
Identifies the workstation name for the current operation.

ADID The application name for the current operation.

IA Contains the input arrival date and time for the current operation in the format YYMMDDHHMM.

OPNUM
The operation number of the current operation.

JOBNAME
The job name defined for the current operation

AREALEN
Contains the length of the data area pointed to by DATAP. If the length is zero, no data has been passed.

DATAP
The address of an area that contains the data passed to the exit from the JCL library or the job-setup file. The data can have any format; it would probably not be z/OS JCL.

RETC The return code set by the exit. These values are valid:

0 Normal return, IBM Workload Scheduler for z/OS processing continues.

4 Operation failed. IBM Workload Scheduler for z/OS will take the action specified by the SUBFAILACTION keyword of the JTOPTS statement.

8 Communication failure. IBM Workload Scheduler for z/OS will automatically generate an offline event for all workstations connected to this destination. The exit will not be called again for this destination until the workstation status is reported as active.

The status of the operation that the exit was processing is set according to the SUBFAILACTION keyword.

If the exit returns a value in RETC that is not considered valid, it will be ignored, and IBM Workload Scheduler for z/OS will treat the RETC as if 0 had been returned. In this case message EQQF010I is issued to the controller message log.

If the exit abends, it is flagged as *not executable*, and message EQQF011 is issued. The status of the operation that the exit was processing is set according to the SUBFAILACTION keyword.

Job-tracking log write exit (EQQUX011)

EQQUX011 is called just before IBM Workload Scheduler for z/OS is going to write a job-tracking log record. You can use this exit to set up a ***disaster recovery*** procedure, maintaining online remote job-tracking logs in a warm backup secondary data center.

The sample library SEQQSAMP that was created during installation contains the EQQUX011 exit, which is a sample of job-tracking log write exits. For information about this library and its samples, see Appendix B, "Sample library (SEQQSAMP)," on page 391.

Installing the exit

The load module implementing the Job-tracking log write exit must be link-edited into an APF-authorized library in the LNKLST concatenation or defined by the STEPLIB DD statement in the controller JCL procedure.

The load module can be link-edited with RMODE(24) or RMODE(ANY). The load module must be link-edited with at least reusable attribute.

IBM Workload Scheduler for z/OS invokes the exit in AMODE 31; the AMODE parameter specified at link-edit time has no effect.

Interface to the exit

The Job-tracking log write exit is invoked in task mode, problem state, key 8; the job-step task is APF-authorized. The active task runs with the same access authority as the job-step task. The exit restores this state before returning to its caller.

Control is passed to the exit using the BAL instruction. The exit returns to the caller using the address and addressing mode passed to it in general register 14.

If the exit abends, it is flagged as ***not executable***; IBM Workload Scheduler for z/OS does not try to call the exit again.

When the exit is entered, register 1 contains the address of the parameter list. Each address in this list is used to locate the parameter value. These parameters are passed to the exit:

EQQUX011 parameters

```
MCAUSERF  DS   A       (User field)
JTLOGNUM  DS   F       (Current job-tracking log number)
SIZE      DS   F       (Size of current job-tracking log record)
```

```
TRL       DS   F        (Address of job-tracking log record)
TQTOT     DS   F        (Job-tracking log records since CP backup)
CPLCR     DS   CL10     (Current Plan Creation date and time)
CPLEND    DS   CL10     (DP PLAN PERIOD END)
BUGMT     DS   XL8      (Last CP backup date and time - GMT)
GMTOF     DS   F        (GMT offset minutes)
RETCODE   DS   F        (Return code)
```

MCAUSERF
: A user field that is also passed to the EQQUX000 exit. IBM Workload Scheduler for z/OS does not use or update the MCAUSERF field.

JTLOGNUM
: The job-tracking log number to which a job-tracking log record is about to be written.

SIZE The size, in bytes, of the job-tracking log record that is about to be written.

TRL The address of the job-tracking log record.

TQTOT
: The total number of job-tracking log records (current job-tracking log record included) written since last CP backup.

CPLCR
: The Current Plan Creation date and time in local time. It is in YYMMDDHHMM format. YY is the IBM Workload Scheduler for z/OS internal year representation - 00 is 1972, 24 is 1996 and 99 is 2071.

CPLEND
: The Daily Planning period end date and time in local time. It is in YYMMDDHHMM format. YY is the IBM Workload Scheduler for z/OS internal year representation - 00 is 1972, 24 is 1996 and 99 is 2071.

BUGMT
: Is the last CP backup date and time in GMT. It is in 0nYYDDDFHHMMSSTH format. If n = 0, year is 19YY. If n = 1, year is 20YY.

GMTOF
: Is the GMT offset in minutes.

RETCODE
: Is set by the exit. The following values are recognized:

  0 Normal return. Processing continues.

  8 Stop calling the job-tracking log record write exit. You must stop the controller and start it again to reactivate the exit.

  All other return codes are processed as for return code 8.

Notes:

1. The scheduler performance might be degraded if the exit requires lengthy processing time.
2. While this exit has control, the scheduler does not process any new requests requiring update of the job-tracking log.
3. System waits, including implied waits for I/O operations, should be avoided.
4. The scheduler Job-Tracking log writes is done by several different tasks, therefore the exit will be called from different tasks interchangeably.
5. The above remarks should be taken into account when designing the exit.

Job-tailoring event exit (EQQUX013)

EQQUX013 is called when IBM Workload Scheduler for z/OS is about to submit a batch job or start a started task. Use this exit to prevent some jobs from being tailored with the //TIVDSTXX OUTPUT statements to produce an additional copy of the JESDS data sets for data store processing. The sample library SEQQSAMP that was created during installation contains the EQQUX013 exit, which is a sample of the job-tailoring prevention exit. For information about this library and its samples, see Appendix B, "Sample library (SEQQSAMP)," on page 391.

Installing the exit

The load module implementing the job-tailoring prevention exit must be link-edited into an APF-authorized library in the LNKLST concatenation or to a library defined by the STEPLIB DD statement in the IBM Workload Scheduler for z/OS JCL procedure.

The load module must be link-edited with RMODE(ANY) and AMODE(31) attributes.

Interface to the exit

The job-tailoring prevention exit is invoked in task mode, problem state, and key 8 and the job-step task is APF-authorized. The active task runs with the same access authority as the job-step task. The exit must restore this state before returning to its caller.

IBM Workload Scheduler for z/OS invokes the exit in the addressing mode defined by the load module's AMODE attribute, that is 31 bit addressing mode, therefore the job stream passed to the exit resides above the 16M line.

Control is passed to the exit using the BASSM instruction. The exit must return to its caller using the address passed to it in general register 14. The exit can return in any addressing mode.

When the exit is entered, register 1 contains the address of the parameter list. Each address in this list is used to locate the parameter value. These parameters are passed to the exit:

EQQUX013 parameters

```
JOBNAME  DS CL8 (Job name)
JCLLEN   DS F (Size, in bytes, of current job stream)
JCLAREA  DS n * CL80 (All records in job stream)
LATEOUT  DS CL10     (Latest start, format YYMMDDHHMM)
ESTDUR   DS CL4      (Estimated duration, format HHMM)
NUMPS    DS H        (Number of parallel servers required)
NUMR1    DS H      (Amount required of workstation resource 1)
NUMR2    DS H      (Amount required of workstation resource 2)
SPECRES  DS CL8      (First 8 characters of first sr name)
ADID     DS CL16     (Name of current application)
MCAUSERF DS A        (User field)
GROUP    DS CL8      (Name of current authority group)
RUSER    DS CL8      (Name of RACF user submitting the job)
OPERTYPE DS CL1      (J or S, for job or started task)
UPDAT    DS C        (Y or N, defines job origin)
JCLUSER  DS CL8      (Last user updating this job)
JCLUTIME DS CL10     (Time or last update, format YYMMDDHHMM)
OPNUM    DS F        (Operation number)
IATIME   DS CL10   (Occurrence input-arrival time, YYMMDDHHMM)
OWNER    DS CL16    (Application owner name)
```

```
SPECNR   DS H        (Number of special resources)
SPECBUF  DS A        (Special resource buffer)
WSNAME   DS CL4      (Operation workstation name)
XINFO    DS A        (Extended information address)
XJNAMLEN DS F        (Extended job name length)
WSCHENV  DS CL 16    (Scheduling Environment Name)
CLEANUPT DS CL 1     (Clean-up type)
RETCO    DS CL4      (Operation error code)
```

Note:

1. To allow more flexibility in determining the jobs that are not to be tailored with the //TIVDSTxx OUTPUT statements, the JCL of the job being submitted is provided as input to this exit. However, to work successfully, the exit must not modify this JCL.
2. A job is not tailored for data store processing, if the EQQUX013 exit returns a return code 0012 to the Work Station Analyzer.

JOBNAME
Name of the job that is about to be submitted.

JCLLEN
Size, in bytes, of the job.

JCLAREA
JCL records in the job.

LATEOUT
The latest-start-time value that IBM Workload Scheduler for z/OS has calculated for the job.

ESTDUR
The estimated duration of this job

NUMPS
The number of parallel servers required

NUMR1
The amount of workstation resource 1 required

NUMR2
The amount of workstation resource 2 required

SPECRES
The first 8 characters of the special resource name

ADID Name of the application of which the job is part.

MCAUSERF
A user field that is also passed to the EQQUX000 exit. IBM Workload Scheduler for z/OS does not use the MCAUSERF field.

RUSER
The name of the RACF user that owns the job

OPERTYPE
Has the value J for job or S for started task.

UPDAT
Has the value Y if the current job was retrieved from the EQQJSnDS data set. In all other cases, UPDAT has the value N.

GROUP
The name of the authority group that the current operation belongs to

JCLUSER
: Name of the last TSO user who updated the current job. This parameter is meaningful only if UPDAT is Y.

JCLUTIME
: The date and time of the last update to the current job. This parameter is meaningful only if UPDAT is Y.

OPNUM
: Operation number of the operation representing this job

IATIME
: Input-arrival time of the application occurrence to which this job belongs

OWNER
: The name of the owner of the current application

SPECNR
: The number of special resource names in SPECBUF

SPECBUF
: An address to a buffer that contains a number of 64-byte fields. The number of 64-byte fields in the buffer is indicated by SPECNR. The first 44 bytes of each field contain the name of the special resource. The last 2 bytes of each field are reserved for future use.

WSNAME
: Name of the operation workstation.

XINFO
: The address of the data specified in the Extended Info field o the Current Plan for the corresponding operation. If its value is 0, no extended information is available in the current plan.

XJNAMLEN
: The length that you specify in the Operation Extended Name field of the Current Plan for the corresponding operation. It is a sub-field of the extended information available in the current plan.

WSCHENV
: The scheduling environment name currently stored in the CP operation record. This value can be modified by the exit.

RETCO
: Name of a field that is used by the user exit to prevent the jobs from being tailored with the //TIVDSTxx OUTPUT statements to generate additional copies of the JESDS data sets for data store processing.

Time-dependent-operation exit (EQQUX014)

The controller calls EQQUX014 when a time dependent operation becomes ready in a z/OS environment. It is not applicable to operations scheduled on distributed workstations in an end-to-end with fault tolerance capabilities environment. The exit returns an offset, expressed in minutes, to be added to the operation start time. The result is used by the Workstation Analyzer task, to decide whether the operation can be started.

The sample library SEQQSAMP contains a sample of EQQUX014 exit. It describes the exit logic, based on a criteria table referenced by the UX14IN DD name in the JCL of the controller started task.

If you need to modify the criteria table while the controller is running, you must place the table in a member of a *partitioned* data set and identify it in the controller started task as follows:

```
//UX14IN    DD  DISP=SHR,DSN=TWSDEV.UX014.TABLE(UX14TAB)
```

Then, to dynamically refresh the criteria table, issue the following modify command:

```
/F subsys,RFRUX14T
```

For detailed information about the `RFRUX14T` command, see how to modify the scheduler in *Managing the Workload*.

Note: If you do not need to edit the criteria table while the controller is running, you can leave it in a *sequential* data set.

Use the table to correlate workstation name and offset value. Optionally, you can specify filtering rules to exclude selected operations from the process that adds the offset value to the operation start time. The following layout defines column ranges to place key data in the UX14IN file:

1-2 Data type. Specify one the following:

WS	To correlate workstation name and offset value.
AD	To exclude an operation by application name.
IA	To exclude an operation by input arrival value.
OP	To exclude an operation by operation number.
JN	To exclude an operation by job name.

3 A blank character (filler).

4-19 One of the following:
- *Workstation name* (up to 4 characters) for WS data type. Wildcard character * is supported.
- *Application name* (up to 16 characters) for AD data type. Wildcard character * is supported.
- *Input arrival* (up to 10 characters) for IA data type. Use the format YYMMDDHHMM.
- *Operation number* (up to 3 characters) for OP data type.
- *Job name* (up to 8 characters) for JN data type. Wildcard character * is supported.

20 Sign (+ or -) for WS data type only.

21-24 Offset value for WS data type only.

When editing the UX14IN file, specify the data types in the following specific order:
- All the rows with WS data type first.
- Then all the rows with AD data type (if any).
- Then all the rows with IA data type (if any).
- Then all the rows with OP data type (if any).
- Then all the rows with JN data type (if any).

You can add comment text by using the `/*` string.

See also “Example” on page 251 section. For further details, refer to the prologue of the exit.

The sample library SEQQSAMP that was created during installation contains a sample of EQQUX014 exit. For information about this library and its samples, see Appendix B, “Sample library (SEQQSAMP),” on page 391.

Installing the exit

The load module implementing this exit must be link-edited into an APF-authorized library in the LNKLST concatenation or defined by the STEPLIB DD statement in the controller JCL procedure.

The load module performs input or output operations, therefore must be link-edited with RMODE(24) according to normal z/OS restrictions.

Interface to the exit

When the exit is entered, register 1 contains the address of the parameter list. Each address in this list is used to locate the parameter value. These parameters are passed to the exit:

EQQUX014 parameters

```
MCAUSERF  DS  A     (User field)
TABPTR    DS  A     (Criteria table address)
NUMROW    DS  F     (Number of rows of the criteria file)
RECSIZE   DS  F     (Criteria file record size)
ADID      DS  CL16  (Operation application name)
IATIME    DS  CL10  (Operation input arrival time, format YYMMDDHHSS)
WSNAME    DS  CL4   (Operation workstation name)
JOBNAME   DS  CL8   (Operation job name)
OPNUM     DS  F     (Operation number)
TOFFS     DS  F     (Offset to be added  to, or subtracted from, the
                     operation start time)
RCODE     DS  F     (Return code)
```

MCAUSERF
User field that is also passed to the EQQUX000 exit. The scheduler for z/OS does not use or update this field.

TABPTR
User field containing the address of the criteria table used by this exit. It is not allocated by the scheduler. The exit must return a value that is stored in controller storage and passed back to the exit at each new call. According to the provided sample, the first time the exit is called TABPTR must be set to zero, next times it contains the address of the criteria table to be used by the exit.

NUMROW
User field containing the number of rows of the input criteria file. The first time the exit is called, the exit must set and return this value, that is stored in controller storage and passed back to the exit at each new call.

RECSIZE
User field containing the record size of the input criteria file. The first time the exit is called, the exit must set and return this value, that is stored in controller storage and passed back to the exit at each new call.

ADID The name of the application that the job becoming ready is part of.

IATIME
The input arrival time of the job becoming ready.

WSNAME
The name of the workstation defined for the job becoming ready.

JOBNAME
The name of the job becoming ready.

OPNUM
The operation number of the job becoming ready.

TOFFS
Sign and value of the offset expressed in minutes, to be added to the job start time. This parameter is used by the Workstation Analyzer task, to decide whether the operation can be started or not. The controller uses the returned offset to update the job start time only. The process leaves unchanged the latest start time.

Sample member EQQUX014 uses a criteria table to calculate this offset, however you can set it using a different method.

RCODE
The return code set by the exit. A value different from 0 indicates that the scheduler deactivated the exit.

Example

Suppose that:

- CPU1 is a workstation corresponding to Rome that is the location where the controller runs.
- CPU2 is a workstation corresponding to a branch location in Istanbul.
- There are more workstations Z*, corresponding to different branch locations in London.

To specify the offset information and your selection criteria, define the following data in the UX14IN data set:

```
EDIT       TWSTST.TWSIDD1.UX014                            Columns 00001 00072
Command ===>                                                   Scroll ===> CSR
=COLS> ----+----1----+----2----+----3----+----4----+----5----+----6----+----7--
****** ***************************** Top of Data ******************************
000101 WS CPU2           +0060
000102 WS Z*             -0060
000103 /* TO EXCLUDE SPECIFIC OPERATIONS
000110 AD MYAPPLID
000120 JN JOBX123
000200 JN MYJOB
****** **************************** Bottom of Data ****************************
```

When the operation is ready to start, the controller calls the exit and assigns the specified offsets to any operation associated to CPU2 or Z*, unless:

- The operation belongs to MYAPPLID.
- The operation is defined with job name JOBX123 or MYJOB.

Limitations

The controller uses the returned TOFFS value to update the job start time and leaves unchanged the latest start time. This might affect latest start time-related activities such as:

- The processing of the SUPPRESS IF LATE option.
- Alert conditions or critical jobs monitoring. For example, consider a job with 9:00 as start time and 9:30 as latest start time. Supposing that the exit returns +0060 as TOFFS, the job becomes late as soon as it is ready to start.

EQQDELDS/EQQCLEAN Catalog exit (EQQUXCAT)

EQQUXCAT is called by the sample EQQDELDS or EQQCLEAN program before performing any simple cleanup action. Because in some cases EQQCLEAN might delete a data set by mistake, it is recommended that you protect critical data sets from deletion by using either the RCLOPTS parameters (DDPROT, DDPRMEM, DSNPROT, DSNPRMEM) or the EQQUXCAT exit. EQQDELDS and EQQCLEAN do not perform the action when the return code is set to a nonzero value.

The sample library SEQQSAMP that was created during installation contains the EQQUXCAT exit, which is a sample of EQQDELDS/EQQCLEAN exits. For information about this library and its samples, see Appendix B, “Sample library (SEQQSAMP),” on page 391.

Installing the exit

The load module implementing the exit must be link-edited into an APF-authorized library visible to the EQQDELDS sample (for example in the STEPLIB DD statement of the EQQDELDS procedure). You should use attributes RMODE(24) and AMODE(31).

Interface to the exit

The exit is invoked in task mode, problem state, and key 8; the task is APF-authorized. The exit must restore this state before returning to its caller.

Control is passed to the exit using the BAL instruction. The exit must return to the caller using the address and addressing mode passed to it in general register 14.

When the exit is entered, register 1 contains the address of the parameter list. Each address in this list is used to locate the parameter value. These parameters are passed to the exit:

EQQUXCAT parameters

```
PDSNAME   DS   CL44     (data set name)
PMIGR     DS   CL1      (Y=MIGRATED data set N=otherwise)
PVSAM     DS   CL1      (Y=VSAM data set N=otherwise)
PTAPE     DS   CL1      (Y=tape N=otherwise)
PRETCOD   DS   F        (return code)
```

Note: If a return code different from 0 is returned, the data set action will not be executed.

EQQCLEAN GDG Resolution exit (EQQUXGDG)

About this task

EQQUXGDG is called by the EQQCLEAN program just before executing any single GDG overwrite action into JES control blocks. You can use this exit to prevent GDG overwriting, that is to avoid the substitution of a relative number with a GDG absolute format (GDGRoot.GnnnnVnn). EQQCLEAN will not execute the action when the EQQUXGDG return code is set to a value different than 0. Note that EQQUXGDG can only prevent the simulation of a GDG, not its deletion. To prevent GDG deletion, use the EQQUXCAT exit or the DDPROT, DSNPROT RCLOPTS statements. For this reason, if you want to coordinate deletion and simulation actions (such as: not simulating data sets that have been protected from deletion), you should:

- Use the same logic for EQQUXCAT and EQQUXGDG.
- Add to EQQUXGDG the logic to read tables containing the DDPROT and DSNPROT names list so that EQQUXGDG can decide to not simulate a GDG if its DD or DSN is found in these tables.

For example, suppose you run a JCL with the following step:

```
//STEP1  EXEC PGM=MYPGM
//DDPRO1 DD DSN=MYGDG.TEST(+1),DISP=(NEW,CATLG)
```

The first run allocated data set MYGDG.TEST.G0015V00.

If you want to rerun this JCL, saving the previously allocated GDG G00015V00 and allocating a new generation G00016V00, you can:

1. Define DDPRO1 in the DDPROT RCLOPTS controller statement.
2. Customize EQQUXGDG in order to not simulate the data set having DDNAME equal to DDPROT.

Alternatively, you can:

1. Customize EQQUXCAT to not delete the data set starting with MYGDG.TEST and DD name DDPROT.
2. Customize EQQUXGDG to not simulate the data set starting with MYGDG.TEST and DD name DDPROT.

The sample library SEQQSAMP that was created during installation contains the EQQUXGDG exit, which is a sample of EQQCLEAN exits. For information about this library and its samples, see Appendix B, “Sample library (SEQQSAMP),” on page 391.

DDPROT/DSNPROT interactions

You can protect a data set from deletion by setting the DDPROT and DSNPROT options for the RCLOPTS statement in the controller. You must however consider the following interactions these have with the EQQUXCAT and EQQUXGDG exits:

- EQQUXCAT (deletion prevention exit) and EQQUXGDG (simulation prevention exit) become active as soon as they are available to the EQQCLEAN program in the library. On the contrary, DDPROT and DSNPROT become active only after a Controller Restart or an appropriate MODIFY command are issued. For this reason, it may happen that for some jobs the EQQCLEAN exits apply a logic based on DDPROT/DSNPROT naming that differs from the normal logic of RCLOPTS.
- The prevention logic of RCLOPTS DDPROT/DSNPROT is applied at the controller level, so that by the time the EQQCLEAN exits are to be executed, the controller has already eliminated the protected data set from the action list. At this point, if the EQQCLEAN exits are called, EQQUXCAT can only protect from deletion other data sets, but cannot modify the already protected ones.

Installing the exit

The load module implementing the exit must be link-edited into an APF-authorized library visible to the EQQCLEAN program (for example in the STEPLIB DD statement of the EQQCLEAN procedure). You should use attributes RMODE(24) and AMODE(31).

Interface to the exit

The exit is invoked in task mode, problem state, and key 8; the task is APF-authorized. The exit must restore this state before returning to its caller.

Control is passed to the exit using the BAL instruction. The exit must return to the caller using the address and addressing mode passed to it in general register 14.

When the exit is entered, register 1 contains the address of the parameter list. Each address in this list is used to locate the parameter value. These parameters are passed to the exit:

EQQUXGDG parameters

```
PJOBNAM    DS   CL8   (job name)
PSTEPNUM   DS   CL3   (step number)
PSTEPNAM   DS   CL8   (step name)
PPROCSTE   DS   CL8   (proc step name)
PDDNAME    DS   CL8   (DD name)
PROOT      DS   CL35  (GDG root)
PABSOLUT   DS   CL8   (GDG absolute value)
PSIGN      DS   CL1   (GDG relative number sign)
PRELNUM    DS   CL7   (GDG relative number)
PRETCODE   DS   F     (return code)
```

PJOBNAM
: The job name of the JCL that is going to be run.

PSTEPNUM
: The step number (expressed in character format) where the GDG data set to be overwritten is located.

PSTEPNAM
: The step name where the GDG data set to be overwritten is located.

PPROCSTE
: The proc step name where the GDG data set to be overwritten is located.

PDDNAME
: The DD name where the GDG data set to be overwritten is located.

PROOT
: The GDG Root o f the GDG data set that is going to be overwritten.

PABSOLUT
: The absolute value (GnnnnVnn) that is going to be used to overwrite the input GDG data set.

PSIGN
: The sign (+ or – or blank) of the relative number of the GDG data set to be overwritten.

PRELNUM
: The relative number (expressed in character format) of the GDG data set to be overwritten.

PRETCOD
: The return code set by the exit: 0 means execute the overwriting, 4 do not execute overwriting.

JCL-imbed exit

You can use the JCL-imbed exit to include JCL in the current job stream at either job setup or job submission. The exit is invoked by the FETCH directive on the //*%OPC JCL statement. JCL handling and IBM Workload Scheduler for z/OS directives are described in more detail in *Managing the Workload*

Installing the exit

The load module implementing the JCL-imbed exit must be link-edited into an APF-authorized library in the LNKLST concatenation or defined by the STEPLIB DD statement in the IBM Workload Scheduler for z/OS started task.

The load module can be link-edited with either the AMODE 24 or AMODE 31 attribute. IBM Workload Scheduler for z/OS invokes the exit in the addressing mode defined by the exit load module.

Interface to the exit

The JCL-imbed exit is invoked in task mode, problem state, and key 8. The job-step task is APF-authorized. The active task runs with the same access authority as the job-step task. The exit must restore this state before returning to its caller.

The exit is called using the BASSM instruction and must return to its caller using the address and addressing mode passed to it in general register 14. The recommended method is to restore all registers to their values on entry to the exit and return to the caller using a BSM 0,14 instruction.

If the exit is entered in AMODE 31, it must switch to AMODE 24 before performing any input or output operations, and then switch back to AMODE 31 before returning to the caller.

If the exit abends, it is flagged as *not executable*, and message EQQJ616E is issued. IBM Workload Scheduler for z/OS does not try to call the exit again.

Parameters passed to the exit reside below the 16 MB line. Also, addresses passed to the exit are addresses of areas below the 16 MB line.

When the exit is entered, register 1 contains the address of the parameter list. Each address in this list is used to locate the parameter value. These parameters are passed to this exit:

JCL-imbed exit parameters

```
JOBNAME    DS  CL8   (Job name)
IOAREA     DS  A     (Address of JCL buffer area)
IOAREAL    DS  F     (Size of JCL buffer area)
DATAL      DS  F     (Amount of data returned)
ADID       DS  CL16  (Name of current application)
WSDP       DS  A     (Address of workstation data)
OCCP       DS  A     (Address of occurrence data)
OPRP       DS  A     (Address of operation data)
WEEKD      DS  CL1   (Day of week: 1-7, where 1 is Monday)
WEEKY      DS  CL2   (Week of year)
RC         DS  F     (Return code)
MCAUSERF   DS  A     (Address set by the user in the EQQUX000 exit)
```

JOBNAME
The name of the job that is to be submitted.

IOAREA
: The address of a buffer, allocated by IBM Workload Scheduler for z/OS, where JCL records for the current job must be placed. JCL records must be placed consecutively in this buffer with no intervening space between records.

IOAREAL
: The amount of space, in bytes, in the IOAREA buffer. In the current implementation, this value is always 32768 for each call to the exit. This is enough space to return up to 409 JCL-card images for each call to the exit.

DATAL
: The amount of data returned by the exit. This cannot be a negative value or a value greater than the IOAREAL value. A zero value is valid on the last call.

ADID The name of the current application.

WSDP
: The address of the workstation data. The storage at this address is mapped by the program-interface (PIF) segment WSCOM.

OCCP The address of the occurrence data. The storage at this address is mapped by the PIF segment CPOC.

OPRP The address of the operation data. The storage at this address is mapped by the PIF segment CPOP.

WEEKD
: The day of the week when the exit is called. It is a numeric value from 1 to 7, where 1 is Monday.

WEEKY
: The number of the current week in the current year. It is a numeric value from 1 to 53, and is calculated according to the international standard ISO 8601. This standard uses a cycle of five "week-in-year" values to determine in which week the first of January falls. The cycle is 53, 52, 1, 1, 1. The current cycle started in 1988. That is, the first of January 1988 was in week 53 of 1987. The first of January 1989 was in week 52 of 1988, and the first of January 1990 was in week 1 of 1990.

RC The return code set by the exit. These values are valid:

- **0** Normal return. The amount of data returned by the last call to the exit is given in DATAL. The exit will be called again to return more JCL records for the current job. The total amount of data returned by the exit must be less than 256 KB. This is 3276 JCL records.
- **4** End of data reached for current job. The amount of data returned by the last call to the exit is given in DATAL. The exit will not be called again for the current job.
- **8** The job could not be found in any input data set. The exit will not be called again for the current job.

MCAUSERF
: A user field that lets you allocate resources in the start/stop exit, EQQUX000, which this exit can later use. This field contains the value set by the EQQUX000 exit. IBM Workload Scheduler for z/OS does not use or update this field.

Variable-substitution exit (on JCL or job definition variable)

When you define a JCL or a job definition variable, you can specify the name of an exit that is called when substitution of the variable is required. The exit can be called at different phases depending on what you are submitting:

- If you are submitting a job contained in the job library, the exit is called at either job setup or job submission, but it is not called for promptable setup-variables.
- If you are submitting a job definition contained in the SCRPTLIB library, the exit is called during the job definition parsing process; that is, when either you add an occurrence from the MCP dialog or when you run a task (a daily plan, a replan, or a symphony renew) that results in the production of the symphony file.

You can use the exit to supply the value of a variable. For more information on JCL variables, refer to *IBM Workload Scheduler for z/OS Managing the Workload*.

The sample library SEQQSAMP that was created during installation contains the EQQJVXIT exit, which is a sample of variable-substitution exits. For information about this sample, see "JCL-variable-substitution exit" on page 397.

Installing the exit

The load module implementing the JCL-variable-substitution exit must be link-edited into a library in the LNKLST concatenation or defined by the STEPLIB DD statement in the controller JCL procedure. If the exit is used for members of the SCRPTLIB, the library that contains it must be pointed to by the STEPLIB DD statement of the JCL jobs for the daily plan extension, daily plan replanning, and symphony renewal. The load module can be link-edited with either the AMODE 24 or AMODE 31 attribute. IBM Workload Scheduler for z/OS invokes the exit in the addressing mode defined by the exit load module.

Interface to the exit

The JCL-variable-substitution exit is invoked in task mode, problem state, and key 8 and the job-step task is APF-authorized. The active task runs with the same access authority as the job-step task. The exit must restore this state before returning to its caller.

The exit is called using the BASSM instruction and must return to its caller using the address and addressing mode passed to it in general register 14. The recommended method is to restore all registers to their values on entry to the exit and return to the caller using a BSM 0,14 instruction.

If the exit is entered in AMODE 31, it must switch to AMODE 24 before performing any input or output operations, and then switch back to AMODE 31 before returning to the caller.

If the exit abends, it is flagged as *not executable*, and message EQQJ518E is issued. IBM Workload Scheduler for z/OS does not try to call the exit again.

Parameters passed to the exit all reside below the 16 MB line. Also, addresses passed to the exit are addresses of areas below the 16 MB line.

When the exit is entered, register 1 contains the address of the parameter list. Each address in this list is used to locate the parameter value. These parameters are passed to this exit:

Variable-substitution exit parameters

```
VARNAME   DS  CL8    (Name of variable for substitution)
VARTAB    DS  CL16   (JCL-variable-table name)
VARLGTH   DS  F      (Required length of variable value)
VARDEF    DS  CL44   (Variable default value as defined in table)
VARNVAL   DS  CL44   (Variable value, set by exit)
VAROPRP   DS  A      (Address of operation data)
VARWSP    DS  A      (Address of workstation data)
VAROCCP   DS  A      (Address of occurrence data)
VARDWEEK  DS  CL1    (Day of week: 1-7, where 1 is Monday)
VARWEEKY  DS  CL2    (Week of year)
VARRETC   DS  F      (Return code, set by exit)
MCAUSERF  DS  A      (Address set by the user in the EQQUX000 exit)
VARJCL    DS  CL80   (Current JCL line where this variable was found)
VARFIRST  DS  A      (Address of the first JCL line for the job)
VARLINES  DS  F      (Number of JCL lines in the job)
VARUFLN   DS  F      (Number of user fields)
VARUFLB   DS  A      (Address of user fields buffer)
```

VARNAME
The name of the variable for substitution.

VARTAB
The name of the JCL-variable table.

VARLGTH
The required length of the variable value or X'00' if a length is not defined for the variable.

VARDEF
The default value of the variable as defined in the variable table, left-justified and padded with X'40'.

VARNVAL
The value of the variable set by the exit.

VAROPRP
The address of operation data. The storage at this address is mapped by the program-interface (PIF) segment CPOP.

VARWSP
The address of workstation data. The storage at this address is mapped by the PIF segment WSCOM.

VAROCCP
The address of occurrence data. The storage at this address is mapped by the PIF segment CPOC.

VARDWEEK
The day of the week when the exit is called. It is a numeric value from 1 to 7, where 1 is Monday.

VARWEEKY
The number of the current week in the year. It is a numeric value from 1 to 53 and is calculated according to the international standard ISO 8601.

VARRETC
The return code set by the exit. These values are valid:

0 Variable processing normal.

8 Stop tailoring. If the exit was called at job submission, the operation is set to ended-in-error. If it was called at setup via the online dialogs, an error message is issued at the terminal.

MCAUSERF
: A user field that lets you allocate resources in the start/stop exit, EQQUX000, that this exit can later use. This field contains the address that is set in EQQUX000. IBM Workload Scheduler for z/OS does not use or update this field.

VARJCL
: The JCL line where this variable was found. It is set to blank characters if you are submitting a job definition contained in the SCRPTLIB library.

VARFIRST
: The address of the first JCL line. It is set to zero if you are submitting a job definition contained in the SCRPTLIB library.

VARLINES
: The number of JCL lines. It is set to zero if you are submitting a job definition contained in the SCRPTLIB library.

VARUFLN
: The number of user field records in USRFAREA.

VARUFLB
: The address of the user field area. It is laid out as follows:

USRFAREA

```
USRFNAME DS   CL16        (User field name)
USRFVAL  DS   CL54        (User field value)
```

Note:
1. The exit is not called for setup JCL variables that are defined as promptable.
2. If a value is set in the VARNVAL field, it is used by IBM Workload Scheduler for z/OS only if VARRETC is zero.
3. The value passed backed to IBM Workload Scheduler for z/OS in the VARNVAL field must satisfy verification rules defined for the variable.
4. The variable value is taken from the data in the VARNVAL field up to the first X'40'.

Automatic-job-recovery exit

Automatic-recovery control statements can include the JCL rebuild parameter CALLEXIT. This specifies the name of a program exit module that is called before the restart. The exit is called for each JCL line starting with the first record in the JCL file. The exit can decide to accept, modify, or delete the line; or insert one or more lines. The exit routine can also prevent restart or terminate automatic recovery. Refer to *Managing the Workload* for more information on automatic recovery of jobs and started tasks.

Installing the exit

The load module implementing the automatic-job-recovery exit must be link-edited into an APF-authorized library in the LNKLST concatenation or defined by the STEPLIB DD statement in the IBM Workload Scheduler for z/OS JCL procedure. If the load module performs any input or output operations it must be link-edited with RMODE(24) according to normal z/OS restrictions. Or it can be link-edited with RMODE(ANY).

IBM Workload Scheduler for z/OS invokes the exit in AMODE 31; the AMODE parameter specified at link-edit time has no effect.

Interface to the exit

The automatic-job-recovery exit is invoked in task mode, problem state, and key 8 and the job-step task is APF-authorized. The active task runs with the same access authority as the job-step task. The exit must restore this state before returning to its caller.

Control is passed to the exit using the BAL instruction. The exit must return to its caller using the address and addressing mode passed to it in general register 14.

The exit is entered in AMODE 31 but must switch to AMODE 24 before performing any input or output operations, and then switch back to AMODE 31 before returning to the caller.

If the exit abends, it is flagged as *not executable*; IBM Workload Scheduler for z/OS does not try to call the exit again.

When the exit is entered, register 1 contains the address of the parameter list. Each address in this list is used to locate the parameter value. These parameters are passed to the exit:

Automatic-job-recovery exit parameters

```
REC    DS  CL80  (JCL record)
S      DS  CL8   (Name of failed step,
                   or blank if error not associated with a step)
PS     DS  CL8   (Name of failed step in procedure, or blank)
JC     DS  CL4   (Job return code or error code)
SC     DS  CL4   (Return code or error code of failing step,
                   blank if error not associated with a step)
UF     DS  F     (User field, first time zero)
IND    DS  F     (Indicator:
                   0  First record
                   1  Another record than first
                   2  No more records in file)
RC     DS  F     (Return code)
```

The S and PS fields contain blanks if the error occurred in the initialization or completion phase of the job. The fields are also blank if there is no name on the EXEC statement.

The return code to be set by the exit (RC) is dependent on the IND value. For IND=0 and IND=1, the RC values (given as hexadecimal digits) have the following meaning:

00	Save the record (possibly updated) and return the next one.
04	Insert the record (a new one in REC) and return the same record as last time.
08	Delete the record and return the next one.
0C	No more inspection needed. Update the JS file and continue recovery actions for this job.
10	Terminate recovery actions for this job after updating the JS file.
14	Terminate recovery actions for this job without updating the JS file.

For IND=2, the RC values (given as hexadecimal digits) have the following meaning:

00	Invalid.

04	Insert the record (a new one in REC) and return control.
08	Invalid.
0C	No more inspection needed. Update the JS file and continue recovery actions for this job.
10	Terminate recovery actions for this job after updating the JS file.
14	Terminate recovery actions for this job without updating the JS file.

To inspect all JCL lines before making any changes, copy the JCL lines to your own area. Return RC=08 until IND=2 is received. Then modify the JCL and return the lines one by one to IBM Workload Scheduler for z/OS with RC=04 specified. End by returning RC=0C.

Daily-planning-report exit (EQQDPUE1)

The daily-planning-report exit (EQQDPUE1) is called by the IBM Workload Scheduler for z/OS daily-planning batch jobs and is used to manipulate lines in certain daily-planning reports. Lines from workstation-plan reports and the daily operating plan are passed to the exit and can be specially processed here. You can use the exit to add, delete, or alter lines in these reports. The exit is optional.

Installing the exit

The load module implementing the daily-planning-report exit must be link-edited into an APF-authorized library in the LNKLST concatenation or defined by the STEPLIB DD statement in the IBM Workload Scheduler for z/OS JCL procedure. If the load module performs any input or output operations it must be link-edited with RMODE(24) according to normal z/OS restrictions. Or it can be link-edited with RMODE(ANY).

IBM Workload Scheduler for z/OS invokes the exit in AMODE 31; the AMODE parameter specified at link-edit time has no effect.

Interface to the exit

The daily-planning-report exit is invoked in task mode, problem state, and key 8 and the job-step task is APF-authorized. The active task runs with the same access authority as the job-step task. The exit must restore this state before returning to its caller.

Control is passed to the exit using the BAL instruction. The exit must return to its caller using the address and addressing mode passed to it in general register 14.

The exit is entered in AMODE 31 but must switch to AMODE 24 before performing any input or output operations, and then switch back to AMODE 31 before returning to the caller.

If the exit abends, it is flagged as *not executable*; IBM Workload Scheduler for z/OS does not try to call the exit again.

When the exit is entered, register 1 contains the address of the parameter list. Each address in this list is used to locate the parameter value. These parameters are passed to the exit:

EQQDPUE1 parameters

```
REPTYPE   DC   H      (Report type)
REPLINE   DC   CL127  (Line of report)
LINETYPE  DC   H      (Type of line)
WSNAME    DC   CL4    (Name of WS (if REPTYPE=3))
LINEBACK  DC   CL127  (Line to insert from exit)
ACTION    DC   H      (Action for line)
```

REPTYPE
The call type. These values are valid:

1	All reports ended (no line available in this call)
2	Daily operating plan
3	Plan for workstation.

REPLINE
The line supplied to this exit; it has a maximum of 127 characters. This parameter specifies the line to print.

LINETYPE
Specifies the line type to print. These values are valid:

1	Subheader/its underscore/company heading
2	Sub-subheader/its underscore
3	Space line (type ---------------)
4	Space line (type │ │ │)
5	Data line
6	Blank line.

WSNAME
Specifies the name of the workstation. It is used only if REPTYPE=3; for the others, it remains blank.

LINEBACK
The output line; it has a maximum of 127 characters. The first character must be blank (it is an ASA control character).

ACTION
Specifies the action for the line. These values are valid:

0	Line unchanged
4	Line changed
8	Delete line
12	Insert line before line passed
16	Do not call any more; line unchanged.

Application-description-validation exit (EQQUXPIF)

About this task

The application-description-validation exit (EQQUXPIF) is called by the PIF program during the application description update (INSERT AD or REPLACE AD). It is used to validate the application description. The exit is optional.

Installing the exit

The load module implementing the application-description-validation exit must be link-edited into an APF-authorized library defined by the STEPLIB DD statement of the current PIF program. This exit is loaded automatically by the PIF program without the need to specify any parameter.

IBM Workload Scheduler for z/OS invokes the exit in AMODE 31; the AMODE parameter specified at link-edit time has no effect.

Interface to the exit

The application-description-validation exit is invoked in task mode, problem state, and key 8 and the job-step task is APF-authorized. The exit must restore this state before returning to its caller.

Control is passed to the exit using the BAL instruction. The exit must return to its caller using the address and addressing mode passed to it in general register 14.

When the exit is entered, register 1 contains the address of the parameter list. Each address in this list is used to locate the parameter value. These parameters are passed to the exit:

EQQUXPIF parameters

```
REQUEST   DS   CL8   (Request type)
RESOURC   DS   CL8   (Resource type)
RECLEN    DS   F     (Record length)
RECPTR    DS   A     (Record address)
RETCO     DS   F     (Return code)
ERRMSG    DS   CL80  (Error message string)
```

REQUEST
The type of database request (INSERT or REPLACE)

RESOURC
The type of resource to be validated (AD).

RECLEN
The length of the database record to be validated (AD record, see DCLADR layout in the *IBM Workload Scheduler for z/OS Diagnosis Guide and Reference*).

RECPTR
The address of the record area. It can only be validated.

RETCO
The return code of the exit. If its value is greater than 0 the application description update (INSERT AD or REPLACE AD) fails and a message containing the string ERRMSG is displayed.

ERRMSG
The message that explains the reason for the update failure. The message is displayed only if RETCO is greater than 0.

System Automation for z/OS user exit (EQQUXSAZ)

About this task

The system automation exit EQQUXSAZ is called every time an operation defined on a general and automatic workstation, with the Automation option enabled, is scheduled. The exit cannot modify any parameters received or other IBM Workload Scheduler for z/OS data or resources. EQQUXSAZ can examine the parameters and take some actions external to IBM Workload Scheduler for z/OS, based on the parameter information.

You can use EQQUXSAZ to send the command request to System Automation for z/OS, through the NetView PPI interface.

The IBM Workload Scheduler for z/OS sample library that was created during installation contains a sample EQQUXSAZ exit, written in assembler language.

This exit does not submit the command to System Automation for z/OS, it only receives the input parameters and stores them in a local variables area. You can modify the sample exit to create your own submission procedure. You need to compile and link-edit the Tivoli Workload for z/OS exit only if you customized the submission part. The EQQUXSAZ exit that actually submits the command is provided with System Automation for z/OS. You find it defined in the System Automation for z/OS library SINGMOD1, as alias of EVJUXSAZ. You must concatenate that library in the STEPLIB of the controller startup procedure.

For details about the System Automation EQQUXSAZ, refer to System Automation for z/OS publications.

Note:

1. IBM Workload Scheduler for z/OS does not perform any validity or syntax checking on the request to be routed to System Automation for z/OS.
2. In case of System Automation timeouts, the operation could be marked with error OAUT on the IBM Workload Scheduler for z/OS side, even if it completes successfully later, on the System Automation side.
3. For operations defined on automation workstations, the EQQUX007 exit is not invoked; only EQQUXSAZ is invoked.
4. You can disable the loading of the EQQUXSAZ exit by setting CALL12(NO) in the EXIT initialization statement. This setting prevents message `EQQN102W AN OPC USER EXIT LOAD MODULE, EQQUXSAZ, COULD NOT BE LOADED` from being issued. However, it also prevents you from using any System Automation function.

Installing the exit

The load module implementing the system automation exit must be link-edited into an APF-authorized library in the LNKLST concatenation or defined by the STEPLIB DD statement in the IBM Workload Scheduler for z/OS JCL procedure. If the load module performs any input or output operations, it must be link-edited with RMODE(24) according to normal z/OS restrictions. If the load module does not perform any input or output operations, it can be link-edited with RMODE(ANY).

IBM Workload Scheduler for z/OS invokes the exit in AMODE 31; the AMODE parameter specified at link-edit time has no effect.

Interface to the exit

The system automation exit is invoked in task mode, problem state, and key 8. The job-step task is APF-authorized. The active task runs with the same access authority as the job-step task. The exit must restore this state before returning to the caller.

Control is passed to the exit using the BAL instruction. The exit must return to the caller using the address and addressing mode passed to it in general register 14.

The exit is entered in AMODE 31 but must be switched to AMODE 24 before any input or output operation is performed. Before the exit returns to the caller, switch it back to AMODE 31.

When the exit is entered, register 1 contains the address of the parameter list. Each address in this list is used to locate the parameter value. The following parameters are passed to the exit:

EQQUXSAZ parameters

```
ADNAME    DS   CL16  (Application name)
WSNAME    DS   CL4   (Workstation name)
WSDEST    DS   CL8   (Workstation destination name or NetView domain)
JOBNAME   DS   CL8   (Operation job name)
IA        DS   CL10  (Input Arrival date and time, format YYMMDDHHMM)
OPNUM     DS   H     (Operation number)
COMMTEXT  DS   CL255 (SA command text)
COMPINFO  DS   CL64  (SA completion information)
AUTFUNC   DS   CL8   (SA automated function for operation)
SECELEM   DS   CL8   (SA security element)
OWNER     DS   CL16  (Application owner name)
RETCODE   DS   F     (Return code)
```

ADNAME
Application name for the current operation.

WSNAME
Workstation name for the current operation

WSDEST
User-defined workstation destination or destination NetView domain.

JOBNAME
Job name defined for the current operation.

IA Input arrival date and time for the current operation, in the format YYMMDDHHMM.

OPNUM
Operation number for the current operation.

COMMTEXT
Text of the command to be routed to System Automation. It is free format and can contain IBM Workload Scheduler for z/OS variables that are replaced before the command is passed to System Automation for z/OS. If an error occurs during this phase, the operation is set to E with code OJCV. No syntax checking on text content is performed on the IBM Workload Scheduler for z/OS side.

COMPINFO
Completion information. You can optionally specify the following information, in the following order, separated by a comma:

- The maximum wait time, in the format hh:mm:ss. If specified, System Automation for z/OS waits for the completion of the command for the specified time interval. If the command does not complete, System Automation for z/OS posts the operation in error.
- The maximum return code accepted as successful execution. You can specify the name of an optional user-supplied completion checking routine. The completion checking routine ensures that the command achieved the expected results, before posting the operation as complete.

AUTFUNC
Automated function (for operation). This parameter is optional. If specified, the command is run on the NetView task associated with this automated function in System Automation for z/OS. You can use this parameter to serialize commands. If this parameter is not specified, the command is run by any available NetView tasks.

SECELEM
Security element. It is an optional parameter used for security tracking of

the operation. You can use it in alternative or together with the job name, for security validation of the operation on the System Automation for z/OS side.

OWNER
Name of the owner of the current operation.

RETCODE
Maximum return code from the exit. The default is 0 (normal return, IBM Workload Scheduler for z/OS processing continues). For a complete list, refer to the *System Automation for z/OS Messages and Codes* manual.

Note: For the command text string (COMMTEXT), any text case is allowed. The value specified in COMPINFO, AUTOPER, and SECELEM is automatically changed to uppercase by IBM Workload Scheduler for z/OS.

Chapter 5. Open Systems Integration

You can use the controller to provide a single, consistent, control point for submitting and tracking the workload on any operating environment. IBM Workload Scheduler for z/OS provides open interfaces to enable you to integrate the planning, scheduling, and control of units of work such as online transactions, file transfers, or batch processing in any operating environment that can establish communication with z/OS.

This chapter describes how you can use IBM Workload Scheduler for z/OS to control the workload in operating environments that do not support a tracker, through the operation-initiation exit, EQQUX009. Also, an example is provided of an alternate method for controlling VM processing.

Controlling heterogeneous systems

IBM Workload Scheduler for z/OS provides open interfaces to let you submit the workload to and report status from any operating environment that can establish communication with z/OS.

When an operation on a workstation that specifies a user-defined destination ID is ready to be started, IBM Workload Scheduler for z/OS calls the operation-initiation exit, EQQUX009. The exit is called for operations on computer workstations that have satisfied the normal IBM Workload Scheduler for z/OS submission criteria for job or started-task operations. The exit is passed information about the operation to be started, the destination on which it should be started and, if available, any JCL equivalent information from the IBM Workload Scheduler for z/OS job library (EQQJBLIB) or the job-setup file (EQQJSxDS).

The exit is responsible for transmitting the required data to the target destination. There are several methods available to transmit data to the various operating environments and to report the status of the operation back to the controller. For example you could use APPC/MVS, TCP/IP, or NetView FTP. One of the sample exits provided uses TCP/IP to communicate commands to an OS/2 environment.

The SEQQSAMP sample library contains samples that show how you can use IBM Workload Scheduler for z/OS to communicate with heterogeneous systems.

- EQQOS2TR and EQQX9OS2 provide you with sample programs to communicate with an OS/2 environment to start commands and report status back to the controller.
- EQQCMV2 and EQQUX09N allow you to schedule and control the VM workload by defining operations exactly as you would for your z/OS workload.
- EQQAIXTR and EQQX9AIX provide you with sample programs to communicate with an AIX environment to start tasks or issue commands and report status back to the controller. This sample has been developed and tested in an AIX environment but can be ported to any UNIX environment that uses a compatible shell script.

Appendix B, "Sample library (SEQQSAMP)," on page 391 contains descriptions of all samples distributed with IBM Workload Scheduler for z/OS.

Status reporting for operations at user-defined destinations is achieved by using the OPSTAT command in native TSO, in a CLIST or REXX EXEC, or in SYSIN to the EQQEVPGM program, or by invoking the EQQUSIN or EQQUSINT subroutine.

Workstations that specify a user-defined destination are initialized to *unknown* status when the controller is started. The installation is responsible for setting the workstation to *active* status. Status reporting for the workstation is achieved by using the WSSTAT command in native TSO, in a CLIST or REXX EXEC, or in SYSIN to the EQQEVPGM program, or by invoking the EQQUSIN or EQQUSINW subroutine. The logic to set the workstation status to *active* should not be in EQQUX009 because the exit will never be called when the status is offline.

You should consider using the start/stop exit EQQUX000 to set workstation status for user-defined destinations. The sample library member EQQUX0N calls the EQQUSINW subroutine to set the status of a workstation. See "Start or stop exit" on page 394 for more information about the EQQUX0N sample.

Setting up the environment

About this task

You need to perform the following steps to enable submission and tracking for a particular environment:

1. Choose a unique destination ID to represent the target environment.
2. Define the destination in the **USER** keyword of the ROUTOPTS statement.
3. Create a workstation description in the IBM Workload Scheduler for z/OS dialog. The workstation type must be computer with automatic reporting.
4. Write the code necessary to support the environment:
   - EQQUX009.
   - A program to receive and initiate the data on the target system and report status changes to IBM Workload Scheduler for z/OS.
   - Code to handle workstation status. Consider invoking this process in the EQQUX000 exit, which is called at IBM Workload Scheduler for z/OS start.
5. Specify **CALL09(YES)** in the EXITS statement.

You can use the sample programs for your code.

For more information see:

- Chapter 6, "Reporting events to IBM Workload Scheduler for z/OS," on page 273
- "Operation-initiation exit" on page 397
- *Managing the Workload* for a description of the OPSTAT and WSSTAT commands.

Submitting and tracking the workload

Managing the Workload contains details of how IBM Workload Scheduler for z/OS selects work to be submitted at particular workstations. The following is a summary of the submission and tracking process for operations on a workstation that has a user-defined destination ID:

- When the operation is selected as the next operation to be submitted, the status is set to SU. IBM Workload Scheduler for z/OS calls EQQUX009 and provides data stored in the EQQJSxDS or EQQJBLIB data sets. The information does not have to be stored in these libraries; the exit could locate the data elsewhere or the data might be stored at the target destination.

- EQQUX009 transmits the data to the target destination. A good return code from the exit will leave the operation in status `SU` until changed through either the normal IBM Workload Scheduler for z/OS tracking routines (if the job is run on z/OS and an event writer and an event data set are present) or through the OPSTAT interface.
- If the operation enters a queuing system in the target destination, this can be reported to IBM Workload Scheduler for z/OS through the OPSTAT command with STATUS(Q). IBM Workload Scheduler for z/OS will set the operation status to `SQ`.
- When the operation actually starts to execute, this is reported using the OPSTAT command for STATUS(T). IBM Workload Scheduler for z/OS will set the operation status to `SS`.
- Termination is reported using the OPSTAT command for STATUS(C), if normal, or STATUS(E), if abnormal.

It is recommended that you use the TOKEN passed to the EQQUX009 as input to OPSTAT. The TOKEN is a 4-byte value automatically generated by IBM Workload Scheduler for z/OS when the operation is selected to be scheduled. The purpose of the token is to make it easier for OPSTAT users to uniquely identify an IBM Workload Scheduler for z/OS operation.

An alternate method for controlling VM processing

About this task

Under VM, some EXECs must be run in given sequences, and the results must be checked to ensure that the EXECs have completed successfully. Such work might be repeated at regular intervals. In addition, some applications might require that VM processing interface with, or be synchronized with, z/OS processing.

So, there is a need for VM production control mechanisms that correspond to those that exist under z/OS. Here is one way you can automate control of VM operations:

1. Set up a z/OS job to send a request to a VM AUTOLOG user to execute an EXEC; you would normally do this using an NJE-RSCS link. On completion, the EXEC sends a job back to z/OS, which issues an OPSTAT command to inform IBM Workload Scheduler for z/OS of the status of the VM operation. The first z/OS job is scheduled by IBM Workload Scheduler for z/OS in the same way as any other computer workstation operation.
2. Set up a second operation to represent the execution of the VM EXEC. You should define the operation on a general workstation with the automatic reporting attribute, which is set up for this purpose. This is the operation whose status is changed by the OPSTAT command issued by VM.

You can use this method to plan, control, and track the workload of multiple VM users at local and remote sites.

This method does not use any IBM Workload Scheduler for z/OS exits or require other programming effort. The sample library member EQQCVM is maintained for installations that have implemented this method in OPC/A or previous IBM Workload Scheduler for z/OS releases.

Method of use

About this task

A sample for controlling VM workload from IBM Workload Scheduler for z/OS is delivered on the sample file under member name EQQCVM. The sample member includes these parts:

- Batch loader control statements for defining an application description to run a VM job.
- An example of JCL to signal VM that a job is to be started.
- A CMS EXEC called OPCWATCH, which is run in the AUTOLOG VM user. OPCWATCH receives the start signal from z/OS and drives the requested EXEC on VM.
- A CMS EXEC called OPCSTAT, which can be used by OPCWATCH to signal the status change of the VM job to IBM Workload Scheduler for z/OS.

To use IBM Workload Scheduler for z/OS to drive VM operations, follow these steps:

1. Define a new IBM Workload Scheduler for z/OS workstation, for instance VM, as a general workstation with the automatic reporting attribute. The advantage of having a separate workstation is that separate ready lists and reports can be produced for VM operations based on the workstation name.
2. Define an application VMJJJJ with these operations:
   ```
   CPU1 010  JOBNAME RJJJJ      'SEND START ORDER TO VM'
   VM   020  JOBNAME JJJJ       'TRACKS REAL EXECUTION OF EXEC'
   ```
   The run cycle and other characteristics should be the same as for normal z/OS applications.

   The figure below shows the member RJJJJ in the IBM Workload Scheduler for z/OS JCL library. This JCL will be executed by the `CPU1` operation.

```
//RJJJJ    JOB  Job statement parameters according to
//              your installation standards
/*JOBPARM CARDS=100
/*ROUTE PUNCH VMNODE.VMUSER
//*****************************************************************
//*
//*      A z/OS job to signal VM when the controlled job
//*      is ready to be started on the VM system.
//*
//*****************************************************************
//B        EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=Q
//SYSUT1   DD *
JJJJ
/*
//SYSUT2   DD SYSOUT=B
//SYSIN    DD DUMMY
```

Figure 1. Member RJJJJ in the IBM Workload Scheduler for z/OS JCL library

   The first record in the `//SYSUT1` data stream contains the name of the EXEC to be executed, followed by any parameters that should be passed to the EXEC.

3. Set up a VM AUTOLOG user to await the arrival of any reader files. You can use a wait EXEC to drive the VM AUTOLOG user (such as the OPCWATCH EXEC supplied in member EQQCVM in the IBM Workload Scheduler for z/OS

sample library). When reader files are sent to this user, it processes the EXEC named in the //SYSUT1 data. The EXECs that are processed are logged in the file OPCA LOG A.

Note:

- Each VM EXEC that is processed should set a return code to indicate whether it has run successfully.
- The wait EXEC depends on the module WAKEUP to invoke its execution. The WAKEUP module is available in the VM/IPF distribution.

Figure 2 on page 272 shows an example of controlling VM operations from IBM Workload Scheduler for z/OS. In this example:

1. IBM Workload Scheduler for z/OS running under z/OS sends EXEC name JJJJ to a VM user, which is running the OPCWATCH EXEC.
2. OPCWATCH invokes two other VM EXECs, JJJJ and OPCSTAT.
3. Before and after JJJJ is being processed, OPCSTAT reports the status to an IBM Workload Scheduler for z/OS general automatic-reporting workstation VM.
4. In MVS, the jobs from VM execute a program to perform automatic-event reporting for the particular combination of application ID VMJJJJ, job name JJJJ, and status.

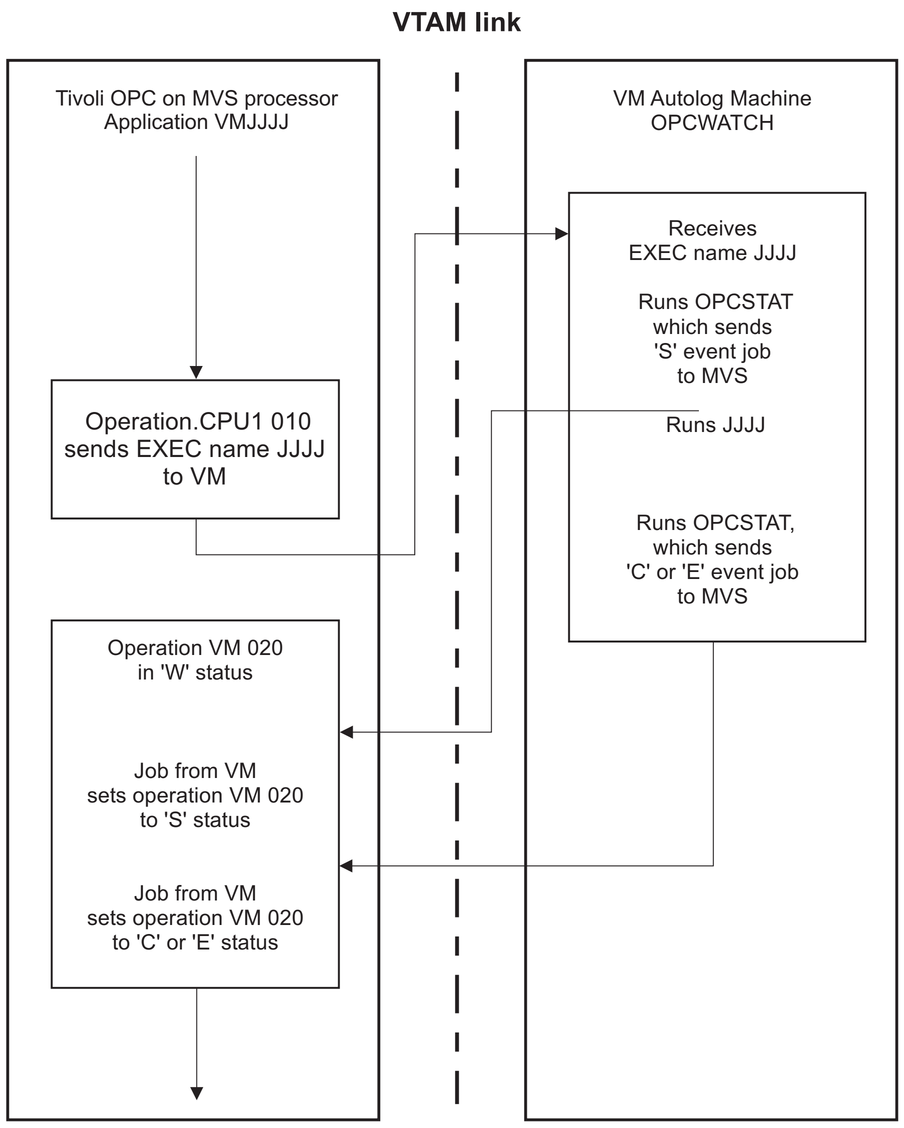


Figure 2. Using automatic-event reporting to control VM operations

If IBM Workload Scheduler for z/OS running under z/OS fails or if the communication link fails, a printed workstation plan and an IBM Workload Scheduler for z/OS ready list from daily planning will still exist. This information tells which jobs must be run and the order in which they must be run. An IBM Workload Scheduler for z/OS VM-user log lists the EXECs that have been started and those that have been completed. With this information, processing can continue either automatically when the link is reestablished or manually.

The jobs transmitted to and from VM add little additional load to z/OS. To avoid delays, a dedicated job class and initiator should be reserved for VM communication.

Because IBM Workload Scheduler for z/OS does not require shared DASD for the preceding method, this method can be used to drive multiple VM users in different locations. You can also use this method to drive other z/OS systems. This method is particularly useful when a remote z/OS system runs a small number of backups and cleanups that are initiated and controlled from a central site.

Chapter 6. Reporting events to IBM Workload Scheduler for z/OS

This chapter describes how you can report events to IBM Workload Scheduler for z/OS. This chapter contains General-use Programming Interface and Associated Guidance Information.

IBM Workload Scheduler for z/OS lets you track activities in your data processing environment. Information about the status of these activities can be reported manually by dialog users or collected automatically by IBM Workload Scheduler for z/OS. These activities are known as *events*. For z/OS systems, IBM Workload Scheduler for z/OS uses SMF and JES exits to collect event information automatically. For example, IBM Workload Scheduler for z/OS reports when a started task has started or when a job has ended. But there might be activities in your production workload that cannot be detected by JES and SMF exits, which you want to report to IBM Workload Scheduler for z/OS. You can do this by supplying event information to IBM Workload Scheduler for z/OS.

For example, assume that an IBM Workload Scheduler for z/OS application is dependent on a data set that is updated by an online transaction. For such an application, the batch job that uses the data set must not be started until the data set has been successfully updated. You ensure this by defining a special resource that is needed by the batch job but is unavailable. The online transaction can then make the special resource available by calling the EQQUSIN subroutine or issuing an SRSTAT command as its last processing step. Alternatively, you can define an operation at a general automatic workstation that is set to complete by the online transaction, using EQQUSIN or the OPSTAT command. The batch job, which is dependent on this operation, does not process before the data set has been updated.

Supplying event information to IBM Workload Scheduler for z/OS

You can report event information to IBM Workload Scheduler for z/OS, both manually and automatically, by establishing procedures that use IBM Workload Scheduler for z/OS TSO commands and IBM Workload Scheduler for z/OS subroutines. You can:

- Request a backup of an IBM Workload Scheduler for z/OS data set using the BACKUP TSO command, or EQQUSIN or EQQUSINB subroutine.
- Make a special resource available or unavailable using the SRSTAT TSO command, or EQQUSIN or EQQUSINS subroutine.
- Change the status of an operation using the OPSTAT TSO command, or EQQUSIN or EQQUSINT subroutine.
- Update the user data field of an operation in the current plan using the OPINFO TSO command, or EQQUSIN or EQQUSINO subroutine.
- Generate a workstation status event for a current plan workstation that specifies a user-defined destination using the WSSTAT TSO command, or EQQUSIN or EQQUSINW subroutine.

If you plan to issue the TSO commands many times per day from a long-running non-TSO address space, for example NetView, it is recommended that you use an IBM Workload Scheduler for z/OS subroutine instead. When you issue the commands from TSO or as input to the EQQEVPGM program, a TSO environment must be established each time, and some of the resources remain allocated until the task ends, which might lead to a storage shortage if the commands are issued many times.

For more information about IBM Workload Scheduler for z/OS TSO commands, refer to *Managing the Workload* The remainder of this chapter describes how you use IBM Workload Scheduler for z/OS subroutines.

Diagnosis Guide and Reference describes in detail how events are created and then processed by IBM Workload Scheduler for z/OS.

General information about IBM Workload Scheduler for z/OS subroutines

Read this information before you use IBM Workload Scheduler for z/OS subroutines:

- IBM Workload Scheduler for z/OS provides both individual subroutines (EQQUSINB, EQQUSINS, EQQUSINO, EQQUSINT, and EQQUSINW) and a general subroutine (EQQUSIN), which reside in the AEQQMOD0 distribution library. The subroutines can be either link-edited into the load module that they are called from or moved to an authorized library and dynamically invoked. Regardless of the method you choose, these subroutines must run either APF authorized, in PSW key 0–7, or in supervisor state. If a subroutine is called from an unauthorized environment, for example CICS®, it should be called from a user SVC. Because these subroutines do not perform input/output operations or other operations involving waits, they will not adversely affect the performance of the environment that they are called from.
- The subroutines do not perform RACF security checking of the data passed to them. One reason is that the event information generated by a subroutine could be used in two or more IBM Workload Scheduler for z/OS address spaces where the security rules differ. You are responsible for the security of the function. You can protect these subroutines by placing them in a protected library. Or, the program calling the subroutine can perform security checking.
- The event information that you report has a primary function, but in many instances, you can provide extra information that can cause more updates. If you want to provide extra information through an IBM Workload Scheduler for z/OS subroutine, use EQQUSIN, which has additional parameters that are not available in the individual subroutines. EQQUSIN is a general subroutine that you can use instead of any individual subroutine. It is functionally equivalent to all IBM Workload Scheduler for z/OSTSO commands.

  If you already use individual subroutines, you can continue to use them without changes. But, they are retained only for compatibility; EQQUSIN is preferred.
- The parameters that you pass to the subroutines are checked only for the correct format; that is, numeric fields must contain only numbers within a valid range, date fields must contain valid dates, and so on. The parameters are *not* checked for their validity for a particular IBM Workload Scheduler for z/OS address space. For example, a workstation name that you specify is not verified against the actual workstations that exist in a particular IBM Workload Scheduler for z/OS current plan. Also, a single event record can be generated that is used in two or more IBM Workload Scheduler for z/OS address spaces. A particular parameter (for example, the application description ID) might be valid for one address space and not another.

If the minimum parameter requirements are met and the parameters are in the correct format, the subroutines will execute successfully and generate an event record.

When the event is processed by the controller, it is checked for validity. If errors are found, an error message is written to the controller message log (EQQMLOG).

- You can use the subroutines even if IBM Workload Scheduler for z/OS (in particular, the event writer subtask) is not active. Event records are still generated and placed in the event writer queue. When the event writer starts, the event records are removed from the queue and written to the event data set.
- If, when you write a new process to use the subroutines, an invalid buffer is created, the event manager might abend on the controller receiving the event. In this case, the Event Manager must be restarted before any further testing can take place. The invalid buffer might be created in such a way that the SUBSYSTEM_NAME field is not read, even if you provide one. In this case, the Event Manager might abend on all subsystems that have a tracker active on the LPAR where you run the test.

Using the general IBM Workload Scheduler for z/OS subroutine (EQQUSIN)

EQQUSIN is a general subroutine that supports all event-reporting types. You can use it instead of individual subroutines. It also provides extra update capabilities.

Appendix B, "Sample library (SEQQSAMP)," on page 391 describes samples that help you use EQQUSIN.

Invocation requirements

EQQUSIN has these invocation requirements:

Authorization
APF authorized

Dispatchable unit mode
Task mode

Amode
31-bit

ASC mode
Primary or access register (AR)

Interrupt status
Enabled for I/O and external interrupts

Locks No locks held

Control parameters
All parameters must be addressable by the caller and in the primary address space.

EQQUSIN parameters

Your program passes parameters to EQQUSIN in an *APP buffer*. Register 1 must point to the address of the buffer and the high-order bit must be on. The format of the buffer is the same as that used to communicate with IBM Workload Scheduler for z/OS through the application programming interface (API).

You can invoke EQQUSIN on z/OS systems where IBM Workload Scheduler for z/OS is installed. The request is sent through the subsystem interface (SSI) to IBM Workload Scheduler for z/OS. The return code from the call to the SSI is returned in register 15.

EQQUSIN supports multiple requests in the same buffer. The buffer can contain these sections:

APP Fixed section - identifies the buffer

APPOBJ
Object section - identifies the object (event type)

APPSEL
Selection section - contains a field name that is used in locating one or more instances of the object

APPVAL
Selection value section - contains a field value that is used in locating one or more instances of the object

APPFLD
Field section - identifies the field to update in the selected instance of the object

APPDAT
Data section - contains a new value for each APPFLD section.

Figure 3 is an example of the layout of a buffer (segments might be in other orders). This request uses 2 selection fields to locate an object and updates 1 field in the selected object. The arrows show the buffer parts that each section type points to. APP and APPOBJ point to related sections using triplet fields, which specify the offset, the length, and the number of the section type. APPSEL uses offset and length fields to point to an APPVAL section. All offsets are relative to the start of the buffer (offset 0).

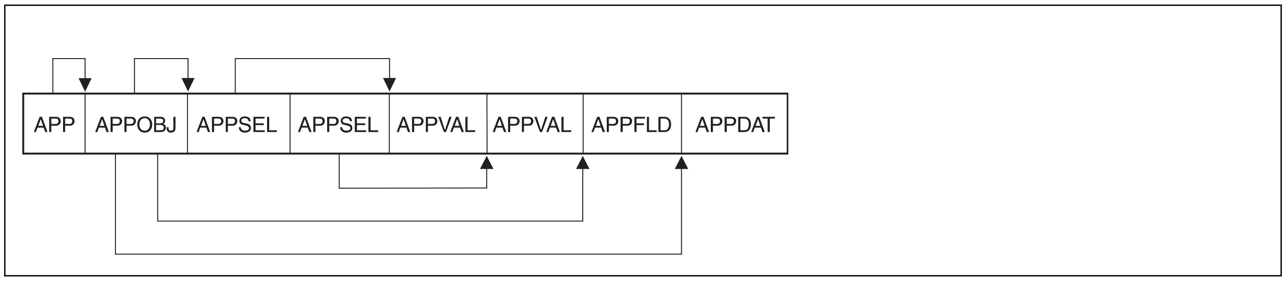


Figure 3. EQQUSIN buffer example

Each section is described here in more detail.

APP - fixed section

The buffer that your program passes to EQQUSIN must contain a fixed section and it must be the first section in the buffer. It identifies the buffer, its size, the default request type, and points to object sections. The buffer must contain only 1 fixed section, even if multiple requests are passed in the same buffer.

The fixed section has this format:

Offsets					
Dec	Hex	Type	Len	Name	Description
0	(0)	STRUCTURE	80	APP	APPC BUFFER MAPPING
0	(0)	CHARACTER	4	APPDESC	BLOCK DESCRIPTOR (APP)
4	(4)	CHARACTER	2	APPVER	VERSION NUMBER (02)
6	(6)	BITSTRING	2	*	RESERVED
8	(8)	CHARACTER	3	APPTYPE	EYE CATCHER (DIA)
11	(B)	BITSTRING	1	APPFLAGS	RESERVED
12	(C)	SIGNED	4	APPTOTSZ	TOTAL SIZE
16	(10)	CHARACTER	8	APP_TYPE	DIALOG DATA TYPE (CREATE for EQQUSIN)
24	(18)	SIGNED	4	APP_RETCODE	*RETURN CODE
28	(1C)	SIGNED	4	APP_RSNCODE	*REASON CODE
32	(20)		12	APP_OBJ_TRIPLET	OBJECT SECTION TRIPLET
32	(20)	SIGNED	4	APP_OBJ_OFF	OFFSET TO FIRST OBJECT SECTION
36	(24)	SIGNED	4	APP_OBJ_LEN	LENGTH OF ALL OBJECT SECTIONS
40	(28)	SIGNED	4	APP_OBJ_NBR	NUMBER OF OBJECT SECTIONS
44	(2C)	SIGNED	4	APP_ERR_OFF	*OFFSET TO VERIFICATION ERROR
48	(30)	CHARACTER	8	*	RESERVED
56	(38)	CHARACTER	16	APPTOKEN	*TOKEN FIELD
72	(48)	CHARACTER	8	*	RESERVED

In the fixed section:

APPDESC
Is the block descriptor and has the value APP.

APPVER
Is the version number and has the value 02.

* Offset 6 (X'6'). Set this reserved field to binary zeros (X'00').

APPTYPE
Is the eye-catcher and has the value DIA.

APPFLAGS
Set this reserved field to binary zeros (X'00').

APPTOTSZ
Is the total size of the buffer.

APP_TYPE
Is the request type that is the default for all requests. It is used if you do not provide a value for APPOBJ_TYPE in an object section of the buffer. If you set this field to blanks (X'40'), you must specify a request in each object section of the buffer. Only CREATE is valid for EQQUSIN.

APP_OBJ_TRIPLET
Contains the offset to the first APPOBJ section, the length of all sections, and the number of sections.

APP_RETCODE
Is the return code that is set by EQQUSIN. In the call to EQQUSIN, set this field to binary zeros (X'00'). For more information, see "Return codes and reason codes generated by EQQUSIN" on page 289.

APP_RSNCODE
Is the reason code that is set by EQQUSIN. In the call to EQQUSIN, set this field to binary zeros (X'00'). For more information, see "Return codes and reason codes generated by EQQUSIN" on page 289.

APP_ERR_OFF
Is set by EQQUSIN when APP_RSNCODE indicates an error that has an offset associated with it. It is the offset in the buffer where a verification error was found. In the call to EQQUSIN, set this field to binary zeros (X'00').

* Offset 48 (X'30'). Set this reserved field to binary zeros (X'00').

APPTOKEN
Is a value that your program can set to uniquely identify a buffer. It could be, for example, a time stamp. APPTOKEN can be useful if you invoke EQQUSIN through the API and there is more than one active request from your ATP at a time.

* Offset 72 (X'48'). Set this reserved field to binary zeros (X'00').

APPOBJ - object section

This section identifies the event type you want to report to IBM Workload Scheduler for z/OS. The EQQUSIN buffer must contain an object section. It can contain more than one object section, but all object sections must be in contiguous storage; that is, they must follow one another. The part of the buffer containing object sections is pointed to by the APP_OBJ_TRIPLET in the fixed section. APPOBJ itself points to APPSEL, APPFLD, and APPDAT sections.

The object section has this format:

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	84	APPOBJ	OBJECT SECTION APPOBJ_PTR = ADDR(APP) + APP_OBJ_OFF
0	(0)		24	APPOBJ_ID	OBJECT IDENTIFIER
0	(0)	CHARACTER	16	APPOBJ_NAME	OBJECT NAME
16	(10)	CHARACTER	8	APPOBJ_KEY_TYPE	KEY TYPE
24	(18)		12	APPOBJ_FLD_TRIPLET	FIELD SECTION TRIPLET
24	(18)	SIGNED	4	APPOBJ_FLD_OFF	OFFSET TO FIRST FIELD SECTION
28	(1C)	SIGNED	4	APPOBJ_FLD_LEN	LENGTH OF ALL FIELD SECTIONS

Offsets					
Dec	Hex	Type	Len	Name	Description
32	(20)	SIGNED	4	APPOBJ_FLD_NBR	NUMBER OF FIELD SECTIONS
36	(24)		12	APPOBJ_SEL_TRIPLET	SELECTION SECTION TRIPLET
36	(24)	SIGNED	4	APPOBJ_SEL_OFF	OFFSET TO FIRST SELECTION SECTION
40	(28)	SIGNED	4	APPOBJ_SEL_LEN	LENGTH OF A SINGLE SELECTION SECTION
44	(2C)	SIGNED	4	APPOBJ_SEL_NBR	NUMBER OF SELECTION SECTIONS
48	(30)		12	APPOBJ_DAT_TRIPLET	DATA SECTION TRIPLET
48	(30)	SIGNED	4	APPOBJ_DAT_OFF	OFFSET TO FIRST DATA SECTION
52	(34)	SIGNED	4	APPOBJ_DAT_LEN	LENGTH OF ALL DATA SECTIONS
56	(38)	SIGNED	4	APPOBJ_DAT_NBR	NUMBER OF DATA SECTIONS
60	(3C)	CHARACTER	8	APPOBJ_TYPE	DIALOG DATA TYPE (CREATE for EQQUSIN)
68	(44)	SIGNED	4	APPOBJ_RET	OBJECT LEVEL RETURN CODE
72	(48)	SIGNED	4	APPOBJ_RSN	OBJECT LEVEL REASON CODE
76	(4C)	CHARACTER	8	APPOBJ_AUTH	RACF AUTHORITY (READ or UPDATE)

In the object section:

APPOBJ_NAME
Is the event type you want to report to IBM Workload Scheduler for z/OS. You can specify these object names:

CP_OPER_EVENT
Current plan operation status. This is equivalent to the OPSTAT TSO command.

CP_OPINFO_EVENT
Current plan operation user data. This is equivalent to the OPINFO TSO command.

CP_SR_EVENT
Current plan special resource. This is equivalent to the SRSTAT TSO command.

BACKUP_EVENT
Backup request. This is equivalent to the BACKUP TSO command.

CP_WS_EVENT
Current plan workstation. This is equivalent to the WSSTAT TSO command.

APPOBJ_KEY_TYPE
Is the key type, which must be SAME for EQQUSIN. If you set this field to blanks (X'40'), SAME is used by default.

APPOBJ_FLD_TRIPLET
Contains the offset to the first APPFLD section, the length of each section, and the number of sections. Set these fields to binary zeros (X'00') when the object is BACKUP_EVENT.

APPOBJ_SEL_TRIPLET
Contains the offset to the first APPSEL section, the length of a single section, and the number of sections.

APPOBJ_DAT_TRIPLET
Contains the offset to the first APPDAT section, the length of all sections, and the number of sections. Set these fields to binary zeros (X'00') when the object is BACKUP_EVENT.

APPOBJ_TYPE
Is the request type. Only CREATE is valid for EQQUSIN. If you set this field to blanks (X'40'), you must specify CREATE in the APP_TYPE field of the fixed section.

APPOBJ_RET
In the call to EQQUSIN, set this field to binary zeros (X'00'). No return code is generated in the object section for a CREATE request.

APPOBJ_RSN
In the call to EQQUSIN, set this field to binary zeros (X'00'). No reason code is generated in the object section for a CREATE request.

APPOBJ_AUTH
Set this field to blanks (X'40') in the call to EQQUSIN. It is not updated for a CREATE request.

APPSEL - selection section

This section identifies a particular field in the object type. The buffer must contain a selection section. APPSEL is pointed to by its object section and must itself point to an APPVAL section where a selection value is specified. To identify a particular instance of an object, you might need to specify more than one APPSEL for each APPOBJ. The selection sections for a particular APPOBJ must be in contiguous storage.

The selection section has this format:

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	36	APPSEL	SELECTION SECTION ADDRESS OF FIRST SELECTION SECTION FOR THIS OBJECT: APPSEL_PTR =ADDR(APP) + APPOBJ_SEL_OFF
0	(0)	CHARACTER	16	APPSEL_NAME	OBJECT FIELD NAME
16	(10)	CHARACTER	2	APPSEL_OPER	OPERATOR
18	(12)	CHARACTER	10	*	RESERVED
28	(1C)	SIGNED	4	APPSEL_VALUE_OFF	VALUE OFFSET
32	(20)	SIGNED	4	APPSEL_VALUE_LEN	VALUE LENGTH

In the selection section:

APPSEL_NAME
Is a field name in the object. “Specifying selection criteria” on page 282 describes the names that you can specify for each object type.

APPSEL_OPER
Is a comparison operator. Only *equal to* (`EQ` or `=`) is valid for EQQUSIN.

* Offset 18 (X'12'). Set this reserved field to binary zeros (X'00').

APPSEL_VALUE_OFF
Is the offset to the APPVAL section.

APPSEL_VALUE_LEN
Is the length of the APPVAL section.

APPVAL - selection value section

This section contains a selection value for the field identified in APPSEL. APPVAL is pointed to by APPSEL and must be included. One APPVAL is required for each APPSEL.

The selection value section has this format:

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	*	APPVAL	DATA SECTION ADDRESS OF FIRST DATA SECTION FOR THIS OBJECT: APPVAL_PTR=ADDR(APP) + APPSEL_VALUE_OFF
0	(0)	(See note)	*	APPVAL_DAT	DATA

Note: The field type depends on the object field name that you specify in APPSEL_NAME.

APPFLD - field section

Each field section identifies a field in the selected object that you want to update; for example, the status of an operation in the current plan. APPFLD is not used when the object name is BACKUP_EVENT but is required for all other object names. Field sections are pointed to by the APPOBJ_FLD_TRIPLET in the object section. You can specify more than one APPFLD for each APPOBJ, but all field sections for a particular APPOBJ must be in contiguous storage.

The field section has this format:

Offsets					
Dec	**Hex**	**Type**	**Len**	**Name**	**Description**
0	(0)	STRUCTURE	24	APPFLD	FIELD SECTION ADDRESS OF FIRST FIELD SECTION FOR THIS OBJECT: APPFLD_PTR= ADDR(APP) + APPOBJ_FLD_OFF
0	(0)	CHARACTER	16	APPFLD_NAME	FIELD NAME
16	(10)	SIGNED	4	APPFLD_LEN	FIELD LENGTH
20	(14)	CHARACTER	4	APPFLD_TYPE	*FIELD DATA TYPE

In the field section:

APPFLD_NAME
Is the name of the field. “Specifying object fields to update” on page 286 describes the fields that you can specify for each object type.

APPFLD_LEN
Is the length of the field and is used in identifying the value in APPDAT for this field.

APPFLD_TYPE
Is the data type. EQQUSIN ignores any value in this field.

APPDAT - data section

The data section is always the last section in the buffer. It contains the new values for the fields identified in the APPFLD sections. The values must be in the same order as their corresponding APPFLD sections. Only one APPDAT is required for each APPOBJ.

The data section has this format:

Offsets					
Dec	Hex	Type	Len	Name	Description
0	(0)	STRUCTURE	*	APPDAT	DATA SECTION ADDRESS OF FIRST DATA SECTION FOR THIS OBJECT: APPDAT_PTR=ADDR(APP) + APPOBJ_DAT_OFF
0	(0)	(See note)	*	APPDAT_DAT	DATA

Note: The field type depends on the object field name that you specify in APPFLD_NAME.

Specifying selection criteria

The field selection values that you can provide in APPSEL and APPVAL for each object type are described here. They are used to identify the instance of the object that you want to create an event for.

Selecting an operation to change the status (CP_OPER_EVENT)

You can specify these fields to identify a current plan operation for which you want to change the status:

Table 34. CP_OPER_EVENT selection fields

Field	Type	Size	Description
SUBSYSTEM_NAME	CHAR	4	Subsystem name
WS_NAME	CHAR	4	Workstation name
JOBNAME	CHAR	8	Job name
APPL_ID	CHAR	16	Application ID
OPER_NUM	BIN	15	Operation number
APPL_IA_DATE	CHAR	6	Input arrival date (YYMMDD)
APPL_IA_TIME	CHAR	4	Input arrival time (HHMM)
FORM_NUMBER	CHAR	8	Form number
CLASS	CHAR	1	SYSOUT class
OPER_TOKEN	CHAR	8	Operation token

SUBSYSTEM_NAME
The name of the tracker subsystem that the event should be reported to. If SUBSYSTEM_NAME is not specified or has the value MSTR, the event is broadcast using the subsystem interface (SSI) to all IBM Workload Scheduler for z/OS subsystems on the z/OS image where EQQUSIN is invoked.

WS_NAME
: The name of the workstation.

JOBNAME
: The name of the job that an event is being reported for.

APPL_ID
: The name of the current application.

OPER_NUM
: The number, in binary format, of the current operation. The number can have a decimal value from 1 to 255.

APPL_IA_DATE
: The input arrival date of the current occurrence in the format YYMMDD.

APPL_IA_TIME
: The input arrival time of the current occurrence in the format HHMM.

FORM_NUMBER
: Contains the printer form number for operations at printer workstations.

CLASS
: Contains the SYSOUT class for operations at printer workstations.

OPER_TOKEN
: A hexadecimal value that uniquely identifies an operation. If you stored the token set in the OPCTOKEN parameter of the operation-initiation exit (EQQUX009), you can provide this token to EQQUSIN to uniquely identify the operation. OPER_TOKEN is valid only for operations at workstations that have a user-defined destination.

  Note: As the OPCTOKEN is defined as a full-word, the OPER_TOKEN should be typed in as follows:

```
FIRST WORD    = X'00000000'
SECOND WORD   = OPCTOKEN FROM EXIT EQQUX009
```

Note:

1. You must specify at least OPER_TOKEN, or WS_NAME with either JOBNAME or APPL_ID.
2. If you do not provide enough information to uniquely identify the operation, IBM Workload Scheduler for z/OS must determine the most applicable operation to update. IBM Workload Scheduler for z/OS considers only operations in status R, A, *, S, I, or E when selecting the operation. IBM Workload Scheduler for z/OS selects the operation to update by investigating these characteristics in the stated order:
   a. The operation has priority 9.
   b. Earliest latest start time.
   c. Priority 8-1.
   d. Input arrival time specified for the operation or the occurrence input arrival if the operation does not have input arrival specifically defined.

   So from the operations that match the selection criteria, the operation with priority 9 is updated. If more than one operation has priority 9, the operation with the earliest latest start time is updated. If latest start is equal, the operation with the highest priority is updated. If priority is equal, the operation with the earliest input arrival time is updated. If input arrival is also equal, the update is performed on a first-in-first-out basis.

Selecting a special resource (CP_SR_EVENT)

You can specify these fields to identify a current plan special resource:

Table 35. CP_SR_EVENT selection fields

Field	Type	Size	Description
SUBSYSTEM_NAME	CHAR	4	Subsystem name
SR_NAME	CHAR	44	Name of special resource

SUBSYSTEM_NAME
: The name of the tracker subsystem that the event should be reported to. If SUBSYSTEM_NAME is not specified or has the value MSTR, the event is broadcast using the subsystem interface (SSI) to all IBM Workload Scheduler for z/OS subsystems on the z/OS image where EQQUSIN is invoked.

SR_NAME
: The name of the special resource.

Selecting an operation to supply user data (CP_OPINFO_EVENT)

You can specify these fields to identify a current plan operation for which you want to update the USERDATA field:

Table 36. CP_OPINFO_EVENT selection fields

Field	Type	Size	Description
SUBSYSTEM_NAME	CHAR	4	Subsystem name
WS_NAME	CHAR	4	Workstation name
JOBNAME	CHAR	8	Job name
APPL_ID	CHAR	16	Application ID
OPER_NUM	BIN	15	Operation number
APPL_IA_DATE	CHAR	6	Input arrival date (YYMMDD)
APPL_IA_TIME	CHAR	4	Input arrival time (HHMM)
FORM_NUMBER	CHAR	8	Form number
CLASS	CHAR	1	SYSOUT class
STATUS	CHAR	1	Operation status

SUBSYSTEM_NAME
: The name of the tracker subsystem that the event should be reported to. If SUBSYSTEM_NAME is not specified or has the value MSTR, the event is broadcast using the subsystem interface (SSI) to all IBM Workload Scheduler for z/OS subsystems on the z/OS image where EQQUSIN is invoked.

WS_NAME
: The name of the workstation.

JOBNAME
: The name of the job that an event is being reported for.

APPL_ID
: The name of the current application.

OPER_NUM
The number, in binary format, of the current operation. The number can have a decimal value from 1 to 255.

APPL_IA_DATE
The input arrival date of the current occurrence in the format YYMMDD.

APPL_IA_TIME
The input arrival time of the current occurrence in the format HHMM.

FORM_NUMBER
Contains the printer form number for operations at printer workstations.

CLASS
Contains the SYSOUT class for operations at printer workstations.

STATUS
The current status of the operation.

Note: If the OPINFOSCOPE keyword of the JTOPTS statement is IP, which is the default, you must specify WS_NAME. If OPINFOSCOPE is ALL, you must specify either APPL_ID or JOBNAME. The OPINFOSCOPE keyword is described in the list of JTOPTS "Parameters" on page 80.

Selecting an IBM Workload Scheduler for z/OS data set (BACKUP_EVENT)

You can specify these fields to identify an IBM Workload Scheduler for z/OS data set:

Table 37. BACKUP_EVENT selection fields

Field	Type	Size	Description
SUBSYSTEM_NAME	CHAR	4	Subsystem name
FILENAME	CHAR	2	Name of data set (CP or JS)

SUBSYSTEM_NAME
Is the name of the tracker subsystem that the event should be reported to. If SUBSYSTEM_NAME is not specified or has the value MSTR, the event is broadcast using the subsystem interface (SSI) to all IBM Workload Scheduler for z/OS subsystems on the z/OS image where EQQUSIN is invoked.

FILENAME
Is either CP (current plan) or JS (JCL repository).

Note: The APPSEL values are sufficient to create a BACKUP event. The APPFLD and APPDAT sections are not used for this event type.

Selecting a workstation (CP_WS_EVENT)

You can specify these fields to identify a current plan workstation:

Table 38. CP_WS_EVENT selection fields

Field	Type	Size	Description
SUBSYSTEM_NAME	CHAR	4	Subsystem name
DESTINATION	CHAR	8	Destination name
WS_NAME	CHAR	4	Workstation name

SUBSYSTEM_NAME
Is the name of the tracker subsystem that the event should be reported to. If SUBSYSTEM_NAME is not specified or has the value MSTR, the event is broadcast using the subsystem interface (SSI) to all IBM Workload Scheduler for z/OS subsystems on the z/OS image where EQQUSIN is invoked.

DESTINATION
Is the name specified in the destination field of the workstation. This must be a user-defined destination.

WS_NAME
Is the workstation name.

Note: You must select at least WS_NAME for a workstation event.

Specifying object fields to update

The fields that you can update in each object type and the new values that you can provide are described here:

Updating operation status (CP_OPER_EVENT)

You can update the following fields in a current plan operation:

Table 39. Operation fields that you can update through CP_OPER_EVENT

Field	Type	Size	Description
STATUS	CHAR	1	New status (C, E, I, Q, S, T, or X)
ERROR_CODE	CHAR	4	Error code (for new status C or E)
ACT_DUR	CHAR	4	Actual duration HHMM (for new status C or E)
ACT_END_TIME	CHAR	4	Actual end time HHMM (for new status C or E)
EV_CREATION_DATE	CHAR	4	Event creation date (01YYDDDF)
EV_CREATION_TIME	BIN	31	Event creation time (100 * seconds)
JOB_NUMBER	CHAR	5	Job number

STATUS
New status of the operation. The following values are valid:

C Set the status of the operation to complete.

E Set the status of the operation to ended-in-error.

I Set the status of the operation to interrupted.

Q Set the extended status of a started operation to Q to indicate that the operation is queued awaiting execution.

S Set the status of the operation to started.

T Set the extended status of a started operation to S to indicate that the operation is executing.

X Reset the current status for this operation.

ERROR_CODE
Error code for an operation that is reported as ended-in-error.

ACT_DUR
Duration, in hours and minutes, of an operation that is reported as

complete or ended-in-error. The operation duration cannot be 0000. If you set it to 0000 (0 hours, 0 minutes), a default duration value of 1 minute is used instead.

ACT_END_TIME
Time the operation is reported as complete or ended-in-error.

EV_CREATION_DATE
Date of the event that is being reported. If the field contains all binary zeros (X'00'), IBM Workload Scheduler for z/OS uses the current date.

EV_CREATION_TIME
Time of the event that is being reported. If the field contains all binary zeros (X'00'), IBM Workload Scheduler for z/OS uses the current time.

JOB_NUMBER
A number that you can provide for the job. JOB_NUMBER is valid only for operations at general automatic workstations and workstations that have a user-defined destination. Do not specify JOB_NUMBER for operations that are submitted through a tracker.

Note:

- You must specify at least STATUS. The remaining field names are optional.
- To use the value set in ERROR_CODE even if the operation status is set to Complete, ensure that the ERROR_CODE field is properly set. For a Complete operation, set ERROR_CODE to 0 or blank unless it is differently required. If ERROR_CODE is set to a value different from 0, it is processed as the original return code even if the operation status is set to Complete.

  To enable ERROR_CODE processing on Complete operations, you must have set USINRC=YES in the JTOPTS statement.

Updating a special resource (CP_SR_EVENT)

You can update these fields in a current plan special resource:

Table 40. Special resource fields that you can update through CP_SR_EVENT

Field	Type	Size	Description
AVAILABLE	CHAR	1	Availability (Y|N|K|R)
QUANTITY	BIN	31	Number available (1 to 999 999)
QUANTITY_OPTION	CHAR	8	Quantity option (KEEP|RESET)
DEVIATION	BIN	31	Number to deviate (-999 999 to 999 999)
DEVIATION_OPTION	CHAR	8	Deviation option (KEEP|RESET)
CREATE	CHAR	1	Create resource if undefined (Y|N)

AVAILABLE
Is the availability status of the special resource. Y indicates that the availability status of the resource should be set to YES; N indicates that the status should be NO. R (RESET) sets the status to the planned availability status in the current plan. K (KEEP), the default, does not change the status.

QUANTITY
Is a numeric value (1–999 999) that updates the Quantity field in the special resource, which overrides interval and default values. QUANTITY and QUANTITY_OPTION fields are mutually exclusive. If you specify both fields, the event is ignored.

QUANTITY_OPTION
: Is either KEEP or RESET. Specify RESET to set the amount to the scheduled value in the current plan or KEEP, the default, to leave the quantity unchanged. If you specify QUANTITY and QUANTITY_OPTION, the event is ignored.

DEVIATION
: Is a numeric value, -999 999 to 999 999, which lets you make a temporary change to the quantity. Deviation is an amount to be added to (positive number) or subtracted from (negative number) the current quantity. For example, if you specify -2 and the current quantity is 10, the total amount that operations can allocate reduces to 8. DEVIATION and DEVIATION_OPTION fields are mutually exclusive. If you specify both fields, the event is ignored.

DEVIATION_OPTION
: Is either KEEP or RESET. Specify RESET to set the deviation to zero. KEEP, the default, does not alter the deviation. If you specify DEVIATION and DEVIATION_OPTION, the event is ignored.

CREATE
: Specifies if IBM Workload Scheduler for z/OS should create a resource in the current plan if the resource does not exist. NO indicates that the resource should not be added to the resource definitions of the receiving IBM Workload Scheduler for z/OS subsystem. If the resource is already defined in the receiving subsystem, NO has no effect. You can specify NO if the resource is being used only as a means to generate an event for ETT: the event is generated even if the resource does not exist.

  If YES is specified and the DYNAMICADD keyword of the RESOPTS initialization statement is set to YES or EVENT, a resource definition is created in the receiving IBM Workload Scheduler for z/OS subsystem if the resource is not already defined.

Note: When you set the quantity or availability of a resource through EQQUSIN (or other interfaces such as the SRSTAT TSO command or the MCP dialog), the specified value lasts over interval boundaries, even though the next interval can specify a different value. Specify RESET to restore the planned value.

Updating an operation user-data (CP_OPINFO_EVENT)

You can update this field in a current plan operation with user-data information:

Table 41. Operation field that you can update through CP_OPINFO_EVENT

Field	Type	Size	Description
USERDATA	CHAR	16	User data (free form text)

USERDATA
: Is the 16-character user-data information that is to be updated for the specified operation.

Updating a workstation (CP_WS_EVENT)

You can update these fields in a current plan workstation:

Table 42. Workstation fields that you can update through CP_WS_EVENT

Field	Type	Size	Description
WS_STATUS	CHAR	1	Workstation status

Table 42. Workstation fields that you can update through CP_WS_EVENT (continued)

Field	Type	Size	Description
STARTED_FAIL_OPT	CHAR	1	Failure option for started operations (R, L, or E)
REROUTE_OPT	CHAR	1	Reroute option (Y or N)
ALT_WS	CHAR	4	Alternate workstation name

WS_STATUS
Is the status you want reported for the workstation:

A	Active
O	Offline
F	Failed.

STARTED_FAIL_OPT
When the workstation status is set to offline or failed, you can specify what IBM Workload Scheduler for z/OS should do with operations that are currently in started status at this destination (workstation):

R	Restart operations automatically on the alternate workstation
L	Leave the operations in started status
E	Set all started operations to ended-in-error.

REROUTE_OPT
When the workstation status is set to offline or failed, you can specify Y for operations to be rerouted to the alternate workstation, or N for no rerouting if you want to leave the operations at the inactive workstation.

ALT_WS
When the workstation status is set to offline or failed, you can specify an alternate workstation where rerouted operations should be started.

Note:
1. You must specify at least WS_STATUS.
2. If the value provided for WS_STATUS is equal to the current status, the event is ignored.

Return codes and reason codes generated by EQQUSIN

Your program can test the results of the call to EQQUSIN by inspecting the return code and the reason code in the APP section of the buffer.

The APP_RETCODE field can contain one of these codes:

0	Execution successful.
12	Execution unsuccessful; the buffer is invalid. No event has been created.

The APP_RSNCODE field can contain one of these codes:

0	Execution successful.
4	Buffer shorter than APP.
8	Eye-catcher in APPDESC field is invalid. It must be APP.
12	Version number in APPVER field is invalid. It must be 02.
16	Type in APPTYPE field is invalid. It must be DIA.
20	APPTOTSZ invalid.
24	Data type invalid. Specify only CREATE for EQQUSIN.
28	Object section not within buffer.
32	Object section overlays APP.
36	Selection section not within buffer.
40	Selection section overlays APP or object section.

44	Field section not within buffer.
48	Field section overlays APP or object section.
52	Required key not complete.
56	Invalid object name in OBJ section.
60	Invalid field name in FLD section.
64	Invalid field name in SEL section.
68	Invalid APPTOKEN value (duplicate).

If an error occurs when IBM Workload Scheduler for z/OS processes the event, check the message log (EQQMLOG) of the controller for information about the error.

Using individual IBM Workload Scheduler for z/OS subroutines

Read the following information to understand how to use individual IBM Workload Scheduler for z/OS subroutines.

At entry to these subroutines, register 1 must point to a parameter list. This parameter list consists of a sequence of 4-byte addresses to the parameters. The following sections describe the parameters in detail for each subroutine.

Note: APAR PQ74854 has changed the addressing mode of the following subroutines from Amode(24) to Amode(ANY), which provides the possibility to use the 31-bit addressing mode before calling z/OS subroutines.

Using EQQUSINB

You use EQQUSINB to request IBM Workload Scheduler for z/OS to copy a resource data set.

Invocation requirements

EQQUSINB has these invocation requirements:

Authorization
APF authorized, or supervisor state, or PSW key 0–7.

Dispatchable unit mode
Task mode.

Amode
24-bit, or ANY if APAR PQ74854 was applied.

ASC mode
Primary or access register (AR).

Interrupt status
Enabled for I/O and external interrupts.

Locks No locks held.

Control parameters
All parameters must be addressable by the caller and in the primary address space.

EQQUSINB parameters

The calling program must pass all these parameters to the subroutine. Initialize RETCODE to zero in the call; it is set by EQQUSINB in the return.

EQQUSINB parameters

```
DATASET   DS   CL2    (Resource data set name)
SUBSYS    DS   CL4    (Subsystem name)
RETCODE   DS   F      (EQQUSINB return code)
```

data set
: Defines the resource data set to be backed up. Valid values are:

CP	The current plan data set
JS	The JCL repository data set.

SUBSYS
: Is the name of the Tivoli OPC tracker subsystem that this event should be reported to. If SUBSYS is blank, the event is broadcast to all IBM Workload Scheduler for z/OS subsystems defined on the z/OS system where EQQUSINB is invoked.

RETCODE
: Is set by EQQUSINB and can have one of these values:

0	Normal return. The event has been reported to IBM Workload Scheduler for z/OS.
8	Error return. There is an error in the information that was passed to EQQUSINB. No event has been reported to IBM Workload Scheduler for z/OS.

Using EQQUSINO

You use EQQUSINO to request IBM Workload Scheduler for z/OS to feed back information to the user data of a current plan operation. The current status of the operation must be R, A, *, S, I, E, C, or W. You can update an operation in status C or W only if the OPINFOSCOPE keyword of JTOPTS has the value ALL. OPINFOSCOPE is described in more detail in the list of JTOPTS “Parameters” on page 80.

Invocation requirements

EQQUSINO has these invocation requirements:

Authorization
: APF authorized, or supervisor state, or PSW key 0–7.

Dispatchable unit mode
: Task mode.

Amode
: 24-bit, or ANY if APAR PQ74854 was applied.

ASC mode
: Primary or access register (AR).

Interrupt status
: Enabled for I/O and external interrupts.

Locks No locks held.

Control parameters
: All parameters must be addressable by the caller and in the primary address space.

EQQUSINO parameters

The calling program must pass all these parameters to the subroutine. If the OPINFOSCOPE keyword is IP, which is the default, WSNAME is a required parameter. If OPINFOSCOPE is ALL, you must specify valid values for either the

ADID or JOBNAME parameters. The other parameters can be blank (zero for OPNUM). Initialize RC to zero in the call; it is set by EQQUSINO in the return.

EQQUSINO parameters

```
WSNAME    DS   CL4    (Workstation name)
JOBNAME   DS   CL8    (Job name)
ADID      DS   CL16   (Name of current application)
OPNUM     DS   H      (Operation number or zero)
OCIA      DS   CL10   (Occ input arrival, YYMMDDHHMM, or blank)
FORM      DS   CL8    (SYSOUT form number or blank)
CLASS     DS   CL1    (Job or SYSOUT class or blank)
SUBSYS    DS   CL4    (Name of the Tivoli OPC tracker or blank)
USERDATA  DS   CL16   (User data to feed back to the operation)
RC        DS   F      (EQQUSINO return code)
```

WSNAME
Is the name of the workstation defined for the operation.

JOBNAME
Is the job name defined for the operation you want to update.

ADID Is the application ID that contains the operation.

OPNUM
Is the number, in hexadecimal format, of the current operation. You can specify X'0000' or a number in the range X'0001' to X'00FF' (decimal 1 to 255).

OCIA Is the input arrival date and time of the current occurrence.

FORM
Contains the printer form name for operations at printer workstations.

CLASS
Contains the job class or SYSOUT class defined for the operation.

SUBSYS
Is the name of the Tivoli OPC tracker subsystem that this event should be reported to. If SUBSYS is blank, the event is broadcast to all IBM Workload Scheduler for z/OS subsystems defined on the z/OS system where EQQUSINO is invoked.

USERDATA
Is the 16-character user-data information that is to be updated for the specified operation.

RC Is set by EQQUSINO and can have one of these values:

0 Normal return. The event has been reported to IBM Workload Scheduler for z/OS.

8 Error return. There is an error in the information that was passed to EQQUSINO; no event has been reported to IBM Workload Scheduler for z/OS.

Note: If you do not provide enough information to uniquely identify the operation, IBM Workload Scheduler for z/OS must determine the most applicable operation to update. IBM Workload Scheduler for z/OS considers first only operations in status R, A, *, S, I, or E when selecting the operation. IBM Workload Scheduler for z/OS selects the operation to update by investigating these characteristics in the stated order:

1. The operation has priority 9.

2. Earliest latest start time.
3. Priority 8-1.
4. Input arrival time specified for the operation or the occurrence input arrival if the operation does not have input arrival specifically defined.
5. Longest in Ready status.

So if you define only the WSNAME parameter and IBM Workload Scheduler for z/OS determines that there is more than one operation in the current plan for that workstation, the operation with priority 9 is updated. If more than one operation has priority 9, the operation with the earliest latest start time is updated. If latest start is equal, the operation with the highest priority is updated. If priority is equal, the operation with the earliest input arrival time is updated.

If no match has been found for operations in status R, A, *, S, I, or E, IBM Workload Scheduler for z/OS uses the value of the OPINFOSCOPE keyword of JTOPTS to determine if operations in status C and W are also considered. OPINFOSCOPE can have the value IP (in progress) or ALL. Operations in status C and W are considered only if the value is ALL. The operation with the earliest latest-start-time is selected.

Using EQQUSINS

You use EQQUSINS to request IBM Workload Scheduler for z/OS to change the availability of a special resource.

Invocation requirements

EQQUSINS has these invocation requirements:

Authorization
APF authorized, or supervisor state, or PSW key 0–7.

Dispatchable unit mode
Task mode.

Amode
24-bit, or ANY if APAR PQ74854 was applied.

ASC mode
Primary or access register (AR).

Interrupt status
Enabled for I/O and external interrupts.

Locks No locks held.

Control parameters
All parameters must be addressable by the caller and in the primary address space.

EQQUSINS parameters

The calling program must pass all these parameters to the subroutine. Initialize RC to zero in the call; it is set by EQQUSINS in the return.

EQQUSINS parameters

```
SRNAME    DS   CL44   (Name of the special resource)
SUBSYS    DS   CL4    (Subsystem name)
AINDIC    DS   CL1    (Availability indicator, Y or N)
RC        DS   F      (EQQUSINS return code)
```

SRNAME
Defines the name of the special resource that should be updated.

SUBSYS
Is the name of the Tivoli OPC tracker subsystem that this event should be reported to. If SUBSYS is blank, the event is broadcast to all IBM Workload Scheduler for z/OS subsystems defined on the z/OS system where EQQUSINS is invoked.

AINDIC
Specifies if the special resource should be set to available (Y) or unavailable (N).

RC Is set by EQQUSINS and can have one of these values:

0 Normal return. The event has been reported to IBM Workload Scheduler for z/OS.

8 Error return. There is an error in the information that was passed to EQQUSINS. No event has been reported to IBM Workload Scheduler for z/OS.

Using EQQUSINT

You use EQQUSINT to request IBM Workload Scheduler for z/OS to change the status of an operation at a workstation. The workstation can be any type except a workstation with the nonreporting attribute.

Invocation requirements

EQQUSINT has these invocation requirements:

Authorization
APF authorized, or supervisor state, or PSW key 0–7.

Dispatchable unit mode
Task mode.

Amode
24-bit, or ANY if APAR PQ74854 was applied.

ASC mode
Primary or access register (AR).

Interrupt status
Enabled for I/O and external interrupts.

Locks No locks held.

Control parameters
All parameters must be addressable by the caller and in the primary address space.

EQQUSINT parameters

The calling program must pass all these parameters to the subroutine. Initialize RETCODE to zero in the call; it is set by EQQUSINT in the return.

EQQUSINT parameters

```
TYPE      DS   CL1    (Event type)
WSNAME    DS   CL4    (Workstation name)
JOBNAME   DS   CL8    (Job name)
ADID      DS   CL16   (Name of current application)
OPNUM     DS   H      (Operation number)
OCIA      DS   CL10   (Input arrival of current occurrence, YYMMDDHHMM)
OPDUR     DS   CL4    (Operation duration, HHMM)
OPERR     DS   CL4    (Error code)
FORM      DS   CL8    (SYSOUT form number)
SCLASS    DS   CL1    (SYSOUT class)
SUBSYS    DS   CL4    (Subsystem name)
EVTIME    DS   F      (Event time, 100*secs)
EVDATE    DS   CL4    (Event date, 0nYYDDDF)
RETCODE   DS   F      (EQQUSINT return code)
```

TYPE Defines the reason that EQQUSINT is called. These values are valid:

C Set the status of the operation to complete.

E Set the status of the operation to ended-in-error.

I Set the status of the operation to interrupted.

Q Set the extended status of a started operation to Q to indicate that the operation is queued awaiting execution.

S Set the status of the operation to started.

T Set the extended status of a started operation to S to indicate that the operation is executing.

X Reset the current status for this operation.

WSNAME
Is the name of the workstation.

JOBNAME
Is the name of the job that an event is being reported for.

ADID Is the name of the current application.

OPNUM
Is the number, in hexadecimal format, of the current operation. You can specify 0000 or a number in the range 0001 to 00FF (decimal 1 to 255).

OCIA Is the input arrival date and time of the current occurrence.

OPDUR
Is the duration, in hours and minutes, of an operation that is reported as complete. The operation duration cannot be 0000. If you set it to 0000 (0 hours, 0 minutes), a default duration value of 1 minute is used instead.

OPERR
Is the error code for an operation that is reported as ended-in-error.

FORM
Contains the printer form name for operations at printer workstations.

SCLASS
Contains the SYSOUT class for operations at printer workstations.

SUBSYS
Is the name of the Tivoli OPC tracker subsystem that this event should be reported to. If SUBSYS is blank, the event is broadcast to all IBM Workload Scheduler for z/OS subsystems defined on the z/OS system where EQQUSINT is invoked.

EVTIME
: Contains the time of the event that is being reported. If the field contains all binary zeros, IBM Workload Scheduler for z/OS uses the current time.

EVDATE
: Contains the date of the event that is being reported. If the field contains all binary zeros, IBM Workload Scheduler for z/OS uses the current date. If n = 0, year is 19YY. If n = 1, year is 20YY.

RETCODE
: Is set by EQQUSINT and can have one of these values:

0	Normal return. The event has been reported to IBM Workload Scheduler for z/OS.
8	Error return. There is an error in the information that was passed to EQQUSINT, and no event has been reported to IBM Workload Scheduler for z/OS.

Note:
1. You must specify valid values for the TYPE and WSNAME parameters and either the JOBNAME or ADID parameters. The remaining values can be initialized to zeros or blanks.
2. OPDUR is processed only if TYPE has value C.
3. OPERR is processed only if TYPE has value E.
4. If you do not provide enough information to uniquely identify the operation and IBM Workload Scheduler for z/OS finds more than one operation that matches the criteria you specified, IBM Workload Scheduler for z/OS determines the most applicable operation to update. IBM Workload Scheduler for z/OS chooses the most applicable operation by investigating these characteristics in the stated order:
   a. The operation has priority 9.
   b. Earliest latest start time.
   c. Priority 8-1.
   d. Input arrival time specified for the operation, or the occurrence input arrival if the operation does not have input arrival specifically defined.

   Therefore, if you define only the WSNAME parameter and IBM Workload Scheduler for z/OS determines that there is more than one operation in the current plan for that workstation in status R, A, *, S, I, or E, then the operation with priority 9 is updated. If more than one operation specifies priority 9, then the operation with the earliest latest start time is updated. If latest start is equal, then the operation with the highest priority is updated. If priority is equal, the operation which specifies the earliest input arrival time is updated. If input arrival is also equal, the update is performed on a first-in-first-out basis.

Using EQQUSINW

You use EQQUSINW to generate a workstation status event for a particular workstation that specifies a user-defined destination. Workstation status events can be generated for active, failed, or offline conditions.

Invocation requirements

EQQUSINW has these invocation requirements:

Authorization
: APF authorized, or supervisor state, or PSW key 0–7.

Dispatchable unit mode
Task mode.

Amode
24-bit, or ANY if APAR PQ74854 was applied.

ASC mode
Primary or access register (AR).

Interrupt status
Enabled for I/O and external interrupts.

Locks No locks held.

Control parameters
All parameters must be addressable by the caller and in the primary address space.

EQQUSINW parameters

The calling program must pass all these parameters to the subroutine. Any parameter except STATUS and WSNAME can be left blank. Initialize RC to zero in the call; it is set by EQQUSINW in the return.

EQQUSINW parameters

```
DUMMY     DS   CL8    (Reserved parameter, value ignored)
WSNAME    DS   CL4    (Workstation name must be specified)
STATUS    DS   CL1    (Workstation status)
STARTOPS  DS   CL1    (Action for started operations or blank)
REROUTE   DS   CL1    (Reroute indicator or blank)
ALTWS     DS   CL4    (Alternate workstation name or blank)
SUBSYS    DS   CL4    (Name of the tracker subsystem or blank)
RC        DS   F      (EQQUSINW return code)
```

DUMMY
A parameter reserved for future use. Any value supplied is ignored by the subroutine.

WSNAME
The workstation name.

STATUS
The status you want reported for the workstation, where:

A	Active
O	Offline
F	Failed.

STARTOPS
When the workstation status is set to offline or failed, you can specify what IBM Workload Scheduler for z/OS should do with operations that are currently in started status on the destination, or workstation, where:

R	Restart operations automatically on the alternate workstation.
L	Leave the operations in started status.
E	Set all started operations to ended-in-error.

REROUTE
When the workstation status is set to offline or failed, you can specify R for operations to be rerouted to the alternate workstation or L for no rerouting; that is, you want to leave the operations at the inactive workstation.

ALTWS
When the workstation status is set to offline or failed, you can specify the alternate workstation where reroute operations should be started.

SUBSYS
The name of the tracker subsystem that this event should be reported to. If SUBSYS is blank, the event is broadcast to all tracker subsystems defined on the z/OS system where EQQUSINW is invoked.

RC Is set by EQQUSINW and can have one of these values:

0 Normal return. The event has been reported to IBM Workload Scheduler for z/OS.

8 Error return. There is an error in the information that was passed to EQQUSINW, and no event has been reported to IBM Workload Scheduler for z/OS.

Note: If the value provided in the STATUS parameter is equal to the current status, the event is ignored. A value must be supplied for the WSNAME and STATUS parameters.

Chapter 7. Using the Job Completion Checker

In the following description, a *job* refers to either a batch job or a started task.

IBM Workload Scheduler for z/OS uses the job completion code to determine if an operation has completed normally. The code is either the highest return code of all completed steps or the return code of the last completed step, depending on what you have specified on the RETCODE keyword of the EWTROPTS statement. In some cases, however, success or failure cannot be determined from this return code alone.

In these cases, you can use the job completion checker (JCC) to determine if a job has ended normally. The JCC can scan the SYSOUT data set for a particular job, and then set the error status, depending on the results of this scan. Because the JCC has more information about the job, it is better equipped to decide whether a job has ended normally.

See "Determining the success or failure of a job" on page 172, which describes how IBM Workload Scheduler for z/OS determines the next status of an operation when a job or started-task ends.

Note: The JCC process logic is not applied when the failing job has been obtained by restarting an operation at step or job level and the failure is determined by the EQQCLEAN step ending with RC>=8. See also "Determining the success or failure of a job" on page 172.

JCC message tables

The JCC uses the CHKCLASS keyword you specified on the JCCOPTS statement (see CHKCLASS in the list of JCCOPTS "Parameters" on page 74) to decide which SYSOUT classes to scan for each ending job. You determine how the SYSOUT data is to be scanned by creating JCC *message tables*. Each record in a SYSOUT data set is treated as a message. The JCC message tables determine which character string to search for in each message and what to do if the string is found. You define each message table as a member in the EQQJCLIB library.

The JCC uses two types of message tables:

- Any job can have a *job-specific* message table. The member name in the EQQJCLIB library MUST be the same as the job name.
- The *general* message table is used for all jobs. The general message table must be located in the EQQGJCCT member of the EQQJCLIB library. JCC initialization will fail if the general table cannot be found.

When the JCC starts to process the output for a job, the EQQJCLIB is searched to determine if a job-specific message table is available for the job. If a job-specific message table is found, it is used in conjunction with the general message table. Each SYSOUT record is evaluated against the message tables in isolation, starting with the first record. The job-specific message table is used first, followed by the general message table. The SYSOUT record is evaluated against EVERY condition defined in the message table, as long as the `CA=` does not stop checking.

If a match is found and the checking action specifies STOP or ESTOP, no further JCC processing occurs for the job. If the matching criteria is started by a MULTSTA and a match is found against the MULTSTA, JCC does not treat this as a match until a match has been found for the same SYSOUT record on a subsequent MULTMSG. If the MULTSTA matches but no matching MULTMSG is found, the JCC takes the action defined in the corresponding MULTEND entry. If a MULTSTA M= value has been matched, no subsequent MULTSTA M= with the same M= value will be used, regardless of whether the checking action specified STOP or ESTOP.

If a match is found and the checking action permits processing to continue, or if no match is found in either the job-specific or general tables, the JCC takes the action defined in the ENDTAB entry. If there is no ENDTAB entry, JCC continues to process the SYSOUT from the next record.

When there is a match with a NORMMSG or a MULTSTA, the SYSOUT record is *consumed*. This means that no further processing occurs for that SYSOUT record. Therefore, if there is a match in a job-specific table, processing for that SYSOUT record stops immediately, and the general table is not searched. Processing will continue at the next SYSOUT record only if the CA= allows checking to continue.

A SYSOUT data set is created by the z/OS system or by a user program. All records are checked in SYSOUT data sets created by the z/OS system. The value of the USYSOUT keyword of JCCOPTS determines if a user SYSOUT data set is checked. The value of the UMAXLINE keyword determines how many lines of a user SYSOUT data set are checked.

All records in all SYSOUT data sets are passed to the tracker exit, EQQUX005 (the SYSOUT archiving exit). You can use this exit to copy SYSOUT data sets to a data set that resides on a disk or tape.

Note:

1. The JCC is a tracker function and is, therefore, independent of the contents of the controller current-plan data set. The JCC processes all jobs for which a job termination (3P) event is created in the event data set, regardless of whether the job is defined in the current plan. To prevent the JCC from processing a job or a class of jobs, you must use the tracker event-filtering exit, EQQUX004.
2. Because it is possible to send JES2 job SYSOUT, or parts of the SYSOUT, to several NJE nodes, more than one job termination (A3P) event could be produced for the same job. Each event could also have different job-completion-code information, depending on the output sent to a particular node and the checking that the JCC performs at that node. The status assigned to the operation depends on which of the A3P events is first processed by the controller. You should ensure, therefore, that the value FINAL is used for the OUTPUTNODE keyword of the JTOPTS statement. FINAL is the default value. If the JESYSMSG part (previously $SYSMSGS, DSID=4) of SYSOUT is copied to several final destination nodes where the JCC is active, or you specify the value ANY for OUTPUTNODE, the resulting status of the corresponding operation will be unpredictable. The OUTPUTNODE keyword is described in the list of JCCOPTS "Parameters" on page 80.
3. The technique described in note 2 is not used in a JES3 environment. If you send the output from a JES3 job to different NJE nodes where the JCC is active, the JCC should perform the same checking at each node. Otherwise, the resulting status of the corresponding operation will be unpredictable.

Incident logging function

In addition to determining the ending status of batch jobs by scanning SYSOUT data sets created by the job, the JCC has an incident-logging function. The JCC can be directed to record error conditions in an incident file. The incident file is a sequential file that is updated by a tracker exit, EQQUX006 (the incident-record-create exit). The incident-record-create exit is a required exit. You can use the standard exit that is shipped with the tracker or you can replace it with your own exit.

The incident log data set is never input to any tracker function. It is referenced only by the incident-record-create exit. It can, therefore, be shared by several JCC tasks running on the same or different systems. You can also update and even reallocate the data set manually while the JCC is active because the JCC reallocates the data set each time it is to be updated.

Defining message tables using EQQJCCT

You can create message tables by assembling a message-table-definition file that consists of one or more invocations of the EQQJCCT assembler macro. The output of the assembly job is the message table. You should save this table in a JCC message-table library.

Syntax

```
►►──EQQJCCT──┬─────────────────────────┬──┬─────────────────────────────┬──────────────►
             └─M──=──'──message text──'─┘  │        ┌─1──────────────┐   │
                                           └─S──= ──┴─start position─┴──┘

►──┬──────────────────────┬──┬────────────────────┬──┬───────────────────────────┬─────►
   │      ┌─NORMMSG────┐  │  │        ┌─CONT──┐   │  │          ┌──────────────┐ │
   └─T──= ─┼─MULTSTA────┼──┘  └─CA──= ─┼─CHECK─┼───┘  └─EID──= ──┴─error code───┴─┘
          ├─MULTEND────┤              ├─ERROR─┤
          ├─MULT2STA───┤              ├─ESTOP─┤
          ├─MULTMSG────┤              └─STOP──┘
          ├─SKIPSTA────┤
          ├─SKIPEND────┤
          ├─SKIP──nnn──┤
          ├─SKIPDS─────┤
          └─ENDTAB─────┘

►──┬─────────────────────────────────────┬─────────────────────────────────────────────►◄
   │         ┌────────────────────────┐  │
   └─TID──= ─┴─tracking identifier────┴──┘
```

When coding the EQQJCCT macro, you must follow the IBM assembler language syntax rules. These rules require that you delimit the macro name, EQQJCCT, by one or more blanks; that you place a continuation character (any non-blank character) in column 72 of any statement with a continuation line; and that you start continuation lines in column 16.

Parameters

M='*message text***'**
: Defines a character string that the JCC attempts to find in each SYSOUT

record. The character string must be enclosed in quotation marks. The maximum size of the string is 51 bytes. The M keyword is required for all types except MULTEND or ENDTAB.

M-Example

```
EQQJCCT M='IEF452I'
```

S=*start position***|**__**1**__

Defines the position in the SYSOUT record of the first character of the *message text* character string. Valid values for start position are from 0 to 132. The value 0 indicates that the message text can appear anywhere in the first 132 positions in the SYSOUT record.

T=*entry type***|**__**NORMMSG**__

Defines the message-table entry type. These entry types are supported:

NORMMSG
: Normal entry type. Stop processing the current SYSOUT record when a match is found. Then read the next SYSOUT record and check it, starting over with the first message-table entry.

MULTSTA
: Start of a sequence of related, multiple-condition table entries. It must be paired with a MULTEND statement.

MULTEND
: Defines the last of a sequence of related table entries that was started by a MULTSTA entry.

MULT2STA
: Defines an additional condition that must be fulfilled by a SYSOUT record matched by a MULTSTA entry. There can be only one MULT2STA entry for each MULTSTA entry.

MULTMSG
: Defines an additional condition that must be fulfilled by a SYSOUT record matched by a MULTSTA entry. There can be several MULTMSG entries for each MULTSTA entry. If a MULT2STA entry has been defined, the SYSOUT record is treated as a match only if it fulfills all three conditions: MULTSTA, MULT2STA, and MULTMSG.

SKIPSTA
: Start of a sequence of related skip-definition entries. *Skipping* means that a SYSOUT record is read but not checked. SKIPSTA must be followed by at least one SKIPEND statement.

SKIPEND
: Defines when a forward skip that was started by a SKIPSTA entry should end. There can be several SKIPEND entries for each SKIPSTA entry. Skipping stops when a SYSOUT record is found that matches one of the SKIPEND entries.

SKIP*nnn*
: Defines a forward skip of a fixed number of records. The number is specified as *nnn* in the SKIP*nnn* keyword. The number *nnn* must be 3 digits, 001 to 999.

SKIPDS
: Defines a forward skip of the remaining records in the current SYSOUT data set.

ENDTAB
Defines how records that have not been matched by any preceding table entries should be handled. If specified, the ENDTAB entry must be the last entry in a message table definition.

Here are some examples of the **T** parameter:

T-Example 1

```
EQQJCCT S=1,M='IEF375I'
```

Example 1 shows the default used for normal message entries.

T-Example 2

```
EQQJCCT T=MULTSTA,S=1,M='IEF285I'      DEALLOCATION
EQQJCCT T=MULTMSG,S=56,M='KEPT'
EQQJCCT T=MULTMSG,S=56,M='DELETED'
EQQJCCT T=MULTMSG,S=56,M='UNCATALOGED'
EQQJCCT T=MULTEND
```

In Example 2, the SYSOUT line is scanned for IEF285I. If IEF285I is found, the SYSOUT line is scanned for KEPT, DELETED, and UNCATALOGED.

T-Example 3

```
EQQJCCT T=MULTSTA,S=1,M='IEC501'       MOUNT
EQQJCCT T=MULT2STA,S=0,M='PRIVAT'
EQQJCCT T=MULTMSG,S=0,M='GDG',CA=ESTOP
EQQJCCT T=MULTEND
```

In Example 3, the SYSOUT line is scanned for IEC501. If IEC501 is found, the SYSOUT line is also scanned for PRIVAT. If both IEC501 and PRIVAT are found, the SYSOUT line is scanned for GDG.

T-Example 4

```
EQQJCCT S=49,T=SKIPSTA,M='J E S 2  J O B  L O G'
EQQJCCT S=1,T=SKIPEND,M='ICH0001I'        (RACF LAST ACCESS)
EQQJCCT S=20,T=SKIPEND,M='IEF452I',CA=STOP (JOB NOT RUN-JCL ERR)
```

In Example 4, when the JES2 log starts, checking of SYSOUT records is bypassed until ICH0001I or IEF452I is found in the SYSOUT line.

T-Example 5

```
EQQJCCT S=49,T=SKIP007,M='J E S 2  J O B  L O G'
```

Example 5 shows how you can skip the next 7 records following a record containing `J E S 2  J O B  L O G`.

T-Example 6

```
EQQJCCT T=ENDTAB,CA=CONT
```

In Example 6, a SYSOUT record that does not match any entries in the message table is accepted as normal.

T-Example 7

```
EQQJCCT T=ENDTAB,CA=ESTOP
```

In Example 7, if a SYSOUT record does not match any entries in the message table, an error is generated.

CA=*check action*|**CONT**
Defines what action the JCC should take when a SYSOUT record is found that fulfills the conditions defined by the current table entry. These actions are supported:

CONT
Continue checking. Stop processing the current SYSOUT record (because it matches). Then read the next SYSOUT record and check it, starting again with the first message-table entry.

CHECK
Check the next table entry. In addition, check the current record against the next table entry.

ERROR
Error condition detected. Continue checking, but treat this job as having ended in error.

ESTOP
Error condition detected. Stop checking, and treat this job as having ended in error.

STOP Stop checking. Stop processing the current SYSOUT record; read the remaining SYSOUT records, but do not check their contents.

Here are some examples of the **CA** parameter:

CA-Example 1

```
EQQJCCT M='IEF287I',CA=ERROR,EID=5555
```

Example 1 shows how to flag a job as having ended in error, with an error code of 5555, if message IEF287I is issued by any step in the job.

CA-Example 2

```
EQQJCCT M='B2 TABLE MISSING',CA=ESTOP,EID=1111
```

Example 2 is a specific job message. The JCC signals the error to IBM Workload Scheduler for z/OS and stops further checking of the message output of the job.

CA-Example 3

```
EQQJCCT S=1,T=SKIPSTA,M='IDCAMS SYSTEM SERVICES'
EQQJCCT S=1,T=SKIPEND,M='IDC0002I',CA=CHECK
EQQJCCT S=57,T=NORMMSG,M='CODE WAS 16',CA=ERROR
```

In Example 3, skipping starts when `IDCAMS SYSTEM SERVICES` is found in the SYSOUT line. When `IDC0002I` is found in the SYSOUT record, skipping is stopped, and an error condition is indicated if this text is found in the SYSOUT line: `CODE WAS 16`.

EID=*error code*|____
Defines an error code to be used by IBM Workload Scheduler for z/OS job tracking. The error code is either a 4-digit decimal number or 3 hexadecimal digits preceded by the character X. The default error code is 4 blank characters. If there is no error code, the status of the job is not changed by the JCC. If EID is coded to a no-blank value, you must not perform a NOERROR checking for the same job, even for a different EID return code.

An error code can be defined only when the current check action is ERROR or ESTOP. The JCC normally creates an incident record for all ERROR and ESTOP actions that have been matched. Error code 0000, however, prevents the creation of an incident record. Similarly, the default error code, 4 blanks, causes an incident to be created, but prevents job tracking from treating the match as a real error.

An example of how to use the **EID=** parameter follows:

EID-Example

```
EQQJCCT CA=ERROR,EID=XB37,M='IEC030I'
EQQJCCT CA=ERROR,EID=XD37,M='IEC031I'
EQQJCCT CA=ERROR,EID=XE37,M='IEC032I'
EQQJCCT CA=ERROR,EID=892,M='IEF257'  (SPACE NOT FOUND)
EQQJCCT CA=ERROR,M='DATABASE IS 80% FULL'
```

This example shows how to pair error codes and the messages to be printed. If you specify CA=ERROR or ESTOP but do not specify EID (as in the last line of the example), the error code is 4 blanks. In this case, job tracking is not notified about the error, but a record is written to the incident file. You can use this method to record incidents that do not currently affect the normal processing of jobs but should be investigated later.

TID=*tracking identifier*|________

Defines a tracking identifier code that can be used in the incident log to group similar errors with a common identification. The tracking identifier is a character string with a maximum of 8 characters. The default identifier is 8 blank characters.

TID-Example

```
EQQJCCT CA=ERROR,EID=XB37,TID=SPACE,M='IEC030I'
EQQJCCT CA=ERROR,EID=XD37,TID=SPACE,M='IEC031I'
EQQJCCT CA=ERROR,EID=XE37,TID=SPACE,M='IEC032I'
```

This example shows how you ensure that the error is logged on the incident file and how you match the same comment to different errors.

Sample Message Table

The following macros generate a general message table that you can use to avoid most manual checks of JCL. Exceptions for individual jobs are specified in job-specific message tables, which are not shown here. If you have detailed standards for completion codes, you need only a few job-specific message tables.

Message-table macros

```
( 1) EQQJCCT CA=ESTOP,EID=5555,M='IEF287I',TID=NOTCTLGX
( 2) EQQJCCT CA=ESTOP,EID=4444,M='DATASET LIMIT REACHED',            X
             S=0,TID=REORG-DB
( 3) EQQJCCT CA=ESTOP,EID=0012,M='COND CODE 0012',S=0,TID=RC12
( 4) EQQJCCT CA=ERROR,M='ICE061A',S=21,TID=IOERROR
( 5) EQQJCCT T=MULTSTA,M='IEC501',S=20    MOUNT PRIVATE
( 6) EQQJCCT T=MULTMSG,M='PRIVAT',S=34,CA=ESTOP,EID=6666,TID=PRIVATE
( 7) EQQJCCT T=MULTEND
( 8) EQQJCCT T=MULTSTA,M='COND CODE 0016',S=0
( 9) EQQJCCT T=MULTMSG,M='IBTS',S=0,CA=ERROR,EID=0000,TID=OK-IBTS
(10) EQQJCCT T=MULTMSG,M='GISFR',S=0,CA=ERROR,EID=0000,TID=OK-GIS
(11) EQQJCCT T=MULTEND,CA=ESTOP,EID=0016,TID=RC16
(12) EQQJCCT S=0,T=NORMMSG,M='SPOOL DATASET IS FULL',EID=0016,        X
             TID=IBTSFULL,CA=ESTOP
        END
```

Statement (1)

Detects the IEF287I message, which indicates a `not cataloged 2 job` situation. IBM Workload Scheduler for z/OS job tracking regards this as ended-in-error, with error code 5555.

Statement (2)

Warns that the IMS™ database is full. The job is set to ended-in-error because successors to this job can never terminate successfully.

Statement (3)
: Treats each `COND CODE 0012` as an ended-in-error condition. The error indication detected by job tracking is reset if the remaining steps after the error are not flushed. This definition can save effort in updating old JCL to a common standard. All jobs that have a step resulting in `COND CODE 0012` generate an incident on the incident log.

Statement (4)
: Logs all sort/merge I/O errors to the incident file. IBM Workload Scheduler for z/OS detects if the I/O error is recovered, so no EID is specified.

Statements (5-7)
: Define all MOUNTs or PRIVATs as errors.

Statements (8-12)
: Together, these statements define that `COND CODE 0016` is usually wrong. However, if message IBTS or message GISFR is found on the line (it is always part of the step name at this installation), it is an error only if `SPOOL data set IS FULL` is also found.

Statements (9-10)
: Define that this job is correct, even if IBM Workload Scheduler for z/OS job tracking has detected an error from information collected from the system. If no match is found, statement (11) is processed. This will give EID = 0016 to IBM Workload Scheduler for z/OS because `COND CODE 0016` was found, and it was neither `IBTS` nor `GISFR`.

Statement (12)
: Defines an exception to the exception. Although `COND CODE 0016` is usually wrong, it is correct for IBTS jobs. However, a `COND CODE 0016` IBTS job is incorrect if `SPOOL data set IS FULL` is found later in the scanning.

Chapter 8. Using the data store

The role of the IBM Workload Scheduler for z/OS data store is to locally store a copy of the SYSOUT data that is produced for submitted jobs. This data is transmitted back to the IBM Workload Scheduler for z/OS controller only when requested, that is, only when it is needed for restart and cleanup actions or when explicitly requested for browsing.

In a z/OS environment the data store must be already installed before you can perform either job log retrieval or restart and cleanup.

The data store automatically cleans itself up at a user-specified frequency according to user-specified criteria, in order not to grow excessively.

When the same operation requires multiple restarts, in order to store only the sysouts needed by restart and cleanup to optimize data access, a component of the data store, called Database, is activated within the controller. As a part of the controller, this component is called local data store. Inside the local data store the internal cleanup operations are synchronized with the Current Plan extension.

Overview

The data store runs in a separate address space, and is dedicated to the storing and possible retrieval of SYSOUT data sets belonging to submitted jobs. Key characteristics of the new data store support are listed below:

- A data store should be installed for each JES spool in a system. In a simple JES configuration this would mean a data store for each tracker. In systems with shared spools (for example, JES2 MAS), there will be a data store for each spool, and there will be fewer data stores than trackers.
- It is necessary for the data store to have a specific output destination. This destination must be used only by the data store, which will select the sysout, according to this kind of filter. Note that the reserved destination is unique inside a controller or data store configuration data store. The output destination is used to duplicate the sysouts to be stored in the data store database.
- After the storing has been completed, the duplicated sysouts will be deleted.
- Communication between the controller and the data store is analogous to the controller/tracker communication, although the shared DASD method that is possible for controller/tracker communication is not possible for controller/data store communication. The data store type can be defined either as SNA or XCF, but the same controller can connect to both XCF and SNA data stores. Separate LU and XCF values for controller/tracker and controller/data store connections must be used. The controller is identified by two separate LU values: one for the data stores and one for the trackers. All data stores work on a reserved destination, which must always have the same name.

The following controller subtasks handle the communication with the data store:

FL task
: Sysout Fetch task (including also the XCF communication)

FN task
: FL SNA communication task (started only if SNA communication is used)

Also, in the component of the local data store there are the same subtasks contained in the Main data store Database: the primary index, the secondary index, the data files and error handler subtasks.

The following figure shows an example of data store configuration:

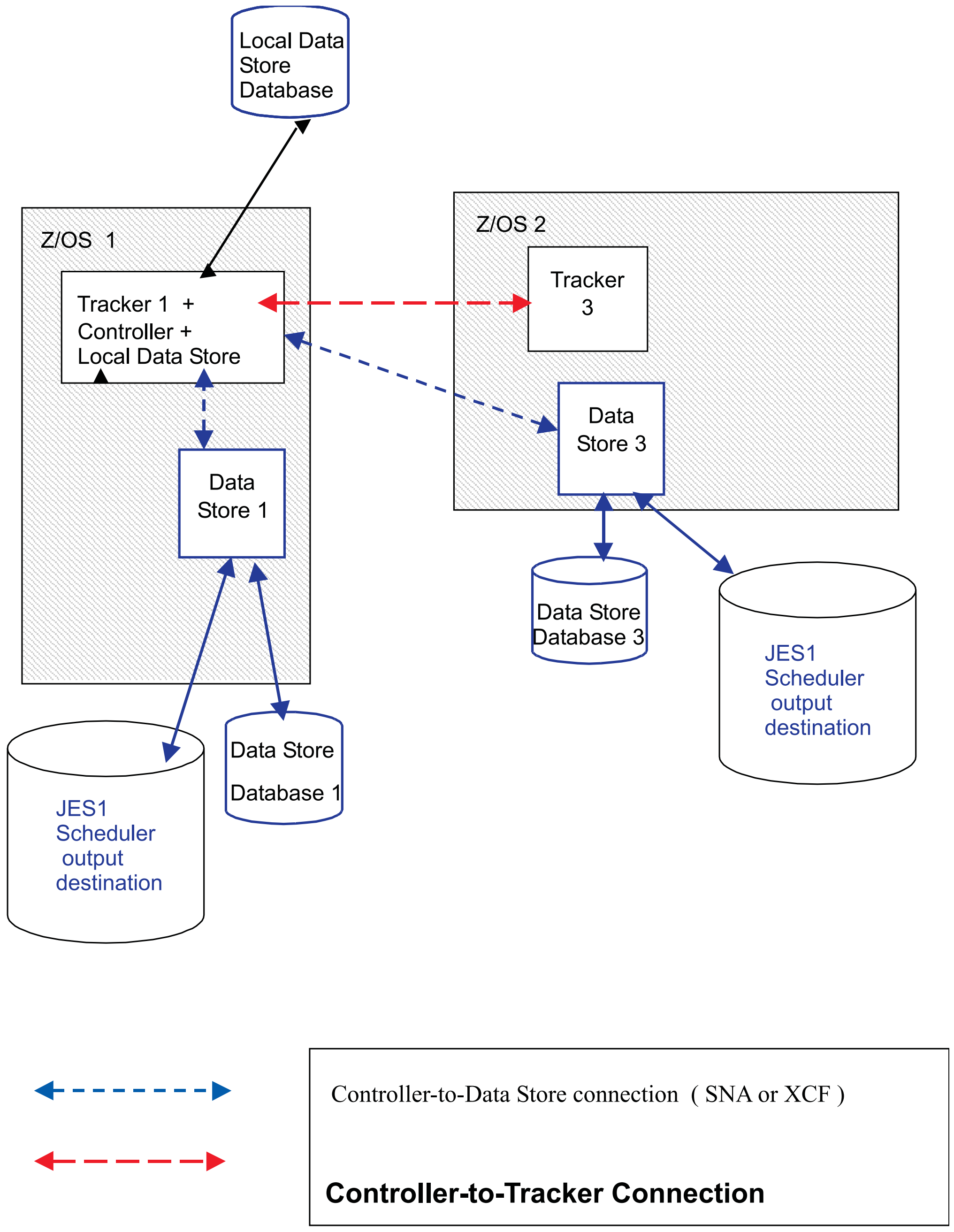

Local Data Store Database
Z/OS 1
Tracker 1 + Controller + Local Data Store
Data Store 1
JES1 Scheduler output destination
Data Store Database 1
Z/OS 2
Tracker 3
Data Store 3
Data Store Database 3
JES1 Scheduler output destination
Controller-to-Data Store connection (SNA or XCF)
Controller-to-Tracker Connection

Figure 4. Controller, tracker, and data store - schematic view

Prerequisites

The data store function can be used only if the following prerequisites are met:

- An output destination is dedicated to the data store

- The OUTPUTNODE (FINAL) keyword is specified on the JTOPTS initialization parameter.

Installing the data store

About this task

A data store should be installed for each JES spool involved in the controller/tracker configuration. To install the data store you must:

- Create and initialize the data store Database. This activity comprises the following steps:
  1. Run the EQQJOBS Clist to create the data store samples
  2. Calculate the data store VSAM file sizes
  3. Allocate the data store VSAM files
  4. Initialize the VSAM files
- Configure the data store. This involves:
  1. Specifying the data store initialization parameter values
  2. Specifying the parameter values for communication with the controller
- Activating the data store
  – Create the startup job for the data store address space

Running EQQJOBS to create installation samples

Run the new EQQJOBS clist. A *Create Data Store Samples* option creates the following set of samples now:

EQQPCS04
To define data store VSAM files and to initialize them

EQQPCS07
To allocate restart and cleanup VSAM data sets

EQQDSCL
To run batch cleanup utility (the input parameters are taken from EQQDSCLP)

EQQASEX
To run batch export utility (the input parameters are taken from EQQDSEXP)

EQQDSIM
To run batch import utility (the input parameters are taken from EQQDSIMP)

EQQDSRI
To run batch recover index utility (the input parameters are taken from EQQDSRIP)

EQQDSRG
To run batch reorg utility (the input parameters are taken from EQQDSEXP and EQQDSIMP)

EQQCLEAN
Sample procedure involving the EQQCLEAN program

EQQDST
Sample data store startup procedure (the input parameters are taken from EQQDSTP)

Estimating the size of data store VSAM data files

The data store SYSOUT database consists of VSAM

- Data files for structured and unstructured data
- Primary index
- Secondary index

Data Files

The data store distinguishes VSAM data-file (DD) types by their names: structured DDs are called EQQSDFnn; unstructured DDs are called EQQUDFnn.

Although the datafile structure for these two types is the same, their content and purpose differ, as described below.

Unstructured data files

The Unstructured data files contain the SYSOUTs in a flat form, as provided by the JES spool. You can check the SYSOUT with the BROWSE JOBLOG function. Note that the unstructured data file can store, if requested, also the user SYSOUTs. The activation of the unstructured data files is optional, depending on appropriate data store parameters.

Within an unstructured data file, every SYSOUT, consisting of n logical records, takes at least one page of data (4096 bytes). The size of the VSAM data file depends on the following factors:

- The typical size of the SYSOUT for jobs that have to be stored (consider also the MAXSTOL parameter that specifies the number of user SYSOUT lines to be stored)
- The average number of jobs that run every day
- The retention period of job logs in data store
- The number of data files that you want to create (from 1 to 99)

You can calculate the number of pages that you need in this way:

- Calculate the maximum number of job logs that can be stored at a given time. To do this, multiply the number of jobs running in a day by the number of days that you want the job logs to be available.
- Calculate the average number of pages that are needed for every job log. This depends on the average number of lines in every SYSOUT and on the average SYSOUT-line length. At least one page is needed for every job log.
- Calculate the total number of pages required. To do this, multiply the number of job logs stored concurrently by the average number of pages for every SYSOUT.
- Calculate the number of pages required for each file. To do this divide the previous result by the number of Data Files you want to create.
- Determine size of each data file according to the media type and space unit for your installation.

Example of calculating for unstructured data files:
A company runs 1000 jobs every day on a single system, and each job generates around 4000 lines of SYSOUT data. Most lines are 80 characters long. Restart and cleanup actions are taken almost immediately if a job fails, and so it is not necessary to keep records in the data store for more than 1 day.

A decision is made to spread the data over 10 files. The maximum number of logs stored at a given time 1 is: 1000 * 1 = 1000. As each log is about 4000 lines long, and each line is about 80 characters long, the number of bytes of space required for each is: 4000 * 80 = 320,000 Thus, the total number of bytes of space required is: 320,000 * 1000 = 320,000,000

If 4 files were used, each file would hold the following number of bytes of data: 320,000,000 / 4 = 80,000,000.

If 3390 DASD was used, each file would require this number of tracks: 80,000,000 / 56664 = 1412 or this number of cylinders: 80,000,000 / 849960 = 94

Structured data files

The structured data files contain job log SYSOUTs in a form based on the parsing of the three components of the job log, the JESJCL, the JESYSMSG and the JESMSGLG, especially the first two. User SYSOUTS are excluded from the structuring mode. Each job log stored consists of two distinct parts:

- A number of pages, each consisting of 4096 bytes dedicated to the expanded JCL
- A number of pages dedicated to a complete, hierarchically ordered set of structured elements for the restart and cleanup functions.

Therefore, the minimum page number used by a structured SYSOUT is 2, and the medium space usage depends on the job complexity.

To determine the optimal dimension for the structured data files, follow the instructions provided for the allocation of the unstructured data file, but take into account that the user SYSOUTs are not present. For the medium structured SYSOUTs, apply the criteria used for the unstructured job log: the larger memory requirement of the small, structured SYSOUTs, compared to the corresponding unstructured form, is balanced by the larger memory requirement of the unstructured form when the SYSOUT complexity increases.

Primary index

Every row in the primary index file has a fixed 77-character length. Each row can represent either one user SYSOUT data set or the three z/OS SYSOUT data sets together (JESMSGLG, JESJCL, and JESSYSMSG). Each row contains a key showing the job name, job ID, start reader date, and start reader time, that points to the data stored in the data files. To set the right size of VSAM Primary Index file, multiply the average number of SYSOUT data sets per job by the maximum number of jobs stored concurrently in the database. This value is the maximum number of rows in the primary index; it should be increased by an adequate margin to cope with peaks in your workload and to allow for growth.

To find the total space quantity to allocate for VSAM primary index, you should multiply this adjusted maximum row number by the total length of the record.

Example

The vast majority of the 1000 jobs run daily by the same company of the previous example generates a single user SYSOUT data set, along with the usual system data sets. Thus, the maximum number of rows in the index is: 2 * 1000 = 2000. Allowing 50% for growth, the space required for the index is: 3000 * 77 = 231000 bytes. On a 3390 this is 231000 / 56664 = 4 tracks.

Secondary index

The secondary index is a variable-length key-sequenced data set (KSDS). Because it can be a single record, that corresponds to a specific secondary-key value, it can trace many primary keys. Currently, a secondary key value is associated to a single primary key only, and, for this reason, each SYSOUT in the secondary index requires one row of 76 characters.

To set the size of the VSAM secondary index file, perform the following steps:

1. Multiply the average number of SYSOUT data sets for each job by the maximum number of jobs stored currently in the database. The result is the maximum number of rows in the Secondary index.
2. Increase this value to cope with peaks in workload and to allow for growth.
3. Multiply this adjusted value by the total length of the record. This gives the total space for allocating for the VSAM secondary index.

Characteristics of the local data store

The criteria for setting the size of the VSAM local data store differ from those for the main data store. Therefore, note the following:

- Only those SYSOUTs in the main data store that are subject to restart and cleanup are also stored in the local data store.
- Because unstructured data is not subject to restart and cleanup, the local data store requires significantly less space.

Allocating data store VSAM

This section describes what you must do to allocate the VSAM files required by the data store. The sample member EQQPCS04 contains the JCL to allocate these files.

Data files

You can create up to 99 structured data files and to 99 unstructured data files. Each must be identified by a unique ddname in the data store start job.

This is an example of the creation of a structured and unstructured data file:

```
//STRUCT    EXEC PGM=IDCAMS
//SYSPRINT  DD  SYSOUT=*
//EQQSDF01  DD  UNIT=SYSDA,VOL=SER=S25PRA,DISP=OLD
//SYSIN     DD  *
    DELETE OPCDEV1.SDF01
    DEFINE CLUSTER (NAME(OPCDEV.SDF01) -
             VOLUMES(S25PRA) -
             TRACKS(1,1) -
             SHAREOPTIONS(2,3) -
             LINEAR)
//UNSTRUCT    EXEC PGM=IDCAMS
//SYSPRINT  DD  SYSOUT=*
//EQQUDF01  DD  UNIT=SYSDA,VOL=SER=S25PRA,DISP=OLD
//SYSIN     DD  *
    DELETE OPCDEV1.UDF01
    DEFINE CLUSTER (NAME(OPCDEV.UDF01) -
             VOLUMES(S25PRA) -
             TRACKS(1,1) -
             SHAREOPTIONS(2,3) -
             LINEAR)
```

Primary index

You must define one primary index for each data store and initialize it with a header record.

This is an example of JCL for the scratch and creation of the primary index file:

```
//DELPKI   EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
           DELETE OPCDEV.PKI0X CLUSTER PURGE
//*----------------------------------------------------------------
//DEFPKI   EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD *
           DEFINE CLUSTER(NAME(OPCDEV.PKI0X)-
           CYLINDERS(2,1)-
           VOLUMES(S25PRA)-
           KEYS(34,0)-
           RECORDSIZE(77,77)-
           CISZ(4096)-
           UNIQUE-
           INDEXED-
           SHR(1,3)-
           FREESPACE(10,10))
```

Secondary index

You must define one Secondary index for each data store and initialize it with a header record.

This is an example of JCL for the scratch and creation of the secondary index file:

```
//DELSKI   EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
           DELETE OPCDEV.SKI0X CLUSTER PURGE
//*----------------------------------------------------------------
//DEFSKI   EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD *
           DEFINE CLUSTER(NAME(OPCDEV.SKI0X)-
           CYLINDERS(2,1)-
           VOLUMES(S25PRA)-
           KEYS(40,0)-
           RECORDSIZE(76,32000)-
           CISZ(4096)-
           UNIQUE-
           INDEXED-
           SHR(1,3)-
           FREESPACE(10,10))
```

Initializing data store VSAM files

This section describes the steps necessary to initialize VSAM files used by the data store. Sample member EQQPCS04 contains JCL to do this task.

Data files

You do not need to initialize Data files, they are automatically formatted the first time that the data store is started.

Primary index

Every primary index file must be initialized with a header record:

```
POS 1 - 8    blank
POS 9 - 16   '00010101'
POS 17 - 77  'b   (binary zeros)
```

This is a sample of JCL that initializes the header record of a primary index file. It can be run separately or added to the job that creates the VSAM files.

```
//INIPKI   EXEC PGM=SORT
//SYSOUT   DD SYSOUT=*
//SORTIN   DD *
        00010101
/*
//SORTOUT  DD DSN=OPCDEV2.RES.PKI0X,DISP=SHR
//DFSPARM  DD *
  RECORD TYPE=V
  SORT FIELDS=(1,16,CH,A)
  OUTREC FIELDS=(9:C'00010101',61Z)
```

Secondary index

Every secondary index file must be initialized with a header record having the minimum record length, 76 characters, all set to binary zeroes.

This is a sample of JCL that initializes the header record of a secondary index file. It can be run separately or added to the job that creates the VSAM files.

```
//*-----------------------------------------------------------------*
//* PREPARE HEADER RECORD
//*-----------------------------------------------------------------*
//INIT01   EXEC PGM=SORT
//SYSOUT   DD SYSOUT=*
//SORTIN   DD *
 0
/*
//SORTOUT  DD DSN=OPCDEV2.RES.SKIHDR,DISP=(NEW,CATLG),UNIT=SYSDA,
              DCB=(RECFM=F,LRECL=76,BLKSIZE=76),SPACE=(TRK,(1))
//DFSPARM  DD *
  RECORD TYPE=F
  SORT FIELDS=(1,1,CH,A)
  OUTREC FIELDS=(76X'00')
//*-----------------------------------------------------------------*
/* INITIALIZE SECONDARY INDEX
//*-----------------------------------------------------------------*
//INIT02   EXEC PGM=IDCAMS
//SYSPRINT   DD SYSOUT=*
//SYSIN      DD *
             REPRO INDATASET(OPCDEV2.RES.SKIHDR)-
             OUTDATASET(OPCDEV2.RES.SKI0X)
//*-----------------------------------------------------------------*
//* DELETE INPUT FILE
//*-----------------------------------------------------------------*
//INIT03   EXEC PGM=IEFBR14
//SORTOUT    DD DSN=OPCDEV2.RES.SKIHDR,DISP=(OLD,DELETE,DELETE)
```

Post-installation actions on data store VSAM files

About this task

While primary and secondary indexes need to be initialized, data files are initialized automatically at data store startup. To add a new data file, create it and add it to the data store startup procedure. However, after a data file has been initialized, it cannot be removed from the procedure. To reallocate a data file, the primary and secondary indexes must be reallocated too.

When VSAM local and remote data files fill up:

1. Make a copy of all the data files and primary and secondary indexes to temporary files by using the REPRO function of IDCAMS.
2. DELETE and DEFINE clusters, to increase the allocated space.

3. Use the REPRO function to copy the temporary files into the new allocated VSAM files.
4. If needed, define new data files and add them to the data store procedure. The data files will be initialized the first time data store starts up.

Instead of IDCAMS, you can also use the EXPORT and IMPORT utilities. This method is slower but reorganizes the data files as well. Reorganizing the data files means that all the lost space is recovered, although this has no impact on data store performances.

To reduce the number of data files, use the EXPORT and IMPORT utilities. After the EXPORT phase, you can DELETE and DEFINE clusters and reduce their number.

When a primary or secondary index is corrupted, use the RECOVER utility that, starting from data files, reconstructs them. The recovery of both the primary and secondary indexes starts from the same utility, which reads data files to recover the primary index first.

Configuring the data store

To see an example of the DSTOPTS and FLOPTS statements that you need to set to configure the data store read the following information. For detailed examples about the data store configuration refer to *IBM Workload Scheduler for z/OS Planning and Installation*.

Data store initialization statements

The data store initialization statements are coded in a member of the partitioned data set specified by the EQQPARM DD statement in the start JCL. The member is identified by the PARM parameter on the EXEC statement. EQQDSTP is provided as a sample data store initialization member.

The following is an example of a simple data store configuration:

```
DSTOPTS
           HOSTCON(SNA)
         MAXSTOL(0)
         NWRITER(3)
         SYSDEST(OPC2)
         STOUNSD(Y)
         QTIMEOUT(15)
         WINTERVAL(15)
         DSTLUNAM(I9PC45A3)
         CTLLUNAM(I9PC45R3)
         DELAYTIME(15)
         CINTERVAL(60)
         CLNPARM(EQQCLNPA)
            HDRJOBNAME(JOBNAME)
         HDRSTEPNAME(STEPNAME)
         HDRPROCNAME(PROCSTEP)
         HDRJOBLENGTH(21)
         HDRSTEPLENGTH(30)
         HDRPROCLENGTH(39)
```

Set UP controller/tracker initialization statements

The FL task (job log requester) initialization statements are defined in the same member as the current controller initialization statements by means of the statement FLOPTS. The EQQCONP and EQQCONOP sample members contain examples of these statements.

The following is an example of a simple FLOPTS statement:

```
FLOPTS CTLLUNAM(I9PC45R3)
         SNADEST(I9PC45T3.I9PC45A3)
```

Considerations about RCLOPTS statements

The options used by the controller during the restart and cleanup functions are defined with the RCLOPTS statements (the EQQCONP and EQQCONOP sample members contain examples of this statement). Because in some cases EQQCLEAN might delete a data set by mistake, it is recommended that you protect critical data sets from deletion by using either the RCLOPTS parameters (DDPROT, DDPRMEM, DSNPROT, DSNPRMEM) or the EQQUXCAT exit.

The following is an example of a simple RCLOPTS statement:

```
RCLOPTS DSTDEST(OPC)
        DSNPRMEM(MYPROT)
        STEPRESCHK(NO)
```

Activating the data store

The data store start JCL can be created by copying the supplied sample, EQQARCH, and customizing it to meet your installations needs with regard to naming convention and the number of files to be used.

The following shows the JCL statements that are typical of the data store, and which do not apply to other IBM Workload Scheduler for z/OS components (like the controller or the tracker):

```
//OCDST1 JOB CLASS=Y
//OCDST1 EXEC PGM=EQQFARCH,REGION=0M,PARM='EQQDSTP',TIME=1440
.
.
.
//EQQSDF01  DD  DISP=SHR,DSN=OPCDEV.SDF01
//EQQUDF01  DD  DISP=SHR,DSN=OPCDEV.UDF01
//EQQUDF02  DD  DISP=SHR,DSN=OPCDEV.UDF02
//EQQPKI01  DD  DISP=SHR,DSN=OPCDEV.PKI01
//EQQSKI01  DD  DISP=SHR,DSN=OPCDEV.SKI01
```

Chapter 9. Miscellaneous customization

This chapter contains these topics:

- "Customizing IBM Workload Scheduler for z/OS messages" describes how to change the routing of IBM Workload Scheduler for z/OS messages.
- "Customizing IBM Workload Scheduler for z/OS panels" on page 319 describes how to update IBM Workload Scheduler for z/OS panels for installation-specific requirements.
- "Customizing ended-in-error-list and ready-list default layouts" on page 319 describes how to create and display your own default layouts.
- "Invoking hiperbatch support" on page 320 describes how IBM Workload Scheduler for z/OS-controlled batch jobs and started tasks use Hiperbatch.
- "Customizing GMT clock" on page 321 describes how IBM Workload Scheduler for z/OS updates the GMT clock.
- "Monitoring special resources through RODM" on page 321 describes how you can use the Resource Object Data Manager to monitor real resources used by IBM Workload Scheduler for z/OS operations.
- "Creating case-code-definition modules" on page 323 describes how to create the modules that are used by automatic job recovery.
- "Invoking the data set deletion utility" on page 324 describes the program that you can use to delete data sets based on the disposition specified in the JCL and the current status of the data set in the catalog.
- "Customizing IBM Workload Scheduler for messages in end-to-end with fault tolerance capabilities environment" on page 324 describes how to enable messages for end-to-end scheduling with fault tolerance capabilities (AWS and EQQPT), so they can be issued to the server MLOG and to the System Output Console.

This chapter contains diagnosis, modification, and tuning information.

Customizing IBM Workload Scheduler for z/OS messages

Read the following information to understand how to customize IBM Workload Scheduler for z/OS messages.

All IBM Workload Scheduler for z/OS messages are prefixed by EQQ. They are normally written to the IBM Workload Scheduler for z/OS message log or ISPF dialog user. Messages can, however, be routed to other destinations as well. This is accomplished with the write-to-operator (WTO) routing codes.

For example, two network communication function (NCF) messages (EQQV028 and EQQV033) are defined to route as master console information messages (routing code 2). You can change this routing in the message members that hold the NCF messages. The member name for a particular message is simply the message number minus the last digit. For example, EQQV028 is found in member EQQV02. Each member, therefore, holds up to 10 messages.

Like ISPF messages, messages defined in the IBM Workload Scheduler for z/OS message library consist of two or more records. The first record holds the message number and keywords. The second record holds the message text.

A message definition with the following first line indicates that the corresponding message will be routed to the destination represented by routing code 2, in addition to being written to the message log:

Unmodified message record

```
EQQV028  ' '  WTO=YES  ROUTE=2
```

To choose another destination, for example, routing code 8, modify this record to:

Modified message record

```
EQQV028  ' '  WTO=YES  ROUTE=8
```

If you want to eliminate all destinations except the message log, remove the WTO and ROUTE keywords altogether. You can also add the WTO and ROUTE keywords to messages that normally appear only on the message log.

Note: When a message is issued as a WTO, the message text cannot exceed 70 characters. For this reason, it may be necessary to reorganize the text in messages that are modified to include WTO=YES. If the text reorganization adds lines to accommodate the entire message, be sure to modify or add the LINES keyword to the message definition.

IBM Workload Scheduler for z/OS messages are stored in members of the message library data set that is created during installation. This data set contains messages for both the IBM Workload Scheduler for z/OS message log and the dialog user. The WTO and ROUTE keywords are not valid for dialog messages, which are read and displayed by ISPF dialog functions. It is not easy to distinguish between dialog messages and message-log messages because they use the same format. If you want to add the WTO or ROUTE keywords to a message, the safest way is to modify only messages that you have seen on the IBM Workload Scheduler for z/OS message log. Also, be aware that the messages written by the Daily Planning batch job in the data set pointed to by the EQQDIN ddname cannot be routed to the system log when you set the keyword WTO=YES.

The message library is a normal partitioned data set with 80-byte fixed-length records. If you want to modify one or more messages as described previously, you should create an additional message library and copy the relevant members to it from the message library. Then you can modify your own library, and leave the message library in the same state as when it was installed or updated by maintenance. You should then include your message library in the JCL for the started task, concatenated in front of the message library on the EQQMLIB DD statement.

If you create your own message library in this way, you must review any changes that occur in the message library as a result of maintenance activity.

Refer to *z/OS Routing and Descriptor Codes* for more information about routing codes.

Note: The text `MESSAGE IS NOT DEFINED` might be replacing the text of any IBM Workload Scheduler for z/OS issued message id. Moreover, the message type is set to `E` although the message manual states it is an `I` or `W` message. This occurs if IBM Workload Scheduler for z/OS is unable to locate the normal text for the message id in the `EQQMLIB` library (for example, `SEQQMSG0`). This might be caused by a wrong message customization, or by a wrong user setup in the message libraries concatenation.

Customizing IBM Workload Scheduler for z/OS panels

If required, you can customize the IBM Workload Scheduler for z/OS panels to add installation-specific information or to extend the online help with examples specific to your business application systems.

The panel library is a normal partitioned data set with 80-byte fixed-length records. If you want to modify one or more panels you should create an additional panel library and copy the relevant members to it from the IBM Workload Scheduler for z/OS panel library. Then you can modify your own library, and leave the IBM Workload Scheduler for z/OS panel library in the same state as when it was installed or updated by maintenance. You should then include your panel library in your ISPF concatenation in front of the IBM Workload Scheduler for z/OS panel library on the ISPPLIB DD statement.

If you create your own panel library in this way, you must review any changes that occur in the IBM Workload Scheduler for z/OS panel library as a result of maintenance activity.

Customizing ended-in-error-list and ready-list default layouts

About this task

IBM Workload Scheduler for z/OS takes the layout of ended-in-error lists and ready lists from two sources in the following sequence:

- The ISPF profile data set (ISPPROF), member names EQQELOUT and EQQRLOUT. These members contain user-defined layouts that each user creates and maintains individually, using the IBM Workload Scheduler for z/OS dialog.
- The ISPF table input data set (ISPTLIB), member names EQQELDEF and EQQRLDEF. These members contain installation-defined default layouts. Sample EQQELDEF and EQQRLDEF tables are shipped with the product in the ISPF tables (SEQQTBL0) library. SEQQTBL0 is allocated to the ISPTLIB DD statement during the installation procedure.

If you want to modify the default layouts contained in EQQELDEF or EQQRLDEF, perform the following steps:

1. Use the IBM Workload Scheduler for z/OS dialog to set up the layouts as you want them to appear. This will cause a modified EQQELOUT or EQQRLOUT table to be written to your ISPF profile data set.

   Note: Also edit all default layouts that you want to include in the new default table. You need not update the layouts, but each layout you edit is written to your ISPF profile data set.
2. Copy the modified table from your ISPF profile library to the IBM Workload Scheduler for z/OS table library allocated to ISPTLIB, and rename it to the default table name (EQQELDEF or EQQRLDEF). The layouts you created or edited will now be the default layouts for all users.

Note: The tables EQQLUOUT (*xxxxLUOUT* with **PQ92255** APAR installed) and EQQLUDEF can be customized in the same way. For details, see “Setting Up the ISPF Tables” in the *IBM Workload Scheduler for z/OS: Planning and Installation*.

Invoking hiperbatch support

About this task

Hiperbatch is a z/OS performance enhancement that works with DLF (data lookaside facility) to allow batch jobs and started tasks to share access to a data set, or *data object*. IBM Workload Scheduler for z/OS provides control information to DLF concerning which operations are allowed to connect to which DLF object and which data sets are eligible for Hiperbatch.

Within IBM Workload Scheduler for z/OS, a data set eligible for Hiperbatch is treated as a special resource. Using the Special Resource Description dialog, you define a special resource with the name of the data set and specify Y (yes) in the Hiperbatch field. The DLF exit sample, EQQDLFX, can then make the following decisions about the DLF component:
- Will this data set be eligible for Hiperbatch?
- Should this operation be connected to this data object?

IBM Workload Scheduler for z/OS issues enqueues on the job and data set name to notify the DLF exit that the job to be scheduled will use Hiperbatch. When the job ends, IBM Workload Scheduler for z/OS checks if the same data set is required by the immediate successor operation or other ready operations. If the data set is not required, IBM Workload Scheduler for z/OS initiates purge processing (that is, IBM Workload Scheduler for z/OS removes the data object from Hiperspace™) also for operations that have ended in error, unless the *keep on error* value specifies that the resources allocated to the operations must be kept.

Only the system where the controller is started and systems participating in the same global resource serialization (GRS) ring interact with DLF. Before you can use IBM Workload Scheduler for z/OS Hiperbatch support, you must:
1. Install the DLF connect/disconnect exit. SEQQSAMP member EQQDLFX contains an assembler program that provides control information to DLF based on information provided by IBM Workload Scheduler for z/OS. Refer to *Planning and Installation*
2. Add started-task procedure EQQPROC. When an object in Hiperspace is no longer needed by the jobs, IBM Workload Scheduler for z/OS initiates a PURGE of this object. A start command is issued from within IBM Workload Scheduler for z/OS:

   EQQPROC

   S EQQPROC, PARM='*resource name*

   (Sample installation JCL for this started task is contained in sample member EQQPROC.)
3. Create a file containing purge JCL. EQQPROC initiates an IBM Workload Scheduler for z/OS batch program, EQQPURGE. EQQPURGE requires input JCL to submit to the JES internal reader. Sample member EQQJCLIN contains sample input JCL. When the DLF exit is installed on a z/OS system other than the IBM Workload Scheduler for z/OS controller, the JCL must contain routing information to transmit the job to the correct z/OS system.

Customizing GMT clock

About this task

If you change the local time on a tracker, IBM Workload Scheduler for z/OS updates the GMT clock automatically as follows:

Procedure

1. It creates a new clock record and sends an IBM Workload Scheduler event to the affected tracker.
2. The event triggers a refresh of the GMT clock.
3. The event is also forwarded to the controller for synchronization.

Results

Note: The auto-detect function works regardless of whether the local time is changed using the **set date** or **set clock** command or using the sysplex timer. The SMF type 90 record that is required to auto-detect the local time change is created regardless of how the local time is changed.

Monitoring special resources through RODM

You can use the Resource Object Data Manager to track the status of real resources used by IBM Workload Scheduler for z/OS operations. RODM is a data cache that contains information about real resources at your installation. Products such as AOC/MVS report actual resource status to RODM; RODM reflects the status by updating values of fields in classes or objects that represent the real resources. Subsystems on the same z/OS image as RODM can subscribe to RODM fields. When RODM updates a field, all subscribers to the field are notified.

IBM Workload Scheduler for z/OS support for RODM lets you subscribe to RODM fields for fields in special resources. When RODM notifies a change, IBM Workload Scheduler for z/OS updates resource fields that have a subscription to RODM. You can subscribe to RODM for these fields:

AVAILABLE
The Available field in the resource. This value overrides the default and interval values.

QUANTITY
The Quantity field in the resource. This value overrides the default and interval values.

DEVIATION
The Deviation field. You use this field to make a temporary adjustment to quantity. IBM Workload Scheduler for z/OS adds quantity and deviation together to decide the amount that operations can allocate. For example, if quantity is 10 and deviation is -3, operations can allocate up to 7 of the resource.

You specify these keywords to invoke monitoring through RODM:

RODMTASK
Is specified on the OPCOPTS statement for the controller and for each tracker that communicates with a RODM subsystem.

RODMPARM
Is specified on the OPCOPTS statement for the controller and identifies the member of the parameter library that contains RODMOPTS statements.

RODMOPTS
Is specified for a controller and contains destination and subscription information.

A RODMOPTS statement is required for each field in every resource that you want to monitor. Each statement is used to subscribe to a field in a RODM class or RODM object for a field in a special resource. The RODM field value is used to set the value of the resource field.

RODMOPTS statements are read when the controller is started. When a tracker that communicates with RODM is started, it requests parameters from the controller. The controller sends subscription information to the tracker, which then subscribes to RODM. An event is created when RODM returns a value, which is used to update the special resource field in the current plan. IBM Workload Scheduler for z/OS does not schedule operations that use a special resource until RODM has returned the current field value and IBM Workload Scheduler for z/OS has updated the resource.

To use RODM monitoring you must ensure that:
- A tracker is started on the same z/OS image as the RODM subsystem that requests are sent to, and RODMTASK(YES) is specified for both the tracker and the controller.
- An event writer is started in the IBM Workload Scheduler for z/OS address space that communicates with RODM. This address space creates resource events (type S) from RODM notifications, which IBM Workload Scheduler for z/OS uses to update the current plan.
- The controller is connected to the tracker through XCF, NCF, or a submit/release data set.
- Each address space has a unique RACF user ID if more than 1 IBM Workload Scheduler for z/OS address space communicates with a RODM subsystem, such as when you start production and test systems that subscribe to the same RODM subsystem.

IBM Workload Scheduler for z/OS does not load or maintain data models in the RODM cache, or require a specific data model. You need not write programs or methods to use RODM through IBM Workload Scheduler for z/OS, or define specific objects or fields in RODM. IBM Workload Scheduler for z/OS does not update RODM-defined data.

RODM fields have several subfields. The RODM field that IBM Workload Scheduler for z/OS subscribes to must have a *notify* subfield. Through a subscription to this subfield, RODM notifies IBM Workload Scheduler for z/OS of changes to the *value* subfield. IBM Workload Scheduler for z/OS uses changes to the value subfield to monitor special resources. But only these data types are valid for IBM Workload Scheduler for z/OS RODM support:

Table 43. Valid RODM data types for value subfields

Abstract data type	**Data type ID**
CharVar (Char)	4
Integer (Bin 31)	10

Table 43. Valid RODM data types for value subfields (continued)

Abstract data type	Data type ID
Smallint (Bin 15)	21

IBM Workload Scheduler for z/OS maintains a RODM status for all special resources in the current plan. You can check the current status in the Special Resource Monitor dialog. Each special resource has one of these values:

N	Not monitored. The special resource is not monitored through RODM.
I	Inactive. Monitoring is not currently active. IBM Workload Scheduler for z/OS sets this status for all subscriptions to a RODM subsystem that the controller cannot communicate with. This can occur when communication is lost with RODM or with the tracker. The controller sets the value of each monitored field according to the RODMLOST keyword of RODMOPTS.
P	Pending. IBM Workload Scheduler for z/OS has sent a subscription request to RODM, but RODM has not returned a value.
A	Active. IBM Workload Scheduler for z/OS has received a value from RODM and the special resource field has been updated.

Note:

1. The names of RODM classes, objects, and fields are case-sensitive. Ensure you preserve the case when specifying RODMOPTS statements in the parameter library. Also, if a name contains anything other than alphanumeric or national characters, you must enclose the name in double quotation marks.
2. If IBM Workload Scheduler for z/OS subscribes to RODM for a resource that does not exist in the current plan and the DYNAMICADD keyword of RESOPTS has the value YES or EVENT, the event created from the data returned by RODM causes a dynamic add of the resource. DYNAMICADD is described in the list of RESOPTS "Parameters" on page 139.
3. If a request from IBM Workload Scheduler for z/OS cannot be processed immediately because, for example, long-running programs in RODM access the same data that IBM Workload Scheduler for z/OS requests need access to, be aware of possible delays to operation start times.

Creating case-code-definition modules

EQQCASEM is a nonexecutable module that holds case-code definitions. A case-code definition is a list of abend codes and return codes that require the same recovery actions, grouped so that they can be referenced under a single name. These lists are used by the automatic-recovery function.

You create your own case-code definitions by assembling a file consisting of a number of EQQCASEC assembler macro invocations. The EQQCASEC invocations must be the only codes in the assembler file. Give the load module the name EQQCASEM when it is link-edited, and place it in a load module library that is available to IBM Workload Scheduler for z/OS. Use RMODE(24) and AMODE(24) when linking the module.

```
    ┌──────────────────────────────────────────────────────────┐
    │                                          ┌─,──┐          │
    ▼                                          ▼    │          │
►►──EQQCASEC──┬─CASE─=─code name─,─CODES─=─(───code─┴─)─┬──────┴────────────►◄
              └─END─────────────────────────────────────┘
```

CASE Specifies the case code to be defined when this macro is invoked. It can be 1 to 4 characters and should follow JCL naming standards.

CODES
Specifies a list of IBM Workload Scheduler for z/OS job completion codes, return codes, and case codes. The code name in the CASE parameter represents all the codes in the CODES parameter.

Each macro invocation except the last defines a case code. The last record of the module must be `EQQCASEC END`.

The distributed load module defines two case codes, **NOAR** and **SYST**. They are created by the following assembler file:

Example of EQQCASEM module definition

```
EQQCASEC CASE=NOAR,CODES=(S122,S222,CAN,JCLI,JCL,JCCE)
EQQCASEC CASE=SYST,CODES=S222
EQQCASEC END
```

Invoking the data set deletion utility

EQQDELDS is an IBM Workload Scheduler for z/OS-supplied program that deletes data sets based on the disposition specified in the JCL and the current status in the catalog. You can use this program if you want to delete data sets that are cataloged by your applications.

JCL to run EQQDELDS and details about its parameters are provided in member EQQDELDI in the SEQQSAMP library. See “Deleting data sets based on JCL disposition and catalog status” on page 406 for more information.

Customizing IBM Workload Scheduler for messages in end-to-end with fault tolerance capabilities environment

The IBM Workload Scheduler for z/OS EQQPT and the IBM Workload Scheduler AWS message records are not customizable, as it happens for the IBM Workload Scheduler for z/OS messages (EQQ) in the SEQQMSG0 library.

All AWS and EQQPT messages are normally written to the following log files:
- Both AWS and EQQPT are written to the TWSMERGE.log, E2EMERGE.log, and NETMAN.log HFS or ZFS files.
- Both AWS and EQQPT messages can be routed to either or both the Server MLOG and the System Output Console.

You can customize the `TWSCCLog.properties` file located in the fault-tolerant end-to-end work directory in UNIX Systems Services to specify which messages are to be routed to the server MLOG and to the Syslog (for details, see the *Scheduling End-to-end with Fault Tolerance Capabilities* manual).

Messages regarding errors occurred while parsing the `TWSCCLog.properties` file, are issued from the CCLOG tool in the stderr file and in the `<date>` files in the stdlist directory.

Part 2. Data integrity

This part describes how you back up data sets and data store sets and how you plan for disaster recovery.

Chapter 10. Backup and recovery of data sets

IBM Workload Scheduler for z/OS is, in a sense, an online system that creates and manages two resources, the long-term plan (LTP) and the current plan (CP). These two resources and the data sets needed to re-create them are important assets that must be protected from damage. To do this, you should establish data set backup procedures to enable the administrator of IBM Workload Scheduler for z/OS to recover these resources if they are damaged or lost.

The task of submitting and tracking your batch processing is necessarily complex and involves a number of data sets. For this reason, IBM Workload Scheduler for z/OS automatically handles the backup and synchronization of the current plans. This process is explained in detail in *Managing the Workload*

Your recovery procedures can include the use of IBM Workload Scheduler for z/OS *hot standby* facilities. If the z/OS system that your controller resides on fails, or the controller itself fails, the IBM Workload Scheduler for z/OS controller function can be transferred to a standby z/OS system. To use this facility, your systems must be running in a z/OS sysplex using the cross-system coupling facility (XCF).

Backup procedures

These VSAM data sets should be backed up on a daily basis:
- Application description (AD) data set
- Workstation description (WS) data set
- Operator instruction (OI) data set
- Special resource description (RD) data set
- Side information (SI) data set (if high activity)
- LTP data set

You need not back up the JCL repository (JS) data set on a daily basis for recovery purposes. IBM Workload Scheduler for z/OS automatically backs up the JS file based on the value specified for the MAXJSFILE keyword of the JTOPTS statement for the subsystem. However, you can disable this automatic backup and schedule JS file backups as job operations using the BACKUP command; for example, if you want to schedule backups during times when the workload on the system is low. The JS data set consists of two data sets, one active, the other inactive. Recovery is simply a matter of copying the inactive data set to the active data set.

You can back up VSAM data sets using the REPRO function of the IDCAMS utility program. But you cannot use REPRO if the backup is to a sequential file and the record length is greater than 32 760. Instead, use DFSMSdss, or an equivalent product, to perform the backup. If you use DFSMS, consider DFSMShsm ABARS for data backup and restore. ABARS simplifies the backup and recovery process.

Normally, VSAM data sets are unloaded to a sequential data set. A recommended practice is to use a generation data group as the backup data set. If backup data sets are allocated on DASD, they must be on a different volume from the VSAM data set being backed up. On a 3380 direct access storage device, the backup data set should be on a different head disk assembly from the data set being backed up.

Note: Non-VSAM data sets, such as the job library data set, the procedure library data set, and the JCC message library data set, are never updated by IBM Workload Scheduler for z/OS. These data sets are therefore not regarded as IBM Workload Scheduler for z/OS-owned resources. Use your normal backup and recovery procedures for these data sets.

The long-term plan data set principally provides input for the daily plan batch jobs that create a new current plan. Some of the daily plan batch jobs update the long-term plan data set to set the status of occurrences in the long-term plan. The long-term plan and the current plans must synchronize otherwise daily-plan processing will fail.

IBM Workload Scheduler for z/OS creates a backup of the long-term plan when long-term plan and daily-planning batch jobs are run. The backup is written to a VSAM file with the ddname EQQLTBKP.

Before and after the new long-term plan is generated, a backup is made for long-term batch jobs. This ensures that a usable long-term plan is available in the backup file with the ddname EQQLTBKP after a long-term plan batch run.

For data processing batch jobs, a long-term backup is made after the new current plan has been successfully created. This ensures that exists a long-term plan backup that is synchronized with the new current plan produced in the file with ddname EQQNCPDS.

Table 44 summarizes how the data sets are backed up while the controller is running.

Table 44. Backup procedures while the controller is running

Data set	Backup procedure
Current plan (CP)	The controller backs up the CP data set at each turnover or when you issue a `backup` command. However, if you want to save an additional copy, run the IDCAMS REPRO function against the inactive CP data set. The inactive data set is indicated in the FROMDD field of message EQQN057I related to the latest backup performed.
Long-Term plan (LTP)	The controller backs up the LTP data set after each LTP and DP batch run in the data set pointed by EQQLTBKP DD. However if you want to save an additional copy of the LTP data set, after the batch run, issue the REPRO command against the EQQLTBKP data set.
Application description (AD), Operation instruction (OI), Special resource description (RD), Workstation description (WS)	Run the IDCAMS REPRO function to copy the data sets when it is appropriate in your environment. To produce a VSAM backup, run the DP trial with the `Copy VSAM` option set to Yes.
Side information (SI)	Because this data set uses the local shared resources (LSR) buffering, to ensure that you copy also the changes processed by the buffering technique run the IDCAMS REPRO function after the data set has been closed and opened again. This occurs when message EQQN018I about the VSAM LSR buffer pool successfully built for the EQQSIDS data set is issued.

How IBM Workload Scheduler for z/OS manages recovery of the current plan

The scheduler automatically backs up the current plan. Depending on your workload, the current plan might be updated many times per second. For this reason, and because the current plan is a critical resource, the scheduler handles it in a different way from the other databases and data sets. With the introduction of fault-tolerant workstations, a new file, called Symphony, can be produced during the daily-plan batch job runs. Recovering the current-plan from an error situation may imply recovering also the Symphony file. For more information about the current-plan back-up process and the Symphony file refer to *Managing the Workload*.

Current-plan recovery principles

IBM Workload Scheduler for z/OS is designed so that in most error situations, the current plan can be automatically recovered without any action required by you.

To achieve the recovery, the following data sets are used:

EQQCP1DS
: Primary current plan data set

EQQCP2DS
: Alternate current plan data set

EQQSCPDS
: Current plan copy used to produce the Symphony file.

EQQCXDS
: Current plan extension data set

EQQNCPDS
: New current plan data set

EQQNCXDS
: New-current plan extension data set

EQQJTnn
: Current and inactive job-tracking logs

EQQDLnn
: Current and inactive dual job-tracking logs

EQQJTARC
: Job-tracking archive log

EQQCKPT
: Checkpoint data set.

However, the descriptions that follow use these logical terms to describe the CP and its associated data sets:

Current plan
: Used when describing the current plan in general. The current plan consists of the active current-plan data set and the extension (CX) file.

Active current plan
: Refers to the current-plan data set that is currently in use within IBM Workload Scheduler for z/OS. It is either EQQCP1DS or EQQCP2DS. Every time a current plan backup is performed, IBM Workload Scheduler for z/OS switches the active current plan to the other data set. *Managing the Workload* describes the current plan backup process.

Current plan extension
: Refers to the file that contains special resource information. The extension

file is held in a data space, which is backed by DASD file EQQCXDS. When a current plan backup is performed, the data space is refreshed to the DASD file.

Inactive current plan
Refers to the current-plan data set that is not currently in use. It contains a backup copy of the current plan. It is either EQQCP1DS or EQQCP2DS.

New current plan
Refers to a new version of the current plan, which is created by one of the daily planning batch jobs. It refers to the EQQNCPDS data set and EQQNCXDS data set.

Current and inactive job-tracking log
Refers to the data sets used by IBM Workload Scheduler for z/OS to log updates to the current plan and to record audit information for requested files. You *must* use at least two job-tracking logs, referenced by ddnames EQQJT01 and EQQJT02. You can use up to 99 job-tracking logs. The JT logs are used in a cyclic manner, and IBM Workload Scheduler for z/OS automatically switches to the next available JT log after a CP backup. The data from the inactive data set is copied to the archive log, and the data set is emptied in preparation for future use. You should use at least 5 job-tracking logs. This is the default number on the JTLOGS keyword of JTOPTS.

Current and inactive dual job-tracking log
If the dual logging function is requested, IBM Workload Scheduler for z/OS duplicates the JT records in the corresponding dual JT log. Dual logs are switched at the same time, and in the same sequence, as the JT logs. So the number of dual job-tracking data sets is determined by the number of normal job-tracking data sets.

Job-tracking archive log
Represents the accumulated job-tracking data since the new current plan was created. When the JT log is switched, the data from the inactive data set is appended to the archive log. The archive log is copied to the data set referenced by the EQQTROUT ddname during the daily planning process. When IBM Workload Scheduler for z/OS takes over the new current plan, the archive data set is emptied.

Checkpoint
Refers to the EQQCKPT data set, which contains information about the current status of the IBM Workload Scheduler for z/OS system, including which current plan and job-tracking data sets are currently active.

Symphony file
Represents a local plan for a set of fault-tolerant workstations and is updated accordingly to the local and current plan changes.

The basic principle of IBM Workload Scheduler for z/OS current plan recovery is that if the active current plan becomes unusable for any reason, IBM Workload Scheduler for z/OS should always be able to re-create an up-to-date current plan from the backup current plan and the various job-tracking logs. The way IBM Workload Scheduler for z/OS actually carries out this task is described in "Current-plan recovery processing" on page 333.

Current-plan recovery processing

About this task

When IBM Workload Scheduler for z/OS suspects that the active current plan is unusable, it automatically carries out recovery processing. IBM Workload Scheduler for z/OS uses the alternate current plan or the new-current-plan data set (EQQNCPDS) and the active job-tracking log to create a current plan that is fully up-to-date. Here is a step-by-step description of the current plan recovery process:

1. The current plan is locked to prevent updates from job-tracking events and dialog users.
2. The active current plan is erased.
3. The alternate current plan or new current plan is copied to the active current plan. Indicators in the checkpoint data set determine which of the two are actually used. This is explained further in "Current-plan processing at IBM Workload Scheduler for z/OS startup" on page 334.
4. The identity of the active job-tracking log is obtained from the checkpoint record. Every record of the current JT log is used to update the active current plan.

   If recovery is performed from the new current plan, the current JT log and the JT archive log are used to reapply the events that have occurred since the new current plan was created.
5. The active current plan is now up-to-date. A current plan backup is performed.
6. Normal IBM Workload Scheduler for z/OS processing starts or continues.

If you are scheduling end-to-end with fault tolerance capabilities, perform the following manual actions to make sure that the Symphony file is aligned with the rebuilt current plan:

1. From OPC dialog select the option 3, DAILY PLANNING. The Producing OPC Daily Plans dialog is displayed.
2. Then, select option 5, SYMPHONY RENEW.
3. Submit the symphony renew batch job to create a Symphony file aligned with the Current Plan.

Current plan recovery is performed for these situations:

- During IBM Workload Scheduler for z/OS startup, if the current plans (CP1 and CP2) are not equal.
- During IBM Workload Scheduler for z/OS startup, if CURRPLAN(NEW) has been specified on the JTOPTS statement.
- During normal IBM Workload Scheduler for z/OS processing, if the active current plan becomes damaged or is not accessible.

If the data space CX file becomes unusable, IBM Workload Scheduler for z/OS performs these recovery actions:

1. The current plan is locked to prevent updates from job-tracking events and dialog users.
2. The CX data space is deleted.
3. The CX DASD file is copied to a new data space
4. The identity of the active job-tracking log is obtained from the checkpoint record. Records in the current JT log are used to update the data space.
5. The data space file is now up-to-date. A current plan backup is performed.
6. Normal IBM Workload Scheduler for z/OS processing starts or continues.

When IBM Workload Scheduler for z/OS performs recovery from the new current plan, the EQQNCXDS file is copied to EQQCXDS, a data space is created, and events are reapplied using the JT archive log and the current JT log. The data space is refreshed to the DASD file during the subsequent current plan backup.

If the CX DASD file becomes unusable, follow the instructions in "Recovering from errors on the current-plan-extension data set" on page 343.

Current-plan processing at IBM Workload Scheduler for z/OS startup

This section describes the two current plan processing conditions that can occur when:
- Starting IBM Workload Scheduler for z/OS with an empty checkpoint data set
- Starting IBM Workload Scheduler for z/OS with a valid checkpoint data set.

Starting IBM Workload Scheduler for z/OS with an empty checkpoint data set

About this task

The checkpoint data set is empty the first time IBM Workload Scheduler for z/OS is started. It is also empty if it has been deleted and reallocated for some reason, such as if it was damaged.

When IBM Workload Scheduler for z/OS is started for the first time, the following occurs:
1. The checkpoint data set is formatted and initial values written to it.
2. When JTOPTS CURRPLAN(CURRENT) is specified, IBM Workload Scheduler for z/OS will issue message

   ```
   EQQN026W A NEW CURRENT PLAN (NCP) HAS BEEN REJECTED
   ```

   and stop processing.
3. When JTOPTS CURRPLAN(NEW) is specified, IBM Workload Scheduler for z/OS carries out recovery processing using the new current plan (EQQNCPDS and EQQNCXDS) as described in "Current-plan recovery processing" on page 333. There might be a new current plan if you are migrating from a previous release or version and you have placed the converted current plan in the new current plan data sets as part of the migration procedure.

   If the new current plan is empty, recovery processing ends, IBM Workload Scheduler for z/OS becomes active without a current plan, and job tracking is not started.
4. If recovery processing with the new current plan is successful, IBM Workload Scheduler for z/OS starts normal processing.

If you are scheduling end-to-end with fault tolerance capabilities, perform the following manual actions to make sure that the Symphony file is aligned with the rebuilt current plan:
1. From OPC dialog select the option 3, DAILY PLANNING. The Producing OPC Daily Plans dialog is displayed.
2. Then, select option 5, SYMPHONY RENEW.
3. Submit the symphony renew batch job to create a Symphony file aligned with the Current Plan.

If you start IBM Workload Scheduler for z/OS after having deleted and reallocated the checkpoint data set, and you specified JTOPTS CURRPLAN(NEW), the following occurs:

1. IBM Workload Scheduler for z/OS carries out recovery processing with the new current plan as described in "Current-plan recovery processing" on page 333.
2. IBM Workload Scheduler for z/OS normal processing starts.

Note: When current plan recovery processing is performed after IBM Workload Scheduler for z/OS is started with an empty checkpoint data set, IBM Workload Scheduler for z/OS uses the primary data sets EQQJT01, EQQDL01, and EQQJS1DS as the active data sets. If a primary data set was not the active data set at the last shut down, copy the data from the previously active data set to the primary data set. Or events since the last current plan backup will not be applied. You can check which data sets were active by reviewing the message log or by looking for the data sets with the latest time stamp.

Starting IBM Workload Scheduler for z/OS with a valid checkpoint data set

A valid checkpoint data set should exist even if IBM Workload Scheduler for z/OS ended unsuccessfully the previous time.

When IBM Workload Scheduler for z/OS starts, it reads the checkpoint data set to determine which is the active current plan and job-tracking log.

The job-tracking log is opened. If no job-tracking records are found, this indicates that IBM Workload Scheduler for z/OS ended normally because a JT log switch is performed when the CP backup completes at normal termination.

If there are records on the job-tracking log after the backup position, this indicates that IBM Workload Scheduler for z/OS did not end normally. In this case, recovery processing using the backup current plan and CX data set is performed as described in "Current-plan recovery processing" on page 333. Normal IBM Workload Scheduler for z/OS processing is then started.

If the Symphony file is not up to date with the Current Plan, select the option 5 on the panel Producing OPC Daily Plans or submit the Daily Plan batch job.

Avoiding corruption of the current plan backup

If an error occurs in the current plan while IBM Workload Scheduler for z/OS is running and is not detected by IBM Workload Scheduler for z/OS, it is strongly recommended that you **cancel** the controller address space instead of stopping it as is normally done. This prevents copying a logical error from the active current plan to the alternate current-plan data set during the backup process. When IBM Workload Scheduler for z/OS is restarted, the active current plan will be re-created from the alternate current plan. This will remove any errors in the active current plan.

If IBM Workload Scheduler for z/OS recognizes that the active current plan is damaged or is no longer accessible, a recovery of the active current plan is performed automatically. This is described in "Current-plan recovery processing" on page 333.

Restoring a damaged IBM Workload Scheduler for z/OS file from backup

About this task

The VSAM data sets used by IBM Workload Scheduler for z/OS must be error free if IBM Workload Scheduler for z/OS is to work normally. If a VSAM data set has been damaged, then it must be restored from a backup copy.

When re-creating a VSAM data set from a backup, you should allocate a new data set. But keep the damaged data set for problem determination after normal IBM Workload Scheduler for z/OS service is resumed. You need to either rename the damaged file so that you can use the same name for the new data set, or allocate the data set with a new name.

If you allocate the data set with a new name you must also change the IBM Workload Scheduler for z/OS JCL procedure and all batch jobs that reference the damaged data set. You can update the batch jobs by editing affected members in the ISPF job-skeleton library.

The restore procedure varies a little, depending on which data set is damaged. These differences are explained in the following sections. All restore procedures assume that there is a usable backup available.

Note: If you restore a database file, AD, WS, OI, RD, or SI, IBM Workload Scheduler for z/OS cannot recover updates made since the last backup. You should consider using the AUDIT statement to log accesses to these files so that you have a record of the updates that you need to reapply.

You can also try to access the file before you stop IBM Workload Scheduler for z/OS and sort the list of items on the time of last update in descending order. Note the items changed since the last backup was created.

Restoring the Workstation Description (WS) data set

About this task

To restore the WS data set:

1. Stop IBM Workload Scheduler for z/OS.
2. Allocate a new WS data set.
3. Copy the backup data set to the WS data set.
4. If you allocated the data set with a new name, update all JCL (ddname EQQWSDS) to reference the new WS data set.
5. Start IBM Workload Scheduler for z/OS.
6. Enter the Calendar dialog to verify that the calendar is up-to-date. Reapply any changes made since the last backup of the WS file.
7. Enter the Workstation Description dialog to verify that workstations are correctly defined and that open-time-interval definitions are up-to-date. Reapply any changes made since the last backup of the WS file.

Restoring the Application Description (AD) data set

About this task

To restore the AD data set:

1. Stop IBM Workload Scheduler for z/OS.
2. Allocate a new AD data set.
3. Copy the backup data set to the AD data set.
4. If you allocated the data set with a new name, update all JCL (ddname EQQADDS) to reference the new AD data set.
5. Start IBM Workload Scheduler for z/OS.
6. Enter the Application Description dialog to verify that applications are correctly defined. Reapply all changes since the AD backup was created.

Restoring the Operator Instruction (OI) data set

About this task

To restore the OI data set:

1. Stop IBM Workload Scheduler for z/OS.
2. Allocate a new OI data set.
3. Copy the backup data set to the OI data set.
4. If you allocated the data set with a new name, update all JCL (ddname EQQOIDS) to reference the new OI data set.
5. Start IBM Workload Scheduler for z/OS.
6. Enter the Operator Instruction dialog to verify that instructions are correctly defined. All instructions added since the OI backup was created must be added again.

Restoring the Special Resource Description (RD) data set

About this task

To restore the RD data set:

1. Stop IBM Workload Scheduler for z/OS.
2. Allocate a new RD data set.
3. Copy the backup data set to the RD data set.
4. If you allocated the data set with a new name, update all JCL (ddname EQQRDDS) to reference the new RD data set.
5. Start IBM Workload Scheduler for z/OS.
6. Enter the Special Resource Descriptions dialog to verify that special resources are correctly defined. Reapply all changes made since the RD backup was created.

Restoring the Side Information (SI) data set

About this task

To restore the SI data set:

1. Stop IBM Workload Scheduler for z/OS.
2. Allocate a new SI data set.
3. Copy the backup data set to the SI data set.

4. If you allocated the data set with a new name, update all JCL (ddname EQQSIDS) to reference the new SI data set.
5. Start IBM Workload Scheduler for z/OS.
6. Enter the ETT dialog and verify that ETT criteria are correctly defined. Reapply all changes made since the SI backup was created.

Note:

- It is recommended that you back up the EQQSIDS data set while the controller is down. If the EQQSIDS data set is backed up while the controller is active, some SI updates might be lost because of LSR buffering.
- The EQQSIDS data set cannot be backed up if a CP backup is in progress, because of ongoing LSR buffer delete and allocate phase.
- A backup of the EQQSIDS data set taken after the completion of a CP backup, contains any SI update made prior to the start of the CP backup; any following backup needs to be manually reapplied by the user.

Restoring the Long-term Plan (LTP) data set

About this task

The way you restore the LTP data set depends on when the backup data set was created. There is a strong connection between the current plan and the LTP data set. If possible, the LTP should be restored from a backup that was created after the last time IBM Workload Scheduler for z/OS took over a new current plan created by a daily plan batch job.

If such an LTP backup is available, use it to restore the LTP data set:

1. Stop IBM Workload Scheduler for z/OS.
2. Allocate a new LTP data set.
3. Copy the backup data set to the LTP data set. Use the data set with ddname EQQLTBKP.
4. If you allocated the data set with a new name, update all JCL (ddname EQQLTDS) to reference the new LTP data set.
5. Start IBM Workload Scheduler for z/OS.
6. Enter the Long-Term Plan dialog to verify that occurrences are correctly defined. All occurrences added since the LTP backup was created must be added again.

If the LTP backup that matches the current plan is unusable, you must restore the LTP as described previously and create a new current plan from the restored LTP.

Restoring the JCL Repository (JS) data set

About this task

The JS data set consists of two data sets, one that is active and one that is inactive. If there is an error on the active data set, you can copy the inactive data set to the active data set to recover. Perform the following procedure:

1. If IBM Workload Scheduler for z/OS is active, stop it.
2. Allocate a new data set.
3. Copy the inactive data set to the new JS data set.
4. If you allocated the data set with a new name, update all JCL (EQQJSnDS) to reference the new data set.

5. Review the IBM Workload Scheduler for z/OS message log to determine when the damaged JS data set was first used.
6. Start IBM Workload Scheduler for z/OS.
7. Redo all IBM Workload Scheduler for z/OS JCL editing that has been done since the damaged JS data set was first used and that applies to operations that have not yet started.

Re-creating the current plan from the long-term plan

About this task

If the current plan has, for some reason, become unusable, you can build a new current plan from the long-term plan after using the REFRESH function.

Attention: You should use the REFRESH function only when you have no alternative because it *deletes your current plan*. Make sure that you cannot recover using the backup current plan or new current plan before you use REFRESH.

To refresh the current plan:
1. In the SERVICE FUNCTION panel, select option 5, REFRESH (you must be authorized to use this function). The scheduler displays a confirmation panel.
2. Type `Y` to start the refresh. The request is sent to the subsystem for processing by the normal mode manager task.

If there are some fault-tolerant workstations in the network, the normal-mode manager task sends them a request to stop and then updates the checkpoint data set and the long-term plan data set to show that the current plan no longer exists. The normal-mode manager stops all active event readers and the NCF function if it is active. From this moment, all dialogs that reference the current plan are unavailable.

To re-create the current plan:
1. Enter the IBM Workload Scheduler for z/OS dialog, and select option `3, DAILY PLANNING` on the main menu.
2. On the panel Producing OPC Daily Plan, select option `2, EXTEND`. Define the start and end of the new plan, and submit the daily plan batch job to create a new current plan.
3. On the panel Service Functions, select option `1, DEACTIVATE` job submission.
4. When a new current plan is created by the batch job, it is taken over by IBM Workload Scheduler for z/OS. All IBM Workload Scheduler for z/OS dialogs are then made available again. Once this happens, enter the Modify Current Plan dialog to set the correct status for all operations in the current plan. All application occurrences that have already completed should be deleted, and occurrences that are in progress must be updated with current information.
5. When all operations have been given the correct status, enter the panel Service Functions, and enable job submission again.

Note: If the refresh function is selected in error, you can re-create the current plan from the new current plan. See "Re-creating the current plan from the new-current-plan and the JT archive log" on page 340.

Re-creating the current plan from the new-current-plan and the JT archive log

About this task

If all current plan data sets have been damaged or lost but the new current plan still exists and is valid, it is possible to get IBM Workload Scheduler for z/OS to take over the new current plan again. This is a better alternative than using refresh because the resulting current plan will normally be up-to-date.

Perform the following procedure:

1. Stop IBM Workload Scheduler for z/OS.
2. Add keyword CURRPLAN(NEW) to the JTOPTS statement. When you start IBM Workload Scheduler for z/OS, it will:
   a. Use the new current plan data sets (EQQNCPDS and EQQNCXDS) as input
   b. Copy EQQNCPDS to either EQQCP1DS or EQQCP2DS
   c. Copy EQQNCXDS to EQQCXDS and then load the CX data space from EQQCXDS.
   d. Apply all events from the job-tracking archive log (EQQJTARC). If IBM Workload Scheduler for z/OS did not end normally, events are applied also from the job-tracking log that was in use when IBM Workload Scheduler for z/OS ended.

   In this way, IBM Workload Scheduler for z/OS re-creates the current plan.
3. Start IBM Workload Scheduler for z/OS.
4. Remove CURRPLAN(NEW) from the JTOPTS statement.

If you are scheduling end-to-end with fault tolerance capabilities, perform the following manual actions to make sure that the Symphony file is aligned with the rebuilt current plan:

1. From OPC dialog select the option 3, DAILY PLANNING. The Producing OPC Daily Plans dialog is displayed.
2. Then, select option 5, SYMPHONY RENEW.
3. Submit the symphony renew batch job to create a Symphony file aligned with the Current Plan.

Recovering from errors on the job-tracking log

About this task

IBM Workload Scheduler for z/OS automatically recovers from read errors on a job-tracking log during restart. If the error occurs on the first record of the log, the NMM task regards the job-tracking log as empty. A read error on a record other than the first is treated as end-of-file.

When there are write errors on a job-tracking log, IBM Workload Scheduler for z/OS tries recovery. If an inactive job-tracking log is available, IBM Workload Scheduler for z/OS switches to that data set and continues processing. You can then shut down IBM Workload Scheduler for z/OS to correct the problem with the data set. If IBM Workload Scheduler for z/OS is unable to switch to an inactive JT log, you need to:

1. Stop IBM Workload Scheduler for z/OS.
2. Reallocate the job-tracking-log data set.

3. If a dual JT log is available, copy the data from the corresponding dual logging data set into the new JT log data set.
4. Start IBM Workload Scheduler for z/OS.

If you are not using the dual-logging function, you will lose the job-tracking data, but your current plan should not be impacted. But there might be an impact if recovery is required at a later stage of the current plan. So extend or replan the current plan as soon as possible.

Dual JT log data set problems

About this task

When IBM Workload Scheduler for z/OS finds an error with a dual job-tracking data set, the dual-logging process is disabled. The IBM Workload Scheduler for z/OS current plan and normal job-tracking logs continue as normal. You should stop IBM Workload Scheduler for z/OS, and reallocate the dual JT log data set. The dual-logging function is activated again when the controller is started.

Recovering from errors on the JT archive log

About this task

If there is a write error on the JT archive log, follow this procedure:
1. Stop IBM Workload Scheduler for z/OS.
2. Rename the data set to a temporary name.
3. Allocate a new JT archive-log data set.
4. Copy the old data set into the new data set. This can be done by IEBGENER or IDCAMS REPRO.
5. Start IBM Workload Scheduler for z/OS.

If there is a read error on the JT archive log, follow this procedure:
1. Stop IBM Workload Scheduler for z/OS.
2. Delete the data set, and allocate a new JT archive-log data set.
3. Start IBM Workload Scheduler for z/OS, and submit a daily plan job as soon as possible. Current plan recoverability is jeopardized while you are running an incomplete JT archive log.

Attention: Do not start IBM Workload Scheduler for z/OS with the CURRPLAN(NEW) keyword specified on the JTOPTS statement until a new current plan is taken over by IBM Workload Scheduler for z/OS.

Recovering from errors on the checkpoint data set

About this task

If there is a write error on the checkpoint data set, follow this procedure:
1. Stop IBM Workload Scheduler for z/OS.
2. Rename the checkpoint data set to a temporary name.
3. Allocate a new checkpoint data set.
4. Copy the old checkpoint data set into the new data set. This can be done by ISPF COPY or by IDCAMS REPRO.

5. Start IBM Workload Scheduler for z/OS again.

If there is a read error on the checkpoint data set, follow this procedure:

- If a good new-current-plan does not exist:
  1. Stop IBM Workload Scheduler for z/OS.
  2. Delete the checkpoint data set and reallocate it.
  3. Re-create the current plan using the refresh procedure. See “Re-creating the current plan from the long-term plan” on page 339.
- If a good new-current-plan data set exists:
  1. Stop IBM Workload Scheduler for z/OS.
  2. Check which job-tracking log is the current one. This can be done by reviewing the messages in the message log, or by browsing the JT log and checking the time stamp in position 13 in the first record of the data set. The data set with the latest time stamp in the first record is current.
  3. Copy the data from the active job-tracking log into the job-tracking log referenced by the EQQJT01 ddname.
  4. Determine which JS file was active. If EQQJS1DS defines the current data set, then continue with the next step. Otherwise, either copy the EQQJS2DS to the EQQJS1DS or switch the ddnames in the JCL procedure.
  5. Delete and reallocate the IBM Workload Scheduler for z/OS checkpoint data set.
  6. Change the JTOPTS statement to specify JOBSUBMIT(NO) and CURRPLAN(NEW), and start the scheduler.
  7. Enter the Modify Current Plan dialog to set correct status for all operations in the current plan.
  8. When all operations have correct status, enter the SERVICE FUNCTIONS panel and enable job submission again. Restore the JTOPTS statement if you changed it.

If you are scheduling end-to-end with fault tolerance capabilities, perform the following manual actions to make sure that the Symphony file is aligned with the rebuilt current plan:

1. From OPC dialog select the option 3, DAILY PLANNING. The Producing OPC Daily Plans dialog is displayed.
2. Then, select option 5, SYMPHONY RENEW.
3. Submit the symphony renew batch job to create a Symphony file aligned with the Current Plan.

Recovering from errors on event data sets

About this task

If an event data set (EQQEVDS or EQQHTTP0), an input event data set (EQQTWSIN), or an output event data set (EQQTWSOU) has been damaged, follow this procedure:

Procedure

1. Stop any IBM Workload Scheduler for z/OS subsystem that is writing to or reading from the data set.
2. Rename the event data set, and allocate a new data set.
3. Start IBM Workload Scheduler for z/OS again.

Note: The first time IBM Workload Scheduler for z/OS is started with a newly allocated event data set, an SD37 error occurs when IBM Workload Scheduler for z/OS formats the data set. Expect this; do not treat it as an error.

4. Enter the Modify Current Plan dialog, and check the status of operations in the current plan. Concentrate on automatically reporting workstations whose events were written to the damaged event data set. If any discrepancies are found, set the correct status for the operation. It is unlikely that you will have lost any events.

Recovering from errors on a submit/release data set

About this task

If a submit/release data set has been damaged, follow this procedure:

1. Stop any IBM Workload Scheduler for z/OS subsystem that is writing to or reading from the data set.
2. Delete the submit/release data set, and allocate a new data set.
3. Start IBM Workload Scheduler for z/OS again.

   Note: The first time IBM Workload Scheduler for z/OS is started with a newly allocated submit/release data set, an SD37 error occurs when IBM Workload Scheduler for z/OS formats the data set. Expect this; do not treat it as an error.
4. Enter the Modify Current Plan dialog, and check the status of operations in the current plan. Concentrate on computer workstations that are connected to the controller via the failing submit/release data set. If you determine that a job submission action has been missed, reset the operation to READY status.

Recovering from errors on the current-plan-extension data set

About this task

IBM Workload Scheduler for z/OS maintains the current-plan-extension file in a data space. The data space is loaded from the EQQCXDS file on DASD. When a current plan backup occurs, the data space is refreshed to the EQQCXDS data set. If an error occurs on the data set, IBM Workload Scheduler for z/OS attempts initial recovery by copying the data space to the data set. If this is not possible, the file must be re-created.

If the EQQNCPDS and EQQNCXDS data sets are valid, perform these actions:

1. Cancel IBM Workload Scheduler for z/OS.
2. Allocate a new CX data set.
3. If you allocated the data set with a new name, update all JCL (ddname EQQCXDS) to reference the new data set.
4. Specify CURRPLAN(NEW) on the JTOPTS statement.
5. Start IBM Workload Scheduler for z/OS.
6. Remove CURRPLAN(NEW) from the JTOPTS statement.

If you are scheduling end-to-end with fault tolerance capabilities, perform the following manual actions to make sure that the Symphony file is aligned with the rebuilt current plan:

1. From OPC dialog select the option 3, DAILY PLANNING. The Producing OPC Daily Plans dialog is displayed.
2. Then, select option 5, SYMPHONY RENEW.

3. Submit the symphony renew batch job to create a Symphony file aligned with the Current Plan.

If the EQQNCXDS data set is valid but no valid EQQNCPDS data set exists, perform these actions:

1. Cancel IBM Workload Scheduler for z/OS.
2. Allocate a new CX data set.
3. If you allocated the data set with a new name, update all JCL (ddname EQQCXDS) to reference the new data set.
4. Copy EQQNCXDS to EQQCXDS.
5. Specify CURRPLAN(CURRENT) on the JTOPTS statement. CURRENT is the default value.
6. Start IBM Workload Scheduler for z/OS.

If both EQQNCPDS and EQQNCXDS cannot be used, you must create a new current plan from the LTP. “Re-creating the current plan from the long-term plan” on page 339 describes how you do this.

Automatic recovery from controller failures

If your controller fails or the system that it is running on fails, IBM Workload Scheduler for z/OS can transfer the controller function to another z/OS system that is running IBM Workload Scheduler for z/OS. The systems involved must be part of a cross-system coupling facility (XCF) sysplex. Control can be passed automatically when IBM Workload Scheduler for z/OS detects a tracker or z/OS system failure or can be initiated manually using the MVS MODIFY operator command.

The standby system to which control is passed must have all the data used by the original controller available to it. This usually requires the use of shared DASD between the two systems. The new controller system must also have access to all other systems in the IBM Workload Scheduler for z/OS complex via XCF, VTAM, or shared SUBMIT/RELEASE data sets. You must ensure that these links are available.

Also, the RACF environment of the new controller z/OS system must be set up correctly to ensure that the level of user access to data from the standby system is correct.

For more details on IBM Workload Scheduler for z/OS hot standby, refer to *Planning and Installation*.

Notification of controller failures

You can specify that IBM Workload Scheduler for z/OS produces a message if the controller or one of its components fails. This notification can be sent to the operator console or directly to NetView using the NetView program-to-program interface.

You specify whether the message is produced, and its destination, using the ALERTS statement. For detailed information about this statement, see “ALERTS” on page 7.

Re-creating the symphony file from the current plan

About this task

If the process for creating the Symphony fails, you can create a new Symphony file based on the current plan, if the plan is up-to-date. For more information about creating the Symphony file from the current plan see"Current-plan recovery processing" on page 333.

To re-create the Symphony file:

1. Enter the OPC Dialog and select the option 3, Daily Planning on the main menu.
2. On the panel Producing OPC Daily Plans, select the option 5, SYMPHONY RENEW, then submit the symphony renew batch job to create the Symphony file.

When the Symphony file is created by the batch job, it is distributed to the fault-tolerant workstation in order to start the scheduling activity again.

Chapter 11. Cleanup and recovery of data store data sets

The data store component is equipped with a set of utilities that can be executed 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 will run on a periodic basis, and delete selected records from the database. It is highly recommended that this mode is used, to ensure a regular cleanup of old SYSOUT records and to keep the database at a reasonable size. See below for a full discussion of the cleanup subtask.

Each batch utility is activated by a separate keyword of the DSTUTIL command, as described below.

The base utilities are the following:

cleanup
: *command* DELUNSTR *command* DELSTRUC *command* DELBOTH

export
: *command* EXPUNSTR *command* EXPSTRUC *command* EXPBOTH

import
: *command* IMPORT

recover primary index
: *command* RECOVER

recover secondary index
: is obtained starting from the recover file of the primary index

For both primary and secondary index another utility, the reorg, can be obtained by combining an export and an import utility.

Deleting data from the database

You can delete data from the database using the one of the following keywords of the DSTUTIL command:

DELUNSTR
: To delete the unstructured data

DELSTRUC
: To delete the structured data

DELBOTH
: To delete the structured and unstructured data

It is not possible to code separate cleanup actions for structured and unstructured data because the data store cleanup subtask accepts only one DSTUTIL statement. Therefore, to delete structured and unstructured data with different DSTUTIL statements, use the batch cleanup utility (EQQDSCL batch cleanup sample). The cleanup batch job can run only if the data store is stopped. If you decide to schedule data store cleanup using the EQQDSCL job, set DSTOPTS CINTERVAL(0). For details, see “DSTUTIL” on page 48.

Example:

```
DSTUTIL DELUNSTR
         SEARCH1(JBNMLKJOB00*,
                 SYCLEQU)
         SEARCH2(OLDRMM2)
         SEARCH3(JBIDEQJOB00467)
```

This command deletes records that match the following criteria:

```
  JobName  LIKE  'JOB00*'  and  SysClass = 'U'

or

  job  OLDER  then 2 months

or

  JobId  =  'JOB00467'
```

Exporting data to a backup file

You can export selected SYSOUT records to a sequential file using one of the following keywords of the DSTUTIL command:

EXPUNSTR
: to export the selected unstructured data to an unstructured export file

EXPSTRUC
: to export the selected structured data to a structured export file

EXPBOTH
: to obtain, with a single-command process, the result of the two preceding commands.

The EXPUNSTR keyword of the DSTUTIL command is used to export selected SYSOUT records to a sequential file. See Chapter 1, “Defining initialization statements,” on page 3, under *DSTUTIL* for details.

Example:

```
DSTUTIL IMPUNSTR DDNAME(EQQEXP01)
     SEARCH1(JBNMLKJOB*CC*,JBDTGE19990501)
     SEARCH2(JBDTGE19990515)
```

This command exports the records that match the criteria to a sequential file identified by the DDNAME EQQEXP01:

```
  JobName  LIKE  'JOB*CC*' and jobdate  greater than or
                                              equal to 01/05/1999

or

  jobdate greater equal than  15/05/1999
```

Importing data from a backup file

SYSOUT records that have been previously exported to a sequential file can be imported back into the database by using the IMPORT keyword of the DSTUTIL command. Only the records matching the search criteria are imported. See Chapter 1, “Defining initialization statements,” on page 3, under *DSTUTIL* for details.

Example:

```
DSTUTIL IMPORT DDNAME(EQQEXP01) REPLACE(YES)
     SEARCH1(JBIDLK*)
```

This command imports all SYSOUT records contained in the export file identified by DDNAME EQQEXP01. The only condition specified is a wildcard filter that will match every record in the file:

```
JobId  LIKE  '*'
```

The option REPLACE(YES) means matching records in the database will be overwritten by those imported from the backup file.

You can import separately structured and unstructured data by codifying the command IMPORT with two different ddnames. The import utility recognizes the export data type (structured and unstructured) because of the header record that provides it.

Example

```
DSTUTIL IMPORT DDNAME(EQQEXPST) REPLACE(YES) ← struct. exp. file SEARCH1(JBIDLK*)
DSTUTIL IMPORT DDNAME(EQQEXPUN) REPLACE(YES) ← unstruct. exp. file SEARCH1(JBIDLK*)
```

Note: The command shown above can be used after the whole database, has been reorganized after a previous mass export.

Recovering from failure

The keyword RECOVER of the DSTUTIL command, analyses all of the records in the database, and extracts the keys for each. This information is written to a file identified by ddname EQQPKREC (this name cannot be changed). The file obtained in this way can be used to recover the primary index and ensure consistency with the data. Also, the secondary index recovery file can be obtained from the EQQPKREC file, which represents the input to recovery processes. This utility is useful if either the indexes or the data have been corrupted. For more information, see Chapter 1, “Defining initialization statements,” on page 3, under *DSTUTIL*.

What to do when data files fill up

If data store VSAM files that you have defined are too small or there are not enough of them to store all job logs you need to be online, you might encounter the files fill-up condition that hangs all data store activities. If this happens, there are two things you can do:

- Enlarge primary index, secondary index and data files sizes.
- Define some new data files; to do this you need to define one or more VSAM Data Files and to add them to data store startup procedure, using correct DDNAMEs (refer to *Planning and Installation*); the files are automatically initialized at the first data store startup.

  Note: The primary index is always reconstructed during IMPORT phase. It is also possible to combine the two strategies.

Cleanup subtask

The cleanup subtask is not a batch utility, but an online component of the data store that is responsible for regularly eliminating superfluous records from the database. The records that are to be selected for deletion are identified according to user-defined criteria, which are specified in a member of the data store parameter library. Typically, records older than a certain interval of time, for example 1 day, are deleted, but the selection criteria allow you to treat some jobs differently from others.

The name of the member that contains the cleanup selection criteria is specified in the CLNPARM initialization statement for the data store. The syntax for the selection criteria is identical to that described for the batch cleanup utility above. When coding the selection criteria, be sure to include a catch-all criteria that matches all records and ensures that after a certain period of time all records are eventually deleted. This is essential to prevent the database from growing over time until it fills up and cannot store any more data. Syntax errors in the cleanup selection criteria cause the closure of the task, and cleanup actions are not therefore performed. In this case, you should correct the errors and restart the cleanup task manually, specifying S=ARCU on the modify command (P=ARCU stops the subtask). The cleanup subtask is activated on a periodic basis. The frequency with which it is activated is governed by the CINTERVAL initialization statement. A value of zero means that the Data Store Cleanup subtask is started at data store initialization, but never runs. Therefore, if CINTERVAL(0) is set, the user must schedule regular batch cleanups, otherwise the data store UDF and SDF files grow with no limits.

Chapter 12. Disaster recovery planning

Disaster Recovery Planning (DRP) reduces the business impact of a disaster. DRP ensures that the *business* supported by the data center remains viable. In the event of a disaster at the primary data center, full operations and production work would move to a secondary data center. By using IBM Workload Scheduler for z/OS to direct the recovery effort, you can ensure that the required recoveries are run in sequence, with minimal delays, and that *every* error is detected and reported. This chapter describes:

- Designing your IBM Workload Scheduler for z/OS DRP.
- Implementing your IBM Workload Scheduler for z/OS DRP.

Before you plan for the recovery of data, have a good understanding of the comprehensive built-in recovery actions provided by IBM Workload Scheduler for z/OS. This reading is recommended:

- The *Planning and Installation* chapter on IBM Workload Scheduler for z/OS configurations
- Current plan reference information in the chapter about producing the current plan, which is described in *Managing the Workload*
- Chapter 10, "Backup and recovery of data sets," on page 329.

Designing your IBM Workload Scheduler for z/OS DRP

The complexity of your IBM Workload Scheduler for z/OS DRP depends on the recovery strategies of your business systems. The following sections explain the procedures for an IBM Workload Scheduler for z/OS recovery plan that best meets your business DRP strategies:

- Recovery to start-of-day
- Recovery to a predefined point in processing
- Recovery to point-of-failure.

If your business uses more than one of these recovery strategies, your DRP should be designed for the most complex. Consider this example:

Under normal circumstances, the DP center handles the processing of six business systems. Only four of these systems provide off-site recoverability. Three of the four send backups off-site each hour; the other system creates backups only when processing is complete each day. The IBM Workload Scheduler for z/OS DRP required to support this environment must generate off-site backups each hour. After IBM Workload Scheduler for z/OS is recovered to the most recent backup in the secondary center, application occurrences for those systems that recover to start-of-day are reset to waiting status. Occurrences not required are deleted from the current plan.

Secondary-center options

The secondary center determines how quickly you can get an IBM Workload Scheduler for z/OS environment active:

Hot backup
: A secondary data center already operational and available for *immediate* switch-over

Warm backup
A secondary data center, possibly operational, available for switch-over after some delay (measured in hours or days)

Cold backup
A secondary data center with little or no equipment installed

Pooling
A secondary data center or cold site shared by several users (or subscribers).

The distance between the centers is often the determining factor in your installation's choice of communication options. Data must be brought regularly to the secondary data center from the primary data center to provide disaster backup and recovery. This can be done by:
- Physical transportation of data on media (tape, paper)
- Telecommunication lines
- Channel-to-channel connections
- Fiber-optic channel extenders.

If you have shared DASD with the secondary center either through channel-to-channel connectors or fiber-optic channel extenders, you can recover IBM Workload Scheduler for z/OS to point-of-failure by using the dual job-tracking facilities of IBM Workload Scheduler for z/OS.

Unless your secondary center is a hot backup that exactly mirrors your primary data center, you must identify the IBM Workload Scheduler for z/OS configuration needed to support the environment. With your system programmer, identify the IBM Workload Scheduler for z/OS subsystems required.

Standardizing the environment

Where possible, design the environment so that it duplicates your primary center configuration as much as possible and adopt procedures to ensure that applicable software and configuration changes are reflected in the secondary center.

Naming conventions

Try to use the same subsystem names, NCF LU names, and XCF destination names. This can save you time in the recovery process and lets you use the same initialization statements that you normally use in the primary center.

Library requirements

The SYS1.PARMLIB requirements to support your secondary-center configuration should be permanently defined on the SYSRES used for the DRP exercise. Also maintain up-to-date IBM Workload Scheduler for z/OS software libraries in the secondary center.

Capacity and workload variations

You can direct work to the required MVS image by changing the destination specifications on the applicable workstation definitions. If the work can be separated into logical units of processing (for example, IMS BMPs, or DB2), then consider permanently defining these operations to separate workstations. This implementation can provide everyday benefits for processing in the primary center as well. For example, if you define all the IMS BMPs to a processor workstation reserved for that purpose, then you have more power to control the processing of activities, such as:

- Controlled shut down for scheduled outages to the control region

- Varying the number of BMPs running concurrently depending on the time of day
- Responding to extended recovery facility (XRF) takeovers in cases where z/OS is unaffected
- Handling scheduling of operations during unscheduled outages.

By defining the operations in this way, you can move the entire workload to any destination in your IBM Workload Scheduler for z/OS configuration.

JCL considerations

Where possible, standardize JCL requirements for symbolic variables, such as input and SYSOUT classes between the primary and secondary centers. If differences are unavoidable, consider using the IBM Workload Scheduler for z/OS JCL variable substitution function to reduce the impact of these changes.

Symbolic variables can be permanently defined in the global table for input classes, SYSOUT classes, and SMS storage classes. Utilization of JCL variable substitution for these items can also prevent JCL changes in your primary center. For detailed information, see *Managing the Workload*.

Automation in the secondary center

The extent of automation in the primary data center will, to some degree, determine the environment in the secondary data center. Consider these recommendations:

- Automatic job recovery should not be used until the business systems have completed at least one full processing cycle.
- Automatic workload restart and reroute should be switched to manual control.
- The reporting attribute of WTO workstations should be set to manual start and complete if full NetView services are not available.
- Continue to use ETT if you normally do so, but if your current plan is not extended to the current time, application occurrences added by ETT to the CP fail. When this happens, a message is issued identifying the occurrence. You must then manually add the application to the plan.
- Automation implemented using OPSTAT, EQQUSIN, or EQQUSINT can be handled manually if the automation partner is unavailable.
- In DRP test exercises, the reporting attribute for remotely controlled workstations not involved in the test should be set to nonreporting.

Implementing your IBM Workload Scheduler for z/OS DRP

The backup requirements and recovery process for each of the defined IBM Workload Scheduler for z/OS DRP conditions are detailed in the following sections. The plan *must* be fully documented; you cannot assume that your most experienced IBM Workload Scheduler for z/OS person is available to conduct the recovery.

If both your primary and secondary data centers use DFSMS, consider using DFSMShsm ABARS for data backup and restore. ABARS simplifies the backup and recovery process. Also, if you define backups for all data sets with a particular high-level data set name, you can ensure that all required data sets are backed up.

Recovery to start-of-day processing

A disaster recovery plan that recovers to start-of-day processing is, perhaps, the easiest backup process to implement. The following text describes the backup schedule required to support the recovery process to start-of-day processing.

Backup requirements

Table 45 defines the backup intervals required to ensure that you can successfully restore IBM Workload Scheduler for z/OS to start-of-day processing. N/A shows that a backup is not applicable for DRP purposes.

Table 45. Backup cycle for start-of-day DRP

ddname	Format	Defines	Backup interval
EQQADDS	VSAM	Applications and JCL variables	Daily
EQQCKPT	PS	Checkpoint data set	N/A
EQQCP1DS	VSAM	Current-plan-1 data set	N/A
EQQCP2DS	VSAM	Current-plan-2 data set	N/A
EQQCXDS	VSAM	Current-plan-extension data set	N/A
EQQDL*nn*	PS	Dual job-tracking-log data set	N/A
EQQEVDS	PS	Event data set for the event-writer task	N/A
EQQEVD*nn*	PS	Event data set for an event-reader task	N/A
EQQHTTP0	PS	Event data set for end-to-end scheduling with z-centric capabilities	N/A
EQQINCWK	PS	JCC incident work file	N/A
EQQJBLIB	PDS	JCL library data set	Minimum weekly, daily if high activity
EQQJCLIB	PDS	JCC message-table data set	Minimum weekly, N/A if same as primary
EQQJS1DS	VSAM	JCL-repository-1 data set	N/A
EQQJS2DS	VSAM	JCL-repository-2 data set	N/A
EQQJTARC	PS	Job-tracking-archive data set	N/A
EQQJT*nn*	PS	Job-tracking-log data set	N/A
EQQLDDS	VSAM	Work data set for long-term-plan batch jobs	See note 1 on page 355
EQQLTBKP	VSAM	Long-term plan backup	See note 1 on page 355
EQQLTDS	VSAM	Long-term plan data set	See note 1 on page 355
EQQMLIB	PDS	Message library	N/A (already in secondary)
EQQMLOG	PS	Message log	N/A
EQQNCPDS	VSAM	New-current-plan data set	N/A
EQQNCXDS	VSAM	New-current-plan extension data set	N/A
EQQOIDS	VSAM	Operator-instruction database	Minimum weekly, daily if high activity

Table 45. Backup cycle for start-of-day DRP (continued)

ddname	Format	Defines	Backup interval
EQQPARM	PDS	Initialization-statement-parameter library Initialization-statement-parameter library	Minimum weekly, daily if high activity
EQQPRLIB	PDS	Automatic-recovery-procedure library	Minimum weekly, daily if high activity
EQQRDDS	VSAM	Special resource descriptions	Daily
EQQSCLIB	PDS	Scripts and Command library	Minimum weekly, daily if high activity
EQQSIDS	VSAM	ETT criteria and configuration data	Minimum weekly, daily if high activity
EQQSTC	PDS	Started-task-submit data set	N/A
EQQSUDS	PS	Submit/release data set	N/A
EQQTWSCS	PDSE	End-to-end data set for centralized script support	N/A
EQQTWSIN	PS	Input events for end-to-end scheduling with fault-tolerance capabilities	N/A
EQQTWSOU	PS	Output events for end-to-end scheduling with fault-tolerance capabilities	N/A
EQQWSDS	VSAM	Workstation, calendar, and period definitions	Minimum weekly, daily if high activity
EQQYPARM	PDS	Initialization statements parameter library	Minimum weekly, daily if high activity
STEPLIB	PDS	IBM Workload Scheduler for z/OS load-module library	N/A (already in secondary)
user-defined	PS	Submit/release data set	N/A

Note:

1. When an LTP or daily-planning batch job runs a copy of the LTP is written to the EQQLTBKP data set. Use EQQLTBKP for your DRP backup to ensure that no updates have occurred before the backup is taken. Perform the DRP backup daily. The contents of the LT, LD and LB data sets must be kept in synch in case a new LTP needs to be re-created from scratch.
2. IBM Workload Scheduler for z/OS issues message EQQN057I to show that a CP backup is complete. You should update the message to include WTO=YES so you can use NetView to trigger DRP data set backups. Message EQQN051I shows why the CP backup occurred. The DRP backups can be triggered when the scheduler issues message EQQN057I following message EQQN051I with reason "DP END". In fact, now, EQQLTBKP, EQQNCPDS and EQQNCXDS are all synchronized with the other scheduler data.

Recovery process

About this task

Follow the steps listed below to recover your IBM Workload Scheduler for z/OS environment to start-of-day processing.

1. Allocate all required data sets. The JCL required should be based on the sample library (SEQQSAMP) members EQQPCS01 and EQQPCS02.

2. Restore data from the last backup taken before the start-of-day required. For example, if recovery to start-of-day for Tuesday is required, then backups from Monday should be used for the recovery.
3. Specify JOBSUBMIT (NO) and CURRPLAN (NEW) on the JTOPTS statement and then start the controller address space. Start additional address spaces required for Tivoli OPC tracker subsystems.
4. If required, use the mass update function in the Application Description dialog to make out-of-effect business systems that you do not need to process in the secondary center.
5. Submit a long-term plan MODIFY ALL batch job.
6. Submit a daily plan TRIAL for the required planning period, and check the results. It is recommended that you start the planning period from 00:00 on the date required to reduce the number of undecided occurrences included in the current plan.
7. Submit a daily plan EXTEND batch job for the required planning period. Use 00:00 as the start time. When the new current plan is created, use the IBM Workload Scheduler for z/OS dialogs to check that occurrences and resources are in the correct status before starting job submission.
8. Change CURRPLAN(NEW) to CURRPLAN(CURRENT) on the JTOPTS statement.

Recovery to a predefined recovery point

About this task

This backup and recovery process can be used when the DRP strategy calls for recovery to a particular time or to a certain point in the processing cycle. This could be once per day or hourly. The process is the same regardless of the regularity with which you need to take backups.

Backup Requirements

Table 46 defines the backup intervals required to ensure you can successfully restore IBM Workload Scheduler for z/OS to a predefined recovery point. Not all data sets must be backed up at the acknowledged point in processing. Those that must should be backed up only after BACKUP commands for both CP and JS resources have been processed by the controller. N/A shows that a backup is not applicable for DRP purposes.

Table 46. Backup cycle for predefined recovery point DRP

ddname	Format	Defines	Backup interval
EQQADDS	VSAM	Applications and JCL variables	Daily
EQQCKPT	PS	Checkpoint data set	N/A
EQQCP1DS	VSAM	Current-plan-1 data set	N/A
EQQCP2DS	VSAM	Current-plan-2 data set	N/A
EQQCXDS	VSAM	Current-plan-extension data set	N/A
EQQDL*nn*	PS	Dual job-tracking-log data set	N/A
EQQEVDS	PS	Event data set for the event-writer task	N/A
EQQEVD*nn*	PS	Event data set for an event-reader task	N/A

Table 46. Backup cycle for predefined recovery point DRP (continued)

ddname	Format	Defines	Backup interval
EQQHTTP0	PS	Event data set for end-to-end scheduling with z-centric capabilities	N/A
EQQINCWK	PS	JCC incident work file	N/A
EQQJBLIB	PDS	JCL library data set	Minimum weekly, daily if high activity
EQQJCLIB	PDS	JCC message-table data set	Minimum weekly, N/A if same as primary
EQQJS1DS	VSAM	JCL-repository-1 data set	Predefined time (see note 1 on page 358)
EQQJS2DS	VSAM	JCL-repository-2 data set	Predefined time (see note 1 on page 358)
EQQJTARC	PS	Job-tracking-archive data set	N/A
EQQJT*nn*	PS	Job-tracking-log data set	N/A
EQQLDDS	VSAM	Work data set for long-term-plan batch jobs	See note 2 on page 358
EQQLTBKP	VSAM	Long-term plan backup	See note 2 on page 358
EQQLTDS	VSAM	Long-term plan data set	See note 2 on page 358
EQQMLIB	PDS	Message library	N/A (already in secondary)
EQQMLOG	PS	Message log	N/A
EQQNCPDS	VSAM	New-current-plan data set	After every NCP
EQQNCXDS	VSAM	New-current-plan extension data set	After every NCP
EQQOIDS	VSAM	Operator-instruction database	Minimum weekly, daily if high activity
EQQPARM	PDS	Initialization-statement-parameter library Initialization-statement-parameter library	Minimum weekly, daily if high activity
EQQPRLIB	PDS	Automatic-recovery-procedure library	Minimum weekly, daily if high activity
EQQRDDS	VSAM	Special resource descriptions	Daily
EQQSCLIB	PDS	Scripts and Commands description library	Minimum weekly, daily if high activity
EQQSIDS	VSAM	ETT criteria and configuration data	Minimum weekly, daily if high activity
EQQSTC	PDS	Started-task-submit data set	N/A
EQQSUDS	PS	Submit/release data set	N/A
EQQTWSCS	PDSE	End-to-end data set for centralized script support	N/A
EQQTWSIN	PS	Input events for end-to-end scheduling with fault tolerance capabilities	N/A

Table 46. Backup cycle for predefined recovery point DRP (continued)

ddname	Format	Defines	Backup interval
EQQTWSOU	PS	Output events for end-to-end scheduling with fault tolerance capabilities	N/A
EQQWSDS	VSAM	Workstation, calendar, and period definitions	Minimum weekly, daily if high activity
STEPLIB	PDS	IBM Workload Scheduler for z/OS load-module library	N/A (already in secondary)
user-defined	PS	Submit/release data set	N/A

Note:

1. EQQJSnDS: when the requested JS backup is complete, take a backup of the inactive file. IBM Workload Scheduler for z/OS issues message EQQN015I to show when the copy is complete and to identify the inactive JS ddname. You should update the message to include WTO=YES to trigger the backup using NetView.
2. When an LTP or daily-planning batch job runs a copy of the LTP is written to the EQQLTBKP data set. Use EQQLTBKP for your DRP backup to ensure that no updates have occurred before the backup is taken. Perform the DRP backup after every NCP. The contents of the LT, LD and LB datasets must be kept in synch in case a new LTP needs to be re-created from scratch.
3. IBM Workload Scheduler for z/OS issues message EQQN057I to show that a CP backup is complete. You should update the message to include WTO=YES so you can use NetView to trigger DRP data set backups. Message EQQN051I shows why the CP backup occurred. The DRP backups can be triggered when the scheduler issues message EQQN057I following message EQQN051I with reason "DP END". In fact, now, EQQLTBKP, EQQNCPDS and EQQNCXDS are all synchronized with the other scheduler data.

Recovery process

About this task

Follow the steps listed below to recover your IBM Workload Scheduler for z/OS environment to a predefined point in processing.

1. Allocate all required data sets. The JCL required should be based on the sample library (SEQQSAMP) members EQQPCS01 and EQQPCS02.
2. Data backed up daily or weekly should be recovered from the most recent backup. LTP and NCP should also be recovered from the most recent backup.
3. Restore data backed up at the predefined point. Copy the backup of JS to both EQQJS1DS and EQQJS2DS.
4. Specify JOBSUBMIT(NO) and CURRPLAN(NEW) on the JTOPTS statement and then start the controller address space. Start additional address spaces required for the Tivoli OPC tracker subsystems.
5. Use the IBM Workload Scheduler for z/OS dialogs to delete or complete occurrences that you do not need to process in the secondary center. Check the status of all occurrences and special resources before starting job submission.
6. Change CURRPLAN(NEW) to CURRPLAN(CURRENT) on the JTOPTS statement.

If you are scheduling end-to-end with fault tolerance capabilities, perform the following manual actions to make sure that the Symphony file is aligned with the rebuilt current plan:

1. From IBM Workload Scheduler for z/OS dialog select the option 3, DAILY PLANNING. The Producing OPC Daily Plans dialog is displayed.
2. Select option 5, SYMPHONY RENEW.

3. Submit the symphony renew batch job to create a Symphony file aligned with the Current Plan.

Recovery to point-of-failure

This backup and recovery process is used to recover the IBM Workload Scheduler for z/OS environment to point-of-failure. The strategy relies on the dual job-tracking facility. The dual logs must be allocated on DASD in the secondary center connected to the primary center by either channel-to-channel connectors or fiber-optic channel extenders.

Although the backup process is complex, when in place, this setup provides quick recovery of the environment.

Backup requirements

About this task

Table 47 defines the backup intervals required to ensure you can successfully restore to point-of-failure. Some data sets need only be backed up weekly; others are required to be backed up after *every* CP or JS backup. N/A shows that a backup is not applicable for DRP purposes.

Table 47. Backup cycle for point-of-failure DRP

ddname	Format	Defines	Backup interval
EQQADDS	VSAM	Applications and JCL variables	Daily
EQQCKPT	PS	Checkpoint data set	N/A
EQQCP1DS	VSAM	Current-plan-1 data set	N/A
EQQCP2DS	VSAM	Current-plan-2 data set	N/A
EQQCXDS	VSAM	Current-plan-extension data set	N/A
EQQDL*nn*	PS	Dual job-tracking-log data set	N/A (already in secondary)
EQQEVDS	PS	Event data set for the event-writer task	N/A
EQQEVD*nn*	PS	Event data set for an event-reader task	N/A
EQQHTTP0	PS	Event data set for end-to-end scheduling with z-centric capabilities	N/A
EQQINCWK	PS	JCC incident work file	N/A
EQQJBLIB	PDS	JCL library data set	Minimum weekly, daily if high activity
EQQJCLIB	PDS	JCC message-table data set	Minimum weekly, N/A if same as primary
EQQJS1DS	VSAM	JCL-repository-1 data set	See note 1 on page 361
EQQJS2DS	VSAM	JCL-repository-2 data set	See note 1 on page 361
EQQJTARC	PS	Job-tracking-archive data set	See note 2 on page 361
EQQJT*nn*	PS	Job-tracking-log data set	N/A
EQQLDDS	VSAM	Work data set for long-term-plan batch jobs	See note 3 on page 361
EQQLTBKP	VSAM	Long-term plan backup	See note 3 on page 361
EQQLTDS	VSAM	Long-term plan data set	See note 3 on page 361

Table 47. Backup cycle for point-of-failure DRP (continued)

ddname	Format	Defines	Backup interval
EQQMLIB	PDS	Message library	N/A (already in secondary)
EQQMLOG	PS	Message log	N/A
EQQNCPDS	VSAM	New-current-plan data set	After every NCP
EQQNCXDS	VSAM	New-current-plan extension data set	After every NCP
EQQOIDS	VSAM	Operator-instruction database	Minimum weekly, daily if high activity
EQQPARM	PDS	Initialization-statement-parameter library Initialization-statement-parameter library	Minimum weekly, daily if high activity
EQQPRLIB	PDS	Automatic-recovery-procedure library	Minimum weekly, daily if high activity
EQQRDDS	VSAM	Special resource descriptions	Daily
EQQSCLIB	PDS	Scripts and Commands definition library	Minimum weekly, daily if high activity
EQQSIDS	VSAM	ETT criteria and configuration data	Minimum weekly, daily if high activity
EQQSTC	PDS	Started-task-submit data set	N/A
EQQSUDS	PS	Submit/release data set	N/A
EQQTWSCS	PDSE	End-to-end data set for centralized script support	N/A
EQQTWSIN	PS	Input events for end-to-end scheduling with fault tolerance capabilities	N/A
EQQTWSOU	PS	Output events for end-to-end scheduling with fault tolerance capabilities	N/A
EQQWSDS	VSAM	Workstation, calendar, and period definitions	Minimum weekly, daily if high activity
STEPLIB	PDS	IBM Workload Scheduler for z/OS load-module library	N/A, already in secondary
user-defined	PS	Submit/release data set	N/A

Table 47. Backup cycle for point-of-failure DRP (continued)

ddname	Format	Defines	Backup interval
Note: 1. EQQJS*n*DS: after every JS copy is complete, make a backup of the inactive file. IBM Workload Scheduler for z/OS issues message EQQN015I to show when the copy is complete and to identify the inactive JS ddname. You should update the message to include WTO=YES to trigger the backup using NetView. 2. EQQJTARC: after every CP backup, the contents of the current job-tracking log are copied to this data set. Take the backup after message EQQN090I is issued to show that the JT data is copied to the archive data set. 3. When an LTP or daily-planning batch job runs a copy of the LTP is written to the EQQLTBKP data set. Use EQQLTBKP for your DRP backup to ensure that no updates have occurred before the backup is taken. Perform the DRP backup after every NCP. The contents of the LT, LD and LB data sets must be kept in synch in case a new LTP needs to be re-created from scratch. 4. IBM Workload Scheduler for z/OS issues message EQQN057I to show that a CP backup is complete. You should update the message to include WTO=YES so you can use NetView to trigger DRP data set backups. Message EQQN051I shows why the CP backup occurred. The DRP backups can be triggered when the scheduler issues message EQQN057I following message EQQN051I with reason "DP END". In fact, now, EQQLTBKP, EQQNCPDS and EQQNCXDS are all synchronized with the other scheduler data.			

Recovery process

About this task

Follow the steps listed below to recover your IBM Workload Scheduler for z/OS environment to the point-of-failure.

1. Allocate all required data sets. The JCL required should be based on the sample library (SEQQSAMP) members EQQPCS01 and EQQPCS02.
2. Data backed up daily or weekly should be recovered from the most recent backup. LTP, NCP, and JTARC should be recovered from the most recent backup.
3. Restore data backed up at regular intervals. Copy the backup of JS to both EQQJS1DS and EQQJS2DS.
4. Browse EQQJTARC and obtain the time stamp of the last record. The time stamp starts at decimal location 12 and is in the format 00YYMMDDFHHMMSSTH. Examine EQQDLnn data sets, and identify the files that contain job-tracking records not included in the archive log.
5. Copy the required EQQDLnn data set to EQQJT01. If more than one dual log contains job-tracking data not included in the archive log, append all the records to EQQJT01 in strict time order.
6. Specify JOBSUBMIT(NO) and CURRPLAN(NEW) on the JTOPTS statement and then start the controller address space. Start additional address spaces required for the Tivoli OPC tracker subsystems.
7. Use the IBM Workload Scheduler for z/OS dialogs to delete or complete occurrences that you do not need to process in the secondary center. Check the status of all occurrences and resources before starting job submission.
8. Change CURRPLAN(NEW) to CURRPLAN(CURRENT) on the JTOPTS statement.

If you are scheduling end-to-end with fault tolerance capabilities, perform the following manual actions to make sure that the Symphony file is aligned with the rebuilt current plan:

1. From IBM Workload Scheduler for z/OS dialog select the option 3, DAILY PLANNING. The Producing OPC Daily Plans dialog is displayed.
2. Select option 5, SYMPHONY RENEW.

3. Submit the symphony renew batch job to create a Symphony file aligned with the Current Plan.

DASD mirroring considerations

If you use products such as PPRC or other OEM products for DASD mirroring of IBM Workload Scheduler for z/OS data sets, consider the following:

- If you need to recover on the mirrored system, allocate to DASD the EQQMLOG for the IBM Workload Scheduler for z/OS tasks, to have data sets mirrored and checked.
- The use of mirroring makes dual logging unnecessary if JT files are mirrored.
- NCP/NCX are mirrored unless the mirroring is done during the run of CP EXTEND or REPLAN, because they make NCP/NCX unusable. In this case, you must use the most recent and complete NCP backup. The message sequence EQQN121 and EQQN115 not followed by message EQQN116, shows that NCP is not yet completed and, as a consequence, the mirrored copy is unusable.
- Note that, even if you use the DASD mirroring in your environment, you must still set CURRPLAN(NEW) when starting the controller after a Disaster Recovery because with LSR buffering CP1/CP2 files cannot be completely mirrored when the disaster occurs. For the same reason, check that you use, at restart, a complete NCP copy either from mirroring or from the most recent backup.

Considerations about testing disaster recovery in an end-to-end environment

During a disaster recovery, the USS BINDIR (either HFS or ZFS) is also to be restored. If you are testing a disaster recovery while the production environment is still running, the IP addresses contained in the Symphony file could cause the fault-tolerant agents to be inadvertently shut down. To prevent this problem, the WRKDIR must not be restored at the disaster recovery site or the Symphony file must be deleted before starting the end-to-end server at the disaster recovery site.

Therefore, if you are testing a disaster recovery, perform either of the following:

- Do not restore the WRKDIR at the disaster recovery site. Instead, run the EQQPCS05 job to create an empty WRKDIR, then start the end-to-end server and run a SYMPHONY RENEW.
- After restoring the production WRKDIR and *before* starting the controller and end-to-end server, delete or rename the `Symphony` and `translator.chk` files from WRKDIR. Start the controller and end-to-end server, then run a SYMPHONY RENEW.

Part 3. Tuning

This part describes how you can tune IBM Workload Scheduler for z/OS.

Chapter 13. Analyzing performance

This part of the book describes how IBM Workload Scheduler for z/OS performance might be improved. It also provides reference information to help you achieve such improvement.

Good performance is the achievement of agreed service levels. This means that system availability, throughput, and response times meet user's expectations using resources available within the budget.

The performance of IBM Workload Scheduler for z/OS should be considered when you:
- Plan to install a new system
- Want to review an existing system
- Contemplate major changes to a system.

There are several basic stages in tuning a system, some of which might be iterative until performance is acceptable. These are:
- Agree what good performance is
- Set up performance objectives
- Decide on measurement criteria
- Measure the performance of the production system
- Make adjustments as required
- Continue to monitor the performance of the system and anticipate future constraints.

Recommendations given in this book, based on current knowledge of IBM Workload Scheduler for z/OS, are general in nature, and cannot be guaranteed to improve performance of any particular system.

Setting performance objectives

Performance objectives often consist of a throughput rate plus a list of dialog or batch functions and expected timings for each. Ideally, through them, good performance can be easily recognized and you know when to stop further tuning. So they must be:
- Practically measurable
- Based on a realistic workload
- Within the budget.

Such objectives might be defined in terms such as:
- Acceptable response times, for example, within which 90% of all responses occur.
- Average or peak number of controlled operations through the system.
- System availability, including mean time to failure, and recovery after a failure.

Measuring performance

After you have defined the workload and estimated the resources required, you must reconcile the response that you want with what you consider attainable. Performance of a production system depends on the usage, paging rates, and virtual storage requirements placed on the main processor, the traffic to and from the disk devices, the traffic of messages throughout the network and a variety of other factors.

You should monitor all of these factors to determine when constraints in the system might develop. A variety of programs could be written to monitor all these resources. Some of these programs are currently supplied as part of IBM products such as IBM Workload Scheduler for z/OS or are supplied as separate products. This topic describes some of the programs that can give performance information on different components of a production system.

The list of products in this topic is far from being an exhaustive summary of performance monitoring tools, yet the data provided from these sources comprises a large amount of information. To monitor all this data is an extensive task. Furthermore, only a small subset of the information provided is important for identifying constraints and determining necessary tuning actions, and you must identify this specific subset. You often have to gather a lot of data before you can fully understand the behavior of your own system and determine where a tuning effort can provide the best overall performance improvement. You must be familiar with the analysis tools and the data they provide to successfully tune a system. But remember that all monitoring tools cost processing effort to use.

Performance Reporter for MVS and Tivoli Decision Support for OS/390

Performance Reporter for MVS (formerly Performance Data Manager for MVS) and Tivoli Decision Support for OS/390 are IBM products that collect and analyze data from IBM Workload Scheduler for z/OS and other IBM systems and products. You can build reports that help you with:
- System overviews
- Service levels
- Availability
- Performance and tuning
- Capacity planning
- Change and problem management
- Accounting.

Many predefined reports are available. You can also generate your own reports to meet specific needs.

The reports use data from the IBM Workload Scheduler for z/OS job-tracking files. Performance Reporter for MVS and Tivoli Decision Support for OS/390 also collect data from the MVS system and from products such as RMF™, TSO, IMS and NetView. This means that data from IBM Workload Scheduler for z/OS and other systems can be shown together, or can be presented in separate reports.

Reports can be presented as plots, bar charts, pie charts, tower charts, histograms, surface charts, and other graphic formats. Performance Reporter for MVS and Tivoli Decision Support for OS/390 simply pass the data and formatting details to Graphic Data Display Manager (GDDM), which does the rest. Performance Reporter for MVS and Tivoli Decision Support for OS/390 can also produce line graphs and histograms using character graphics, where GDDM, is not available or the output device does not support graphics. For some reports, where you need the exact figures, numeric reports such as tables and matrices are more suitable.

For details on Performance Reporter for MVS or on Tivoli Decision Support for OS/390, refer to *Performance Reporter for OS/390, System Performance Feature Reference, Volume 2.*

RMF

The Resource Management Facility (RMF) collects system-wide data that describes the processor activity (WAIT time), I/O activity (channel and device usage), main storage activity (demand and swap paging statistics) and system resources manager (SRM) activity (workload).

RMF Version 4 is a centralized measurement tool that monitors system activity to collect performance and capacity planning data. The analysis of RMF reports provides the basis for tuning the system to user requirements. They can also track resource usage.

RMF Version 4 measures these activities:

- Processor usage
- Address space usage
- Channel activity:
  - Request rate and service time per physical channel
  - Logical-to-physical channel relationships
  - Logical channel queue depths and reasons for queuing.
- Device activity and contention for the following devices:
  - Unit record
  - Graphics
  - Direct access storage
  - Communication equipment
  - Magnetic tapes
  - Character readers.
- Detailed system paging
- Detailed system workload
- Page and swap data set
- Enqueue.

RMF Version 4 lets the z/OS user:

- Evaluate system responsiveness:
  - Identify bottlenecks
  - The detailed paging report associated with the page and swap data set activity can give a good picture of the behavior of a virtual storage environment.
- Check the effects of tuning:
  - Results can be observed dynamically on a screen or by postprocessing facilities.
- Perform capacity planning evaluation:
  - The workload activity reports include the interval service broken down by key elements such as processor, input/output, and main storage service.

  – Analysis of the resource monitor output (for example, system contention indicators, swap-out broken down by category, average ready users per domain) helps in understanding user environments and forecasting trends.
  – The postprocessing capabilities make the analysis of peak load periods and trend analysis easier.
- Manage the larger workloads and increased resources that z/OS can support.
- Identify and measure the usage of online channel paths.
- Optimize the usefulness of expanded storage capability.

RMF measures and reports system activity and, in most cases, uses a sampling technique to collect data. Reporting can be done with one of three monitors:

1. Monitor I measures and reports the use of system resources (that is, the processor, I/O devices, storage, and data sets on which a job can enqueue during its execution). It runs in the background and measures data over a time period. Reports can be printed immediately after the end of the measurement interval, or the data can be stored in SMF records and printed later with the RMF postprocessor. The RMF postprocessor can be used to generate reports for exceptions: conditions where user-specified values are exceeded.
2. Monitor II, like Monitor I, measures and reports the use of system resources. It runs in the background under TSO or on a console. It provides snapshot reports about resource usage, and also allows data to be stored in SMF records. The RMF postprocessor can be used to generate exception reports.
3. Monitor III primarily measures the contention for system resources and the delay of jobs that such contention causes. It collects and reports the data in real time at a display station, with optional printed copy backup of individual displays. Monitor III can also provide exception reports, but its data cannot be stored in SMF records.

RMF should be active in the system 24 hours a day. Run it at a dispatching priority above other address spaces in the system so that:
- The reports are written at the interval requested
- Other work is not delayed because of locks held by RMF.

A report is generated at the time interval specified by the installation. The largest system overhead of RMF occurs during the report generation: the shorter the interval between reports, the larger the burden on the system. An interval of 60 minutes is recommended for normal operation. When you are addressing a specific problem, reduce the time interval to 10 or 15 minutes. The RMF records can be directed to the SMF data sets with the NOREPORT and RECORD options; the report overhead is not incurred, and the SMF records can be formatted later.

For details on RMF Version 4, see *MVS/ESA Resource Measurement Facility™ (RMF) Version 4 Program Summary* and *MVS/ESA Resource Measurement Facility (RMF) General Information*.

ACF/VTAM

ACF/VTAM (program number 5735-RC2) provides information about buffer usage either to GTF in SMF trace data or to the system console through DISPLAY and BFRUSE commands. Other tuning statistics can also be recorded on the system console through the MODIFY procname, TNSTAT command. (This command is described in *ACF/VTAM Diagnostic Techniques*.)

VSAM

VSAM LISTCAT provides information that interprets the actual situation of VSAM data sets. This information includes counts of:

- Whether and how often control interval (CI) or control area (CA) splits occur (splits should occur very rarely).
- Physical accesses to the data set.
- Extents for a data set (secondary allocation). Avoid this secondary allocation, if possible, by making the primary allocation sufficiently large.
- Index levels.

IBM Workload Scheduler for z/OS performance data

You can automatically receive performance information from IBM Workload Scheduler for z/OS using the STATMSG keyword of the JTOPTS initialization statement. Three values are supported for STATMSG:

CPLOCK

The event-manager subtask issues messages EQQE004 and EQQE005, which describe how often different tasks have referenced the current-plan data set. The `SYSZDRK/`*`ssname`*`CP` enqueue is taken by the subtasks that access the current plan. In most cases the enqueue is taken exclusively. Only the general service subtask, which handles requests from host dialog users, PIF, the API, enqueues against the resource shared. A degree of contention on the resource is expected, because the subtasks continually take the lock, process work, and release the lock. If there is resource contention for more than 50% of the time during a measured interval, review the tuning recommendations.

IBM Workload Scheduler for z/OS issues these messages when the number of events that it has processed is greater than approximately half the value of the BACKUP keyword. If you specified BACKUP(NO), IBM Workload Scheduler for z/OS uses the default value of the BACKUP keyword (400) to calculate when to issue messages for CPLOCK. The number of events processed includes all events that IBM Workload Scheduler for z/OS processes, and not just those events for operations in the current plan.

EVENTS

The event-manager subtask issues messages EQQE000I, EQQE006I, and EQQE007I, which describe how many events were processed and provide statistics for the different event types. IBM Workload Scheduler for z/OS issues these messages when the number of events that it has processed is greater than approximately half the value of the BACKUP keyword. If you specified BACKUP(NO), IBM Workload Scheduler for z/OS uses the default value of the BACKUP keyword (400) to calculate when to issue messages for EVENTS. The number of events processed includes all events that IBM Workload Scheduler for z/OS processes, and not just those events for operations in the current plan.

WSATASK

The event manager task issues messages EQQE008I and EQQE009I, which describe statistic information collected by the WSA task.

All these messages are issued according to the following criteria:

- If STATIM has been set to a value different from 0 (by specifying STATIM(*n*) in the JTOPTS keyword, or by using the modify command /F *subsys*,STATIM=*n*), the message is issued approximately every *n* minutes, if any events have been processed.

- Otherwise, if EVELIM has been set to a nonzero value (by specifying EVELIM(*n*) in the JTOPTS keyword, or by using the modify command /F *subsys*,EVELIM=*n*), the message is issues approximately every *n* events.
- Otherwise, the message is issued approximately once every *n* events, where *n* is half the JTOPTS BACKUP keyword value (default BACKUP value is 400).

GENSERV
The general-service subtask issues messages EQQG010 to EQQG011, which describe how often different request types have been processed, and how long the general-service queue has been. IBM Workload Scheduler for z/OS issues these messages every 30 minutes if any requests have been processed.

Performance information for the JCC subtask can be obtained from the SYSOUT-archiving exit (EQQUX005).

Chapter 14. Basic tuning activities

This chapter describes the system resources and provides some basic tuning for IBM Workload Scheduler for z/OS administrators, system programmers, and system engineers to improve the performance of the IBM Workload Scheduler for z/OS environment.

In general, if you have a medium to large production workload controlled by IBM Workload Scheduler for z/OS, more than 5000 computer operations per day, consider the performance requirements for IBM Workload Scheduler for z/OS as you would for any large-systems software application.

System resources

As with any application, resources are used by IBM Workload Scheduler for z/OS to accomplish the required work. This section introduces the key resources and discusses the performance ramifications of each related to an interactive workload in the IBM Workload Scheduler for z/OS product. The key resources are I/O time, processor, and storage.

There is a direct correlation between real storage availability, virtual storage availability, I/O activity, and processor utilization. A shortage of real or virtual storage causes an increase in I/O activity, which in turn causes an increase in processor utilization. So, a change to any one of these areas has some effect on the use of the other resources.

I/O activity

Input/output operations are generally *the* most significant factor in any performance equation. A reduction of I/O, either real or physical, often generates the most significant payback in any tuning exercise.

Many techniques are available to both eliminate I/O and to achieve optimum performance for physical I/O. Some of those techniques are discussed in this book. If you have a very high workload, consider any performance enhancing options, either hardware or software, available to you. However, we recommend you do not use software that alters the physical organization of a data set, because this can have detrimental effects on your IBM Workload Scheduler for z/OS data and availability.

The number of real I/Os can be reduced by removing any unnecessary processing and modifying VSAM cluster definitions.

Performance of physical I/Os can be optimized by utilization of DASD hardware facilities.

The duration of a physical I/O is very dependent on the performance of the I/O subsystem, which is influenced greatly by careful placement of data to reduce channel, path, and control unit contention.

Removal of unnecessary processing

This section describes IBM Workload Scheduler for z/OS functions that can cause unnecessarily high processing, with little or no benefit, when not used in the intended manner.

- Alerts, specifically ALERT(LATEOPER) when the deadlines and estimated durations of the operations are not accurate. This can cause a massive number of I/Os against the current plan and affects most other IBM Workload Scheduler for z/OS functions because the current plan lock is held exclusively for the duration of the scan for late operations. The workstation analyzer (WSA) subtask performs this scan every two minutes. When a large percentage of the plan is late, many operation records must be retrieved.
- Poorly defined security environment. AUTHDEF definitions for subresources which have no rules cause FRACHECKs to be sent across the SAF interface only to be essentially ignored by the security system. Be sure to define only subresource names on the AUTHDEF statement for which resource rules actually exist.
- Too frequent current plan or JCL repository backups. CP backups are probably too frequent if there is more than 1 every 20 minutes during the peak processing period. Most installations do not require more than one JCL repository backup per day. If an optimum value cannot be found for the BACKUP or MAXJSFILE keywords to achieve the backup frequency required, consider defining NO as the value and scheduling the backups using the BACKUP command, EQQEVPGM, EQQUSIN, or the EQQUSINB subroutine.
- Unnecessary scanning of all JCL submitted. If you use the input tailoring facilities provided by IBM Workload Scheduler for z/OS, be sure to use VARSUB(SCAN) rather than VARSUB(YES) on the OPCOPTS initialization statement if performance is important and less than 70% of submitted operations have input tailored by IBM Workload Scheduler for z/OS at submit time.
- Do **not** use the job-completion checker (JCC) function to reset acceptable nonzero return codes. The NOERROR initialization statement is a much more efficient and much safer alternative. Additionally, z/OS provides functions to fail a job with a JCL error at step termination if a *not cataloged x* condition is detected, thereby removing the requirement to use JCC to detect such conditions.
- Audit functions. AUDIT with AMOUNT(DATA) causes a large increase in I/O, and a significant amount of DASD space, on the job-tracking logs. If the actual changed record is not required, then consider auditing with AMOUNT(KEY). Multiple audit statements are supported, thereby allowing you to define some files with AMOUNT(DATA) and others with AMOUNT(KEY).
- Consider GETMAINing and FREEMAINing storage for user exits in EQQUX000, the subsystem start-stop exit, rather than getting and freeing on each call.
- Delay retrieval of job logs until a dialog user has requested to browse a job log for non z/OS trackers.

Chapter 15, “Tuning the controller, the tracker, and the TCP/IP server,” on page 379 contains further recommendations more specific to either the tracker or the controller, many of which also result in the removal of unnecessary processing.

VSAM cluster definitions

The IBM Workload Scheduler for z/OS databases and plans are stored in VSAM KSDS clusters. Default cluster definitions provided with IBM Workload Scheduler for z/OS do not consider the I/O performance to the clusters. While changes to the cluster definitions can provide significant performance improvements, such changes cost either in additional storage usage by the controller address space or by increased DASD space.

Consider these performance recommendations:

- Use the IMBED and REPLICATE options if the clusters are not behind CACHE.

  IMBED specifies that the sequence-set record for each control area is written as many times as it fits on the first track adjacent to the control area. If the allocation is less than a cylinder, one track is added to the primary and secondary allocation quantities.

  REPLICATE specifies that each index record is to be written on a track as many times as it fits. With REPLICATE, rotational delay is reduced and performance is improved. But the cluster's index usually requires more direct access device space.
- Define some free space on clusters that have significant insert activity, particularly the JCL repository (EQQJS1DS and EQQJS2DS), which requires at least FREESPACE(20, 20). Consider a free space allocation for the current plan (EQQCP1DS, EQQCP2DS, EQQCXDS, EQQNCPDS, and EQQSCPDS), extended data (EQQXD1DS, EQQXD2DS, and EQQNXDDS), application descriptions (EQQADDS), resource descriptions (EQQRDDS), and operator instructions (EQQOIDS).

  FREESPACE(*CI-percent*, *CA-percent*) specifies the percentage of each control interval and control area that is to be set aside as free space when the cluster is initially loaded, during a mass insert, and after any split of control intervals (CI-percent) and control areas (CA-percent). Empty space in the control interval and control area is available for data records that are updated and inserted after the cluster is initially loaded. CI-percent translates into a number of bytes that is equal to or slightly less than the percentage value of CI-percent. CA-percent translates into a number of control intervals that is equal to or less than the percentage value of CA-percent.

  CI-percent and CA-percent must be equal to or less than 100. When you specify FREESPACE(100 100), one data record is placed in each control interval used for data and one control interval in each control area is used for data (that is, one data record is stored in each control area when the data set is loaded). When no FREESPACE value is coded, the default specifies that no free space be reserved when the data set is loaded.

  When you define the cluster using the RECORDS parameter, the amount of free space specified is not taken into consideration in the calculations to determine primary allocation.
- Allocate more buffers on clusters that do not have LSR buffering.

  BUFFERSPACE specifies the minimum space to be provided for buffers. The buffer space size you specify helps VSAM determine the data component's and index component's control interval size. If BUFFERSPACE is not coded VSAM attempts to get enough space to contain two data component control intervals and one index component control interval. Try to have one buffer for each index CA, plus one extra.
- If you do not require SPANNED support, do not allocate the cluster with the spanned attribute. By default, EQQADDS, EQQLTDS, and EQQLDDS are spanned clusters.

  SPANNED specifies that if the maximum length of a data record (as specified with RECORDSIZE) is larger than a control interval, the record is contained on more than one control interval. This allows VSAM to select a control interval size that is optimum for the direct access device.

  When a data record that is larger than a control interval is put into a cluster that allows spanned records, the first part of the record completely fills a control interval. Subsequent control intervals are filled until the record is written into the cluster. Unused space in the record's last control interval is not available to contain other data records.

Spanning a cluster causes the number of physical I/Os to be dramatically increased. If you must define a cluster with the spanned attribute and performance is important, consider using CACHE and DASD-FAST-WRITE for these clusters.

- Specify at least `AMP=('BUFND=5,BUFNI=5')` in the IBM Workload Scheduler for z/OS JCL procedure for the controller on the EQQCP*x*DS and EQQJS*x*DS ddnames. CP and JS copies are NSR, they run faster with additional buffers.
- This is true for CP only during CP copy process which uses NSR. For other processing where the CP uses LSR buffering, this does not make a difference. The selection of the optimal values to use for `BUFND` and `BUFNI`, can be progressively tuned by observing the time interval between the following two messages:
  - EQQN056I: A current plan copy process has started.
  - EQQN057I: A current plan data set was successfully copied assuming that the same number of CP records is involved.

Data set placement

Place your key IBM Workload Scheduler for z/OS data sets on volumes that minimize contention for the volumes on their controllers.

Carefully consider the placement of these performance critical data sets:
- Current plan data sets (EQQCP1DS, EQQCP2DS, and EQQCXDS)
- Current plan side-information (EQQSIDS)
- Event data sets (EQQEVDS)
- Submit/release data sets (EQQSUDS)
- JCL repository (EQQJS1DS and EQQJS2DS)
- Job-tracking logs (EQQJT*nn* and EQQJTARC)
- JCL variable tables when job input tailoring is used (EQQADDS).

Hardware performance options

Use CACHE and DASD-FAST-WRITE if you can for the files recommended below. While these facilities do not reduce the number of EXCPs, they can sharply reduce the time it takes to complete them. CACHE and DASD-FAST-WRITE can be important ways to reduce enqueue contention for two reasons. First, DASD-FAST-WRITE can reduce the write times. VSAM buffering, of course, only reduces the read times, and that can make write-time reductions even more important. Second, both CACHE and DASD-FAST-WRITE are effective on spanned records.

CACHE

Appropriate for all files except the job-tracking logs. The best candidates are:

- The long-term plan (EQQLTDS)
- Operator instructions (EQQOIDS)
- Application descriptions and JCL variable tables (EQQADDS)
- The JCL libraries (EQQJBLIB)
- Event data sets (EQQEVDS), but only when an event reader task has been started using the ERDRTASK keyword on the OPCOPTS initialization statement.

DASD-FAST-WRITE

Suitable for all files. The best candidates are:

- The job-tracking logs (EQQJT*nn*)
- Submit/release data sets (EQQSUDS)
- The JCL repository (EQQJS1DS and EQQJS2DS)
- Event data sets (EQQEVDS)
- Current plan data sets (EQQCP1DS and EQQCP2DS).

If you submit a very high number of computer operations every day (greater than 50 000) and you have solid-state devices, consider placing the JCL repository (EQQJS1DS and EQQJS2DS) and any submit/release data sets (EQQSUDS) on such devices.

Processor

Normally, the amount of processor power is not the most significant factor in IBM Workload Scheduler for z/OS performance.

Of course, no matter what the power of the processor, if most of the processor is normally busy, then the processor can be the most critical resource for any application. In such a case, you can help IBM Workload Scheduler for z/OS get access to the processor resource by giving it higher dispatching priority. This might not be a serious impact on lower priority users, because in most cases, IBM Workload Scheduler for z/OS needs the processor frequently, but does not need it very much. The IBM Workload Scheduler for z/OS subsystems should have a priority just below that of the JES subsystem.

Processor storage

Some amount of paging and swapping is normal for systems that serve a lot of users, or have a significant amount of data in virtual storage. The impact on your system performance from this activity is not normally severe. The few more I/O operations per transaction for paging and swapping are not usually a significant increase on the I/O that would be required without paging and swapping.

If performance is important, define the IBM Workload Scheduler for z/OS address spaces as non-swappable.

The virtual storage of a processor may far exceed the size of central storage available in the configuration. Any excess must be maintained in auxiliary storage (DASD), or in expanded storage. This virtual storage occurs in blocks of addresses called *pages*. Only the most recently referenced pages of virtual storage are assigned to occupy blocks of physical central storage. When reference is made to a page of virtual storage that is not currently in central storage, the page is brought in from DASD or expanded storage to replace a page in central storage that is not in use and is least recently used. The newly referenced page is said to have been *paged in*. The displaced page may need to be *paged out* if it has been changed.

IBM Workload Scheduler for z/OS uses a significant amount of virtual storage, both in the controller address space and by the batch programs that create the current plans.

The local shared resources (LSR) buffering technique is used by IBM Workload Scheduler for z/OS for the current plan data set; a percentage of the current plan determined by the CPDTLIM keyword on the OPCOPTS statement is kept in LSR buffers above the 16-megabyte line. These LSR buffers are deleted and rebuilt at every current plan backup to ensure that the percentage of the plan that you want is always in storage.

Two data spaces, one for calendars and one for special resources, are also created and maintained by the controller address space. The size of calendar data space is dependent on the number of calendars defined. The special resource data space contains all special resource referenced in the current plan. The data spaces are automatically extended by IBM Workload Scheduler for z/OS to accommodate additional data.

Paging problems

The page-in rate is of primary concern, because page-in activity occurs synchronously (that is, a z/OS task stops until the page fault is resolved). Page-out activity overlaps with IBM Workload Scheduler for z/OS processing, so it does not appreciably affect throughput.

A page-in from expanded storage incurs only a small processor usage cost but a page-in from DASD incurs a time cost for the physical I/O and a more significant increase in processor usage.

Thus, extra DASD page-in activity slows down the rate at which transactions are processed by the controller. If you suspect that a performance problem is related to excessive paging, you can use RMF to obtain the paging rates.

Common Storage Area allocated for user connections

IBM Workload Scheduler for z/OS allocates memory blocks sized to 64KB in the common storage area (CSA) to support the communication between the engine and the remote user interfaces (ISPF, Dynamic Workload Console, and PIF). Every active connection is by default allocated a 64KB memory block.

This provides good performance when running sizeable queries on the plan, but if you are worried about the memory consumption (for example, 300 concurrent users will consume 19MB of CSA), you can reduce to 32KB the size of the memory blocks. To do this, specify NO for the LARGEUSERBUFFER keyword of the JTOPTS initialization statement.

Indicators for performance-related problems

These external indicators can identify performance-related problems:

- Long queue of computer operations in R status with no extended status and no reason stated on panel EQQSOPSP.
- Significant delay between when a job ends and the time IBM Workload Scheduler for z/OS reports success or failure for the operation. This delay can be measured as the difference between the event-creation time in the event record and the time of the associated job-tracking-log record. The IBM Workload Scheduler for z/OS audit program lists creation time and controller processing time for events. Additionally, both Performance Reporter for z/OS and Tivoli Decision support for OS/390 include tables to calculate the event processing delay.

See the recommendations in the next chapter to address more specific performance-related problems in job submission, event processing, and event communication.

Preventing bottlenecks

You can avert performance-related problems by regular examination of information that can identify potential problems. Consider performing these tasks as regular intervals:

- Examination of the IBM Workload Scheduler for z/OS statistics generated when the STATMSG keyword is defined in the OPCOPTS initialization statement.
- Regular reorganizations of the VSAM clusters: EQQADDS, EQQRDDS, EQQSIDS, and EQQOIDS.
- Monitor IDCAMS LISTCAT output of the VSAM clusters to ensure optimum allocation. Increase the free space allocation if control interval (CI) and control area (CA) splits are frequently occurring. Reallocate the cluster with a large primary allocation if it has extents.
- Review the message log (EQQMLOG) to identify how often and when the CP and JS backups are occurring.
- Educate the user community on how they can avoid long-running dialog requests:
  - Encourage the use of FASTPATH options where available.
  - Use generic search characters only when required. Specify as much information as possible to limit the search.
  - Familiarize users with the key structure of the VSAM clusters. Try to build dialog selection criteria with consideration of the key to prevent sequential searches of the clusters.

Chapter 15. Tuning the controller, the tracker, and the TCP/IP server

This chapter helps you identify appropriate tuning activities to address general and specific performance problems related to the controller, the tracker, and the TCP/IP server.

Ensure you review the recommendations in Chapter 14, “Basic tuning activities,” on page 371 before implementing any recommendations in this chapter.

Tuning the controller

Performance problems in the controller are most often associated with contention for the current plan enqueue. The amount of time the lock is held by the subtasks is directly related to the amount of work the subtasks must perform. IBM Workload Scheduler for z/OS generates messages documenting the usage of the current plan lock when STATMSG(CPLOCK) is defined on the JTOPTS initialization statement.

The subtasks using the current plan lock are:

- The workstation analyzer (WSA) subtask to start operations at computer, WTO, and nonreporting workstations. The WSA also generates alerts when LATEOPER or DURATION is defined on the ALERTS initialization statement.
- The event manager (EM) subtask processes the events sent from the trackers and updates the current plan as appropriate. The event manager also acts on behalf of the restart and cleanup (RC), automatic recovery (AJR), and event-triggered tracking (ETT) functions.
- The general service (GS) subtasks provide the interface between the controller data and the user. It provides service to the ISPF and OS/2 dialog users, API transaction programs, and the program interface. GS is the only current plan lock user who can take the lock shared, for true read-only transactions.
- The normal-mode manager (NMM) subtask is responsible for integrity of the controller data. NMM enqueues the CP lock when current plan or JCL repository backups are required, or when a new current plan (NCP) has been created.

Job submission

This section helps you to understand the job submission process and identifies tuning activities.

Recognizing the indicators

These items can identify a bottleneck in job submission:

- STATMSG(CPLOCK) identifies a long HOLD time for the WSA.
- STATMSG(CPLOCK) identifies a long WAIT time for the EMGR, NMM, and GS when compared to the WSA.
- Long queue of operations in R status with no extended status and no reason documented on panel EQQSOPSP.

Breaking down the process

To submit a job, IBM Workload Scheduler for z/OS must:

- Identify the best candidate. After the current plan lock is obtained, the ready operations are sorted according to their relative priority. The value of the QUEUELEN keyword of JTOPTS identifies the maximum number of ready operations that will be started by the WSA each time it enqueues on the current plan lock.
- Retrieve JCL:
  - Influenced by the number of partitioned data sets (PDS) and the number of members in the data sets concatenated on the EQQJBLIB ddname. If the partitioned data sets are placed after a CACHE statement, the directory search is quicker.
  - Size of the EQQJBLIB members.
  - The performance of the user exits EQQUX001, EQQUX002, EQQUX009, and EQQUX013.
- Substitute and tailor JCL.
- Image the job input to the JCL repository (JS) file. The duration of this step is dependent on VSAM performance on the JCL repository.
- Submit to the internal reader. This function is performed by the submit subtask, which does not hold the current plan lock. So performance on the internal reader cannot effect other IBM Workload Scheduler for z/OS subtasks. But it can ultimately affect the throughput of work on the processor.

Recommendations

Consider these recommendations:

- Use the EQQUX002 exit to locate the JCL in cases where there are many libraries concatenated on EQQJBLIB and the target library is predictable, perhaps according to the workstation name, or jobname, for example.
- Concatenate only libraries of interest to IBM Workload Scheduler for z/OS on EQQJBLIB. Ensure the libraries are concatenated in frequency order. That is, if more than 50% of the JCL is stored in one library, that library should be the first concatenated on EQQJBLIB. If facilities are available, keep the directories for these libraries in storage.
- Defining `FREESPACE` on the JCL repository (EQQJS1DS and EQQJS2DS) is essential.
- Examine the performance of EQQUX001, EQQUX002, and EQQUX013, if used. The current plan lock is held while the three exits are called, performance is critical.
- Use VARSUB(SCAN) instead of VARSUB(YES) when JCL tailoring is required.
- If there are often many ready operations to be started, consider setting a higher QUEUELEN value. Once the WSA has the lock, submissions are handled very quickly. A well-tuned system can submit tens of operations per second.
- Examine JES performance, particularly on checkpoint data set, because this greatly affects the internal-reader submit time.

Job tracking

This section helps you to understand the factors that affect the performance of the event manager and identifies actions to address those factors. The event manager subtask is most often a victim of poor performance by other users of the current plan lock, it is rarely the cause.

Recognizing the indicators

These indicators can highlight performance problems in the event manager subtask:

- STATMSG(CPLOCK) identifies a long HOLD time for the EMGR.
- STATMSG(CPLOCK) identifies a long WAIT time for the WSA, NMM, or GS when compared to EM.
- Long delay from event creation to event processed. Compare the creation time in the event record with the job tracking log record time. Performance Reporter for MVS, Tivoli Decision Support for OS/390 and EPDM provide tables to report event delay; the IBM Workload Scheduler for z/OS audit program lists creation time and controller processing time for events.
- The total events received by the event manager compared to those that are actually of interest to IBM Workload Scheduler for z/OS.
- Unusually high number of suspended events, identified by 2 in column 53 of the job tracking record. You can also identify suspended events by locating SUSPENDED in the report produced by the IBM Workload Scheduler for z/OS audit package.
- User exits involved in tracking, EQQUX007 and in the tracker EQQUX004, EQQUX005, and EQQUX006.

The connection method and the tracker's performance are also indicators.

Recommendations

Consider these recommendations:

- Reduce the number of suspended events by lowering ERWAIT time.
- Tune the trackers.
- Eliminate as many trivial events as possible:
  - Use STEPEVENTS(ABEND) or STEPEVENTS(NZERO) rather than STEPEVENTS(ALL).
  - Specify PRINTEVENTS(NO) if you do not track print operations.
  - Filter the test workload using EQQUX004.
  - Filter type 5 events, except those with EXROPCAN, if printing is not of interest.
- Use NCF or XCF connections rather than starting event readers. When you use NCF or XCF for communications, be sure to use the EWSEQNO option in the EWTROPTS initialization statement. Starting a specific event reader task in the tracker is not required and dramatically increases I/O to the event data set and more importantly the path length for an event to reach the controller.
- Ensure EQQUX007 is performing well, if used. The current plan lock is held when the exit is taken.

Dialog response

This section helps you to understand the factors that affect the performance of the general service subtask and identifies actions to address those factors.

Factors that influence response times

Consider these factors:

- The number of operations in the plan and the size of the networks. The current plan consists of networks of occurrences. Occurrences with dependent operations are inserted on the same network. In many installations, the size of the largest network can be as much as 80% of the entire plan. When a modify request is processed by the general service task, for example, to add an occurrence with dependencies, IBM Workload Scheduler for z/OS must ensure that the change does not cause a loop in the network.

- Specific request types that initiate network scans. For example, modifying a dependency in the MCP dialog.
- Poor construction of list requests resulting in sequential searches of the current plan data set.
- Time taken to search the ISPxLIB concatenation for panels, messages, load modules.

Recommendations

Consider these recommendations:

- Use GSTASK(5), the maximum value to increase parallelism for dialog requests.
- Use FASTPATH=Y for job name table searches on panels 6.3 and 5.3.
- Choose to traverse the network for dependency loops only when the change is stored, rather than on the panel where you define the dependency and also when the change is stored, When external dependencies are modified in the current plan. Use DEP CHECK=N in the MCP dependency panels to remove the network scan on that panel.
- Consider using LIBDEFs for the ISPF allocations if there are many libraries already concatenated on ISPxLIB.
- Consider moving EQQMINOM, EQQXDSPX and EQQXTBLX into an LPA library. These modules are loaded by dialog users every time they enter an option from the IBM Workload Scheduler for z/OS main menu.
- Determine when to use and when not to use generics, avoid using generic characters in the first position of a field. Study the key structure of the current plan records.
- Eliminate much of the active monitoring of operations in the current plan by using the automatic alert functions provided by IBM Workload Scheduler for z/OS.

Background batch processing

This section helps you to understand the factors that affect the performance of the background batch processing and identifies actions to address those factors.

Recognizing the indicators

As with any batch processing, poor performance is indicated by:

- High I/O rate and relatively low CPU time
- Excessive elapsed time
- High demand paging rates, particularly during daily planning.

Recommendations

Consider these recommendations:

- Use half-track blocking as much as possible
- Add VSAM buffers through JCL AMP statements for clusters that are not processed using the LSR buffering technique.
- Consider using Batch LSR on the EQQADDS, EQQWSDS, and EQQRDDS data sets for long-term and daily planning batch processing. If you use Batch LSR on those data sets, it is not required that you specify the value 1, 2, or 3 for the SHRPOOL identifier.
- Perform general z/OS tuning: examine swap rates, dispatching priorities, the UIC, and especially demand paging rates. Reduce demand paging if possible, or consider moving the daily planning batch jobs to a processor with more real storage, in the same GRS ring as the controller, if one is available.

Tuning the tracker

This section helps you identify appropriate tuning activities to address general and specific performance problems related to the Tivoli OPC tracker.

Event creation and communication

The delays are rarely obvious in the tracker address space. Instead problems are often seen to be with the Event Manager. These items can indicate performance problems:

- Message EQQZ035 while the address space is started. This indicates that events have been lost because the event writer queue in ECSA became filled with unprocessed events.
- Significant gap between the creation time recorded in the event record and the job-tracking record create time for the same event.
- Long JCC queue, normally evidenced from SDSF.

Factors influencing performance

Performance in the tracker is influenced by:

- The number of events generated on the system.
- The logical record length of the event data set (EQQEVDS) when the restart and cleanup function is used.
- The size of the job logs when JCC or the restart and cleanup function is used.
- A specific event reader subtask started using ERDRTASK(1) rather than EWSEQNO (Event Data Set Sequence Number) in the EWTROPTS. When a specific event reader subtask is started, events are written to the event data set by the event writer subtask and then immediately read back by the event reader task. If the connection method is XCF, NCF, or the controller and tracker are started in the same address space, use EWSEQNO (Event Data Set Sequence Number) instead. When you use EWSEQNO (Event Data Set Sequence Number) the events are queued to the subtask responsible for communication with the event manager at the same time as they are written to the event data set.
- The value defined by ERWAIT, when event reader subtasks are used, identifies the time the task waits before rechecking the event data set if no new events were found at the last attempt.
- Performance in the user exits EQQUX004, EQQUX005, EQQUX006, EQQUX008, and EQQUX010.
- JES performance when the JCC is started or the restart and cleanup function is used to retrieve job logs.

Recommendations

Consider these recommendations:

- Do **not** use STEPEVENTS(ALL) unless you use the auto recovery function and are interested if a step was flushed.
- Do **not** specify PRINTEVENTS(YES) unless you are interested in tracking print operations, or disable exit IEFU83 for print events if you are using JES2.
- Filter the testing workload using EQQUX004.
- Consider filtering type 5 events, except those with EXROPCAN on if you do not specify PRTCOMPLETE(YES).
- Ensure there is sufficient ECSA defined for the event writer queue to handle peak processing, and the occasional hardware reserve.

- Do **not** HSM manage the event data set, do **not** place the event data set on a volume where you take full-volume backups.
- See *Planning and Installation* for recommendations on calculating the logical record length of the event data set when the restart and cleanup function is used.
- Examine JCC and restart and cleanup tuning recommendations.

JCC

When the JCC function is used, the success or failure of an operation cannot be determined by the controller until the JCC task has processed the output on the JES spool. The JCC checks every line of SYSOUT against tables defining conditions to be detected.

Measuring JCC performance

You can measure JCC performance by:

- Using the EQQUX005 exit. Specific call indicators tell you when the exit has been called to start process a new job and when there is no more output to check for a job. Good performance in your exit is vital because it is called for *every* SYSOUT line unless you tell IBM Workload Scheduler for z/OS to stop calling it.
- Observe the output queue using SDSF. The job is highlighted on the queue while it is being checked.

Performance in the JCC is influenced primarily by JES performance and also by the size of the job logs.

Recommendations

Consider these recommendations:

- If you currently route all output for JCC processing to one system in your configuration, consider checking the output on the system where the job ran to balance the JCC workload among the available processors. The JCC task cannot process multiple jobs in parallel.
- Remove unnecessary JCC processing:
  - Detection of nonzero return codes, use NOERROR instead.
  - Trapping NOT CATLG x, use z/OS to fail the job on step end.
  - SKIP sections of the output where you cannot possibly match a condition defined in the messages tables.
  - Avoid scanning user SYSOUT data sets if possible.

Tuning the TCP/IP server

The TCP/IP server handles the communication with the remote user interfaces: Dynamic Workload Console, ISPF, and PIF. The process includes a main thread, a listener, which starts a new thread every time it receives a request for connection from one of the interfaces. Every user connected through the Dynamic Workload Console or ISPF, or every running PIF, spends a TCP/IP server thread.

On the z/OS system side, you can control how many connection threads can be open at any given time by appropriately setting the following parameters in the BPXPRMxx member of SYS1.PARMLIB:

MAXTHREADS
Specifies the maximum number of `pthread_`created threads, including running, queued, and exited but undetached, that a single process can have concurrently active.

Its value can range from 0 to 100000. Specifying a value of 0 prevents applications from using `pthread_create`. The default is 200.

MAXPROCUSER
Specifies the maximum number of processes that a single OpenEdition user ID can have concurrently active, regardless of how the processes were created.

Its value can range from 3 to 32767. The default is 25.

If you get the following error:

```
CEE5101C During initialization, the callable service BPX1MSS failed.
The system reason code was 0B250012. The application will be terminated.
```

increase the number of MAXPROCUSER.

To determine the current value set for MAXTHREADS and MAXPROCUSER, examine the appropriate BPXPRMxx member of SYS1.PARMLIB, or display the value by using the DISPLAY OMVS,OPTIONS command.

To change the value of MAXTHREADS dynamically, use the SETOMVS or SET OMVS command. To make a permanent change, edit the BPXPRMxx member that is used for IPLs.

Additional information on the BPXPRMxx parmlib member can be found in the publication *z/OS Initialization and Tuning Reference* (SA22-7592).

Restart and cleanup

When the restart and cleanup function is used, IBM Workload Scheduler for z/OS performances is reduced if there is a large number of requests for user SYSOUT archiving and retrieval.

Part 4. Appendixes

Appendix A. IBM Workload Scheduler for z/OS macros

The macros identified in this appendix are provided by IBM Workload Scheduler for z/OS as programming interfaces for customers.

Attention: Do not use IBM Workload Scheduler for z/OS macros as programming interfaces other than those identified in this appendix.

IBM Workload Scheduler for z/OS provides these macros that are programming interfaces:

EQQCASEC
: The case-code-list definition macro creates case-code lists in the case-code-definition module (EQQCASEM). This module is used by the automatic-recovery function. For more information, see "Creating case-code-definition modules" on page 323.

EQQJCCT
: The JCC message-table macro creates message-table definitions for the job completion checker. "Defining message tables using EQQJCCT" on page 301 describes the macro.

Appendix B. Sample library (SEQQSAMP)

The SEQQSAMP library contains samples to help you install, migrate, and customize IBM Workload Scheduler for z/OS. In most cases, you need only add installation-specific JCL to adapt a member in SEQQSAMP to your requirements.Table 48 shows the members of the sample library that you can use to customize or tune IBM Workload Scheduler for z/OS. The pages that follow describe these members in more detail. For a list of all the members in the SEQQSAMP library, refer to the *Planning and Installation*.

If you need to change a sample member, copy the source to a separate library. The original sample member is then available for reference. Also create an SMP/E *usermod* for each sample member you execute in the production environment. Changes to the sample source code are then flagged for your attention, and subsequent updates can be reflected in the production code as soon as possible.

Table 48. SEQQSAMP library members for customizing and tuning IBM Workload Scheduler for z/OS.

Member	**Brief description**
EQQAIXST	Parameters used by the EQQX9AIX and EQQAIXTR samples.
EQQAIXTR	Sample tracker running on AIX, used with EQQX9AIX.
EQQAUDIB	Sample to invoke EQQAUDIT in batch mode outside of the dialog. **Note:** EQQAUDIB can be used successfully only if the EQQTROUT dsname and the EQQAUDIT output dsn fields in the EQQJOBSA panel are filled out.
EQQBENCO	Sample JCL that encrypts the password defined in the OSLCOPTS initialization statement used to configure IBM Workload Scheduler for z/OS to integrate with OSLC.
EQQBENCR	Sample EQQE2EPW JCL to run the utility that encrypts the Windows passwords defined with the USRPSW parameter of the USRREC statement.
EQQCHKEV	Sample JCL to display EQQTWSIN and EQQTWSOU event data set content information.
EQQCLEAN	Sample procedure invoking EQQCLEAN program.
EQQCONOP	Sample initial parameters for a controller and tracker in the same address space.
EQQCONO	Sample started task procedure for controller only.
EQQCONP	Sample initial parameters for a controller.
EQQCON	Sample started task procedure for a controller and tracker in the same address space.
EQQCVM	Sample to enable job-tracking facilities on VM systems.
EQQCVM2	Sample to enable submission and tracking on VM systems using EQQUX009.
EQQDBENC	Contains the JCL to encrypt the password in the DBOPT statement
EQQDBOPT	Sample DBOPT statement
EQQDELDI	Sample JCL to run the EQQDELDS program that deletes data sets based on the disposition specified in the JCL and the current status in the catalog.
EQQDLFX	Assembler installation sample of DLF connect/disconnect exit.
EQQDPX01	DP batch sample user exit to update Scheduling Environment.
EQQDSCL	Batch Cleanup sample.
EQQDSCLP	Batch Cleanup sample parameters.
EQQDSEX	Batch Export sample.
EQQDSEXP	Batch Export sample parameters.

Table 48. SEQQSAMP library members for customizing and tuning IBM Workload Scheduler for z/OS. (continued)

Member	**Brief description**
EQQDSIM	Batch Import sample.
EQQDSIMP	Batch Import sample parameters.
EQQDSRG	Batch sample rreorg.
EQQDSRI	Batch Recovery index.
EQQDSRIP	Batch Recovery index parameters.
EQQDSTP	Parameters for sample procedure to start data store.
EQQDST	Sample procedure to start data store.
EQQE2EP	Sample initial parameters for server and batch to define when the end-to-end scheduling with fault tolerance capabilities is active.
EQQFLWAT	Sample JCL to call **filewatch** utility to monitor HFS or ZFS file changes
EQQICNVH	Sample job to migrate history DB2 tables.
EQQJCCTB	JCL to assemble a JCC message table macro definition.
EQQJCLIN	Sample JCL to start program EQQPDLF.
EQQJVXIT	Sample assembler JCL-variable-substitution exit. Also used for variables in System Automation commands.
EQQMTWSO	Sample JCL to migrate the EQQTWSOU data set record length from 120 to 160 bytes.
EQQNCFCT	Sample parameters for a SNA connection between controller and tracker.
EQQNETW1	REXX EXEC that receives IBM Workload Scheduler for z/OS WTO messages and issues MVS commands.
EQQNETW2	PL/I NetView command processor that uses EQQUSINT to change the status of operations.
EQQNETW3	REXX EXEC that uses EQQEVPGM to change the status of operations.
EQQOS2ST	Parameters used by the EQQX9OS2 and EQQOS2TR samples.
EQQOS2TR	Sample tracker running on OS/2, used with EQQX9OS2.
EQQPCS04	Allocates VSAM for data store
EQQPCS05	Allocates the work directory (WRKDIR) used by a server for the end-to-end scheduling with fault tolerance capabilities.
EQQPCS06	AllocatesIBM Workload Scheduler for z/OS and IBM Workload Scheduler Integration.
EQQPCS07	Allocates Restart and Cleanup VSAM data sets.
EQQPCS08	Allocates USS files for Java utilities enablement
EQQPCS09	Allocates the GDG root and VSAM data set used as input by the archiving process supporting the Dynamic Workload Console reporting feature
EQQPCS10	Creates the SSL work directory used for TCP/IP communication with the controller
EQQPCS11	Allocates data sets (EQQOUCEV and EQQOUCKP) used for the retrieval of job logs in the z-centric environment with the Output collector.
EQQPCS12	Allocates the GDG root to archive the MLOG files.
EQQPCS13	Allocates the GDG root required for the recovery of a system failure between two remote sites through a remote hot standby controller (also known as a backup controller).
EQQPCS14	Allocates the EQQDB*nn* data sets, used to log data for the extended auditing feature.
EQQREPRO	Is invoked by EQQSMLOG to copy the contents of the outgoing MLOG file onto the GDG data set. You must copy this sample to the PARMLIB of the controller.
EQQPIFJX	Sample to maintain JCL repository.
EQQPROC	Sample procedure, started by IBM Workload Scheduler for z/OS, to initiate purge of DLF objects.

Table 48. SEQQSAMP library members for customizing and tuning IBM Workload Scheduler for z/OS. (continued)

Member	Brief description
EQQRETWT	Sample wait and terminate abends or RC program.
EQQSERP	Sample initial parameters for a server.
EQQSER	Sample started task procedure for a server.
EQQSLCHK	Sample to perform a syntactic check on SCRIPT library members.
EQQSMLOG	Sample procedure that creates the GDG data set where the outgoing MLOG file is archived when the MLOG switching function takes effect. Invokes the EQQREPRO procedure.
EQQTCPCT	Sample definitions for TCP/IP communication between tracker and controller.
EQQTRAP	Sample initial parameters for a tracker.
EQQTRA	Sample started task procedure for a tracker.
EQQTROPT	Sample TRGOPT statement
EQQXML01	Sample XML file for data set triggering event rule definitions
EQQUSIN1	EQQUSIN subroutine sample to change the status of an operation.
EQQUSIN2	EQQUSIN subroutine sample to change the availability of a special resource.
EQQUSIN3	EQQUSIN subroutine sample to change the status of a workstation.
EQQUSIN4	EQQUSIN subroutine sample to back up an IBM Workload Scheduler for z/OS resource data set.
EQQUSIN5	EQQUSIN subroutine sample to update the USERDATA field of an operation.
EQQUX001	Sample job-submit exit.
EQQUX002	Sample job-library-read exit.
EQQUX003	Sample application-description-feedback exit.
EQQUX004	Sample event-filtering exit.
EQQUX011	Sample job-tracking log write exit.
EQQUX013	Sample job-tailoring prevention exit.
EQQUX014	Sample Time Dependent operation exit
EQQUX0N	Sample PL/I start/stop exit, EQQUX000.
EQQUX9N	Sample PL/I operation-initiation exit, communicating with VM (EQQUX009).
EQQUXCAT	EQQDELDS/EQQCLEAN catalog exit sample. You can use it to prevent the deletion of specific data sets.
EQQUXGDG	EQQCLEAN exit sample preventing the running of any single GDG overwrite action into JES control blocks.
EQQUXPIF	Application-description-validation sample.
EQQVTAMN	Sample VTAM/NCF controller/tracker.
EQQVTAMS	Sample VTAM/APPC definition for one server of IBM Workload Scheduler for z/OS.
EQQX5ASM	Sample SYSOUT archiving exit.
EQQX6ASM	Sample incident-record-create exit.
EQQX6JOB	Sample batch-job skeleton JCL used by EQQX6ASM.
EQQX7ASM	Sample change-of-status exit.
EQQX7JOB	Sample batch-job skeleton JCL used by EQQX7ASM.
EQQX9AIX	Sample assembler operation-initiation exit, communicating with AIX.
EQQX9OS2	Sample assembler operation-initiation exit, communicating with OS/2.
EQQXCFCT	Sample definitions for XCF connection between tracker and controller.

EQQUSIN samples

SEQQSAMP contains samples that show you how to use the general subroutine, EQQUSIN. You can use EQQUSIN instead the individual subroutines EQQUSINB, EQQUSINO, EQQUSINS, EQQUSINT, and EQQUSINW. It provides additional functions that are not available in the individual subroutines. You pass parameters to EQQUSIN in an APP buffer, which has the same format as the buffers used by the IBM Workload Scheduler for z/OS application programming interface (API). But you need not invoke APPC services to use EQQUSIN.

These EQQUSIN samples are provided:

EQQUSIN1
Is a sample to change the status of an operation in the current plan. It is equivalent to using EQQUSINT.

EQQUSIN2
Is a sample to change the availability of a special resource. It is equivalent to using EQQUSINS. But EQQUSIN also lets you specify quantity, deviation, and create values, which are not available through EQQUSINS.

EQQUSIN3
Is a sample to change the status of a workstation. It is equivalent to using EQQUSINW.

EQQUSIN4
Is a sample to perform a backup of an IBM Workload Scheduler for z/OS resource data set. It is equivalent to using EQQUSINB.

EQQUSIN5
Is a sample to update the USERDATA field of a current plan operation. It is equivalent to using EQQUSINO.

IBM Workload Scheduler for z/OS exits

The sample exits demonstrate practical implementations but you might need to update them to suit your own requirements. You can use the samples provided as a base for your exits.

When IBM Workload Scheduler for z/OS is started it tries, by default, to load the exits with prefix EQQUX0. You can change the default values by specifying parameters on the EXITS initialization statement. See “EXITS” on page 57 for more information about the EXITS statement.

Start or stop exit

The stop or start exit is called by IBM Workload Scheduler for z/OS when the subsystem, either tracker or controller, is started and also during a normal shut down. The exit is commonly used to allocate resources needed by other IBM Workload Scheduler for z/OS exits.

The SEQQSAMP member EQQUX0N contains a sample start/stop exit written in PL/I. This sample calls the EQQUSINW subroutine to vary the status of a user-defined workstation to `ACTIVE`. The sample has been designed specifically to be used with the EQQCVM2 and EQQUX9N samples which provide a tracker for a VM operating system, but can be used to set the workstation status of any user-defined workstation. See “Tracker for VM” on page 399 for more information about the VM tracker.

Job-submit exit

The job-submit exit is called when IBM Workload Scheduler for z/OS is about to submit a batch job or start a started-task. A common use for this exit is to assign a submitting user ID. If you do not alter the submitting user ID, all jobs submitted by IBM Workload Scheduler for z/OS are, by default, performed under the user ID of the IBM Workload Scheduler for z/OS address space that performs the submission.

Security is often performed through the RUSER field, but you can also use the exit to build a LOGONID card if your installation standards require this. You can determine the submitting user from a variety of sources. These include:
- Job name specified in the JCL
- Job accounting information
- Programmer name field
- Occurrence owner ID
- Occurrence authority group ID.

The SEQQSAMP member EQQUX001 contains a sample job-submit exit. This sample assigns a specific user ID based on the job name defined in the operation.

Job-library-read exit

The job-library-read exit is called when IBM Workload Scheduler for z/OS cannot find the JCL for a job in the JCL repository data set (EQQJSnDS). By default, IBM Workload Scheduler for z/OS searches the concatenation of data sets assigned to the EQQJBLIB ddname in the controller JCL procedure. If you want IBM Workload Scheduler for z/OS to search other data sets, install EQQUX002 to perform this function.

Dynamic allocation of JCL is very useful if your installation operates as a computer services bureau for several independent customers or departments. When independent job libraries are concatenated on the EQQJBLIB statement, duplicate member names can occur in different job library data sets. By placing JCL in separate job libraries and then using EQQUX002 to dynamically allocate a library for a particular application, you can more easily protect each customer's JCL.

Also consider using EQQUX002 to enhance performance if you have many large partitioned data sets (PDS) concatenated to EQQJBLIB. To find a member in the last data set of the concatenation, IBM Workload Scheduler for z/OS must read the directory of all preceding PDSs, which can present a significant overhead. Consider defining a PDS and a corresponding ddname for each computer workstation. EQQUX002 can then search a specific library. If no JCL is found, you can let IBM Workload Scheduler for z/OS search the EQQJBLIB concatenation for the JCL.

The SEQQSAMP member EQQUX002 contains a sample job-library-read exit. This sample searches a ddname MYJOBLIB before EQQJBLIB.

Event-filtering exit

The event-filtering exit is called when an IBM Workload Scheduler for z/OS event writer is about to write an event to the event data set or, where EWSEQNO is used, add the event to an XCF or NCF queue. In this exit, you can choose to discard events created by JES and SMF exits, or you can indicate that an event that would normally be queued to JCC is not processed by the JCC.

EQQUX004 is commonly used to filter the events created by nonproduction work. If you run a significant number of test jobs and other work, and your job naming standards let you do so, consider using EQQUX004 to filter the nonproduction work.

The SEQQSAMP member, EQQUX004, contains a sample event-filtering exit that includes or excludes events based on the job name.

SYSOUT archiving exit

The SYSOUT archiving exit, EQQUX005, is called by the job completion checker during processing of SYSOUT data sets for a job. The exit can be called several times for the same job as the JCC progresses through the various SYSOUT data sets.

This exit is commonly used to change the defined SYSOUT disposition, depending on the success or failure of the job. Note that EQQUX005 is a tracker exit, although the success or failure of an operation is ultimately determined by the controller.

The SEQQSAMP member, EQQX5ASM, contains a sample SYSOUT archiving exit. This sample requeues SYSOUT for failing jobs to a separate class from the one used for successful jobs. This is useful if you want to print the JCL of failed jobs for your job recovery operators. SYSOUT for jobs that complete successfully can be written to an output class managed by a system writer.

Incident-record-create exit

EQQUX006 is called by the job completion checker to build an incident record when a JCC message table specifies that an incident record should be created. The EQQUX006 sample supplied with IBM Workload Scheduler for z/OS produces incident records in one format. But, you can create your own format by replacing the sample exit with your own version.

The SEQQSAMP member, EQQX6ASM, contains a sample incident-record-create exit. This exit is called by the JCC incident file writer to create one or more records that can be written to the incident data set, if required. Selected error codes can be processed and passed as symbolic parameters to a submitted JCL stream so that you can generate records into a problem database or notify a TSO user ID of a particular failure.

The SEQQSAMP member, EQQX6JOB, contains the JCL for the job submitted by EQQX6ASM. EQQUX006 is a tracker exit, and the success or failure of an operation is ultimately determined by the controller.

Operation-status-change exit

The operation-status-change exit is called whenever an operation in the current plan changes status. This exit is often used as an interface to a problem management system, such as Information Management.

The SEQQSAMP member, EQQX7ASM, contains a sample operation-status-change exit (EQQUX007). This exit creates and submits a batch job whenever the status of an operation changes to ended-in-error. The job submitted by the exit, represented by SEQQSAMP member EQQX7JOB contains a CLIST that performs specific tailoring on the data passed to it by EQQUX007. The CLIST generates another job that could be used to create a problem record.

If you choose to implement an interface to your problem management system using EQQUX007, remember that *every* change of status to E invokes the exit. Consider filtering out those operations that were set to error by dialog users because they might not represent *real* errors.

Operation-initiation exit

The operation-initiation exit is called by IBM Workload Scheduler for z/OS when an operation is ready to be started at a workstation that specifies a user-defined destination ID. The exit is used to communicate with various operating environments. Two sample EQQUX009 exits are located in the SEQQSAMP library.

The SEQQSAMP member EQQUX9N contains a sample operation-initiation exit written in PL/I. The sample has been designed specifically to be used with the EQQCVM2 and EQQUX0N samples which provide a tracker for a VM operating system, but could be modified to communicate with other operating environments. See "Tracker for VM" on page 399 for more information about the VM tracker.

The SEQQSAMP member EQQX9OS2 contains a sample operation-initiation exit written in assembler. The sample has been designed specifically to be used with the EQQOS2TR and EQQOS2ST samples which provide a tracker for an OS/2 environment, but could be modified to communicate with other operating environments. See "Tracker for OS/2" on page 400 for more information about the OS/2 tracker.

The SEQQSAMP member EQQX9AIX contains a sample operation-initiation exit written in assembler. The sample has been designed specifically to be used with the EQQAIXTR and EQQAIXST samples, which provide a tracker for an AIX environment. You can use these samples to communicate with other UNIX environments if the shell script is compatible. See "Tracker for AIX" on page 400 for more information about the AIX tracker.

JCL-variable-substitution exit

EQQJVXIT contains an assembler sample of the JCL-variable-substitution exit. When you define a JCL variable, you can specify the name of an exit that is called when substitution of the variable is required. The exit can be called at either job setup or job submission, but is not called for promptable setup-variables. For System Automation command variables, the exit is invoked at command submission. You can use the exit to supply the value of a variable.

Job-tracking log write exit

Member EQQUX011 depicts a possible scenario to use this exit. This member contain also the JCL to assembly and link the load module.

EQQDELDS/EQQCLEAN catalog-exit

EQQDELDS/EQQCLEAN tries to load the EQQUXCAT exit. If the exit module is successfully loaded, EQQDELDS/EQQCLEAN calls it before executing the catalog action for each involved data set. If the exit passes back a return code different from 0, the action is not executed and a message is logged to identify the skipped data set. The provided exit sample prevents the deletion of data sets having the qualifier starting with SYS1.MAC.

EQQCLEAN GDG resolution exit (EQQUXGDG)

EQQUXGDG is invoked by the EQQCLEAN program just before executing any single GDG overwrite action into JES control blocks. The overwrite action will not be executed when the EQQUXGDG return code is set to a value different from 0.

The provided exit sample prevents the GDG overwrite action for a job with jobname MYJOB and the DDNAME is NOSIMDD or GDG data set name starts with TST.GDG.

Application-description-validation exit

The application-description-validation exit (EQQUXPIF) is called by the PIF to validate the application description during its update. The SEQQSAMP member, EQQUXPIF, contains a sample for the validation of the application description update.

DP batch scheduling environment exit

The scheduling environment exit is called by DP batch (extend or re-plan) for every non fault-tolerant workstation operation as soon as the plan is produced, so that users can change the scheduling environment associated to the operation depending on their requirements.

The EQQDPX01 sample, provided with SEQQSAMP, assigns the scheduling environment value according to the special resource name used by the operation each time this starts with 'SCHENV='. For example, an operation allocating the special resource SCHENV=DB2ACTIVE, will have the scheduling environment name set to DB2ACTIVE.

This exit can affect the performance of DP batch. For this reason, it should be used only when necessary and care should be taken to limit I/O processing (for example, needed files should be opened and closed at the start and the end of the exit only).

Job-tailoring prevention exit

The job-tailoring prevention exit is called when IBM Workload Scheduler for z/OS is about to submit a batch job or start a started-task. Typically, this exit is used to prevent some jobs from being tailored by the pre-submitter task with the //TIVDST OUTPUT statements, when the restart and cleanup function is active.

The SEQQSAMP member EQQUX013 contains a sample job-submit exit. This sample sets a return code 0012, if the job name of the JCL being submitted by the Workstation Analyzer matches the job name defined in the user exit.

Time Dependent Operation user exit

The controller calls the Time Dependent Operation user exit each time an operation become ready in a z/OS environment, in order to know if a negative or positive offset must be added to the job start time, to decide if the operation can be started or not. The controller uses the returned offset to update the job start time only. In particular, the process leaves unchanged the latest start time.

The exit receives as input from the controller some data identifying the job, to be used by the exit to decide the appropriate offset to be calculated.

When implementing the exit remember that the workstation name, passed as input parameter, is the one on which the operation is defined. As a consequence, the submission destination might be unknown at exit invocation time, in the following conditions:

- An alternate workstation is defined. In this case, consider defining the alternate workstation equivalent to the primary workstation, from an offset calculation point of view.
- The workstation is defined with the virtual option. In this case, define the virtual workstation in the UX14IN file.

Open Systems integration

The sample library contains a number of programs to demonstrate how you can exploit IBM Workload Scheduler for z/OS open interfaces to communicate with various operating environments. The samples provided include trackers for a VM environment using NJE for communication, an OS/2 environment using TCP/IP for communication, and an AIX environment.

Tracker for VM

The sample library members EQQCVM2, EQQUX9N can be used to create a tracker for VM operating environments. When installed, the tracker enables you to schedule an operation on a computer workstation to initiate and track processing in a VM environment. The request to start the processing is communicated to VM using NJE. Status is reported back to the controller from VM.

To install a VM tracker in your environment, follow these steps:

- Specify a symbolic destination ID in the USER keyword of the ROUTOPTS initialization statement.
- Create a computer automatic workstation which specifies the same destination ID.
- Workstations that specify a user-defined destination are initially set to `unknown` status every time the controller is started. You are responsible for setting the status of the VM workstation to `ACTIVE` status. You can use the WSSTAT command, the EQQUSINW subroutine or the IBM Workload Scheduler for z/OS modify current plan (MCP) dialog. You should consider setting the workstation to `ACTIVE` status in EQQUX000, the start/stop exit. The sample library member EQQUX0N contains a sample EQQUX000 to set workstation status using the EQQUSINW subroutine.
- The sample library member EQQUX9N contains an operation-initiation exit (EQQUX009), written in PL/I which is loaded by the controller. The exit is called by the external router subtask when an operation is ready to be started at a workstation which specifies a user-defined destination ID. You need to update the sample to define the destination ID, the receiving VM user and a nonheld SYSOUT class. This sample assumes that the VM execs are stored on VM and that a one line member in EQQJBLIB describes the name of the EXEC and any required parameters.
- Specify `EXITS CALL00(YES) LOAD(EQQUX0N)` and `CALL09(YES) LOAD(EQQUX9N)` for the controller.
- Modify the JCL procedure for the controller to include:
  - A ddname UX09LOG to get a log of all transmissions to VM
  - A ddname SYSPRINT if you want a message from EQQUX0N to be written, or delete the PUT statements in EQQUX0N.

- The sample library member EQQCVM2 contains the components required on VM. It consists of 2 CMS EXECs OPCWATCH and OPCSTAT. OPCWATCH is a REXX EXEC to drive a VM AUTOLOG machine. When operations are received from the controller by EQQUX9N, OPCWATCH starts the required VM EXEC. The OPCSTAT EXEC reports status back to the controller by submitting batch jobs via NJE. The batch jobs execute the EQQEVPGM program.

Tracker for OS/2

The sample library members EQQOS2TR, EQQOS2ST, and EQQX9OS2 can be used to create a tracker for OS/2 operating environments. When installed, the tracker enables you to schedule an operation on a computer workstation which, when ready to be started, sends commands to be executed on an OS/2 workstation. Status is reported back to the controller from OS/2.

To install an OS/2 tracker based on the samples in your environment, follow these steps:

- Specify a symbolic destination ID in the USER keyword of the ROUTOPTS initialization statement.
- Create a computer automatic workstation which specifies the same destination ID.
- Workstations that specify a user-defined destination are initially set to `unknown` status every time the controller is started. You are responsible for setting the status of the OS/2 workstation to `ACTIVE` status. You can use the WSSTAT command, the EQQUSINW subroutine or the IBM Workload Scheduler for z/OS modify current plan (MCP) dialog. You should consider setting the workstation to `ACTIVE` status in EQQUX000, the start/stop exit. The sample library member EQQUX0N contains a sample EQQUX000 to set workstation status using the EQQUSINW subroutine.
- Download the EQQOS2TR and EQQOS2ST sample library members to your OS/2 workstation. EQQOS2TR contains code which executes in OS/2, EQQOS2ST contains parameters that are passed to EQQOS2TR. The program is written in REXX and OS/2 command language, instructions for modifying the code and installing can be found in a comment block in EQQOS2TR.
- The sample library member EQQX9OS2 contains an operation-initiation exit (EQQUX009), written in assembler, which is loaded by the controller. The exit is called by the external router subtask when an operation is ready to be started at a workstation which specifies a user-defined destination ID. You need to update the sample to define the destination ID and details about the receiving OS/2 workstation.
- Specify `EXITS CALL00(YES) LOAD(EQQUX0N)` and `CALL09(YES) LOAD(EQQX9OS2)` for the controller.

The commands you want to execute in the OS/2 environment can be defined in EQQJBLIB and sent to OS/2 by the EQQX9OS2, or you can choose to keep the information only in OS/2. Status of the operation is reported to the controller by batch jobs executing the EQQEVPGM submitted from the OS/2 tracker.

Tracker for AIX

If you use AIX/6000 Version 3 Release 2 Modification Level 4 with the optional installable TCP/IP feature, you can install the tracker to control the workload from the controller. In other AIX operating environments, you can use the sample library members EQQAIXTR, EQQAIXST, and EQQX9AIX to create a tracker for AIX. When installed, the tracker lets you schedule an operation on a computer workstation which, when ready to be started, sends commands or tasks to be executed in an AIX environment. Status is reported back to the controller from AIX. The samples have been developed and tested in an AIX environment, but can be ported to any UNIX environment if the shell script is compatible.

To install an AIX tracker based on the samples in your environment, follow these steps:

- Specify a symbolic destination ID in the USER keyword of the ROUTOPTS initialization statement.
- Create a computer automatic workstation which specifies the same destination ID.
- Workstations that specify a user-defined destination are initially set to `unknown` status every time the controller is started. You are responsible for setting the status of the AIX workstation to `ACTIVE` status. You can use the WSSTAT command, the EQQUSINW subroutine or the IBM Workload Scheduler for z/OS modify current plan (MCP) dialog. You should consider setting the workstation to `ACTIVE` status in EQQUX000, the start/stop exit. The sample library member EQQUX0N contains a sample EQQUX000 to set workstation status using the EQQUSINW subroutine.
- Download the EQQAIXTR and EQQAIXST sample library members to your AIX system. EQQAIXTR contains code which executes in AIX, EQQAIXST contains parameters that are passed to EQQAIXTR. The program is written in shell script, instructions for modifying the code and installing can be found in a comment block in EQQAIXTR.
- The sample library member EQQX9AIX contains an operation-initiation exit (EQQUX009), written in assembler, which is loaded by the controller. The exit is called by the external router subtask when an operation is ready to be started at a workstation which specifies a user-defined destination ID. You need to update the sample to define the destination ID and details about the receiving AIX environment.
- Specify `EXITS CALL00(YES) LOAD(EQQUX0N)` and `CALL09(YES) LOAD(EQQX9AIX)` for the controller.

The commands or tasks that you want to execute in the AIX environment can be defined in EQQJBLIB and sent to AIX by the EQQX9AIX, or you can choose to keep the information only in the AIX environment. Status of the operation is reported to the controller by batch jobs executing the EQQEVPGM submitted from the AIX tracker.

IBM Workload Scheduler for z/OS auditing package

There are three ways to start the IBM Workload Scheduler for z/OS auditing functions:

- Invoking it interactively (option 10.1 from the main menu)
- Submitting from the dialog a batch job (option 10.2 from the main menu)
- Submitting a batch job outside the dialog

The sample library member EQQAUDIB contains a job that is customized at installation time and that you can submit outside the dialog to start the auditing function in those cases when either of the two simpler methods cannot be used. The first two methods are useful when there is an urgency to create reports from all the auditing data sets (job-tracking, track-log, or extended-auditing). This facilitates searching for answers to critical questions without having to spend too much time to examine the input records with the aid of the mappings listed in the

Diagnosis Guide and Reference. The third way is useful in case of a planned utilization: many installations have a need to create and store auditing trails for a set period of time. In this case, an IBM Workload Scheduler for z/OS auditing job (copied from the EQQAUDIB CUSTOMIZED) can be defined to run automatically after every daily plan EXTEND or REPLAN, using the data set referenced by EQQTROUT or EQQDBOUT as input.

Even if you are not *required* to create an auditing trail regularly, the report generated can provide quick answers in determining who in your organization requested a function that caused some business application to fail, or to trace the processing of a job that failed and was rerun many times. If your AUDIT initialization statement specifies all data for JS update requests, you can use the auditing report to compare against the master JCL to determine exactly which JCL statements were changed.

The auditing program reads the JTARC or DBARC data set and the currently open EQQJT*nn* or EQQDB*nn* in their entirety. Thus the report produced contains almost always some amount of obsolete data generated by the old records still visible in the open data set and not yet overwritten by new data. Any information appearing in the auditing report after this header is old or residual, and you must very carefully review timestamps when using it. Note that this old information is also included in the statistical information at the end of of the EQQAUDIT report. If accurate statistics are required, they can be obtained by generating the report using the EQQTROUT dataset (INPUT TRL).

Figure 5, Figure 6 on page 403, and Figure 7 on page 404 show the parts of the report that you can generate using the auditing package. The level of reporting for database updates is dependent on the values you specify in the AUDIT initialization statement.

```
DATE/TIME OF THIS RUN: 110316 18.02



*****************************************************************
*           SAMPLE OPC/ESA AUDIT/DEBUG REPORT                   *
*            P R O G R A M  P A R A M E T E R S                 *
*****************************************************************
* INPUT SOURCE    : TRL                                         *
* SEARCH-STRING   :                                             *
* START DATE      :                                             *
* START TIME      :                                             *
* END DATE        :                                             *
* END TIME        :                                             *
*****************************************************************



****************************************************
* LINES INSERTED BY PROGRAM ARE MARKED '=====>'    *
* UNKNOWN FIELD VALUES ARE PRINTED AS '?'          *
* SUMMARY OF SELECTED RECORDS PRINTED ON LAST PAGE *
****************************************************
```

Figure 5. EQQAUDIT report sample: header page

```
  DATE/TIME OF THIS RUN: 110316 18.02

 =============> NOW READING FROM EQQTROUT
03/07 11.19.58 CP   UPDT BY PERTICA  MCP MODIFY              APPL: APPLPLUGINS       IA: 110224 1210  PRTY: 5
                                      - OPNO:001  TYPE: EX-COMMAND ISSUED
 03/07 11.19.59 25   SCHD BY OPC      JOBNAME: PLUGIN    AD: APPLPLUGINS       OCC IA: 1102241210 TOKEN: 0000000000000000
 03/07 11.19.59 29                    PROCESSED PC-OCCURRENCE TOKEN: C76158BA53524889 CALLER: WSA REQUEST TIMESTAMP: 0111066F11195946
                                                  CLEANUPEVENT  USER  ADID: APPLPLUGINS      OP: 001 SCHENV:
 03/07 11.19.59 CP   UPDT BY OPC_WSA  OP.ZCE2_001 IN APPLPLUGINS      IS SET TO S   JOBNAME: PLUGIN
 03/07 11.19.59 CP                    OP.ZCE2_001 IN AD/IA: APPLPLUGINS      3902241210  JOBNAME: PLUGIN
 03/07 11.19.59 29                    PROCESSED 2O-DESTINATION NAME: ITAVP2   WS STATUS: A OP SYSTEM: WINDOWS
 03/07 11.19.59 29                    PROCESSED 2P-OCCURRENCE TOKEN: C76158BA53524889 OP: 001 WSNAME: ZCE2
 03/07 11.19.59 CP   UPDT BY PX2      OP.ZCE2_001 IN APPLPLUGINS      IS SET TO E   JOBNAME: PLUGIN   ERROR CODE: OSUB RESTART BYPASS
 03/07 11.19.59 CP                    OP.ZCE2_001 IN AD/IA: APPLPLUGINS      3902241210  JOBNAME: PLUGIN  R CODE: OSUB RESTART BYPASS
 03/07 11.20.00 29                    PROCESSED 2F-OCCURRENCE TOKEN: C76158BA53524889 OP: 001 WSNAME: ZCE2
 03/07 11.20.02 JS   READ BY PERTICA  KEY: APPLPLUGINS     1102241210  OPNO:001  JOBNAME: PLUGIN    WSNAME: ZCE2  READ FROM: JS-FILE
                                       //TASKTYPE=ws
                                       </jsdl:application> </jsdl:jobDefinition>
 03/07 11.21.47 JS   READ BY PERTICA  KEY: APPLPLUGINS     1102241210  OPNO:001  JOBNAME: PLUGIN    WSNAME:
 03/07 11.21.52 JS   DLET BY PERTICA  KEY: APPLPLUGINS     1102241210  OPNO:001
 03/07 11.21.53 JS   READ BY PERTICA  KEY: APPLPLUGINS     1102241210  OPNO:001  JOBNAME: PLUGIN    WSNAME:
 03/07 11.22.25 CP   UPDT BY PERTICA  MCP MODIFY              APPL: APPLPLUGINS       IA: 110224 1210  PRTY: 5
                                      - OPNO:001  TYPE: OP. ADDED          JOBNAME: PLUGIN    WSID: ZCE2  USERDATA:
                                      - OPNO:001  TYPE: JOB OPTIONS        JOBNAME: PLUGIN    TIMEJOB: NO  AUTOSUB: YES CONDRJ: NO
                                      - OPNO:001  TYPE: DELETE EXTENDED INFO
                                      - OPNO:001  TYPE: OP. ADDED          JOBNAME: PLUGIN    WSID: ZCE2  USERDATA:
                                      - OPNO:001  TYPE: JOB STATUS       NEW OP. STATUS: R
 03/07 11.22.27 CP   UPDT BY PERTICA  MCP MODIFY              APPL: APPLPLUGINS       IA: 110224 1210  PRTY: 5
                                      - OPNO:001  TYPE: EX-COMMAND ISSUED
 03/07 11.22.27 25   SCHD BY OPC      JOBNAME: PLUGIN    AD: APPLPLUGINS       OCC IA: 1102241210 TOKEN: 0000000000000000
 03/07 11.22.27 29                    PROCESSED PC-OCCURRENCE TOKEN: C76158BA53524889 CALLER: WSA REQUEST TIMESTAMP: 0111066F11222728
                                                  CLEANUPEVENT  USER  ADID: APPLPLUGINS      OP: 001 SCHENV:
 03/07 11.22.27 CP   UPDT BY OPC_WSA  OP.ZCE2_001 IN APPLPLUGINS      IS SET TO S   JOBNAME: PLUGIN
 03/07 11.22.27 CP                    OP.ZCE2_001 IN AD/IA: APPLPLUGINS      3902241210  JOBNAME: PLUGIN
 03/07 11.22.27 29                    PROCESSED 2P-OCCURRENCE TOKEN: C76158BA53524889 OP: 001 WSNAME: ZCE2
 03/07 11.22.27 CP   UPDT BY PX2      OP.ZCE2_001 IN APPLPLUGINS      IS SET TO E   JOBNAME: PLUGIN   ERROR CODE: OSUB RESTART BYPASS
 03/07 11.22.27 CP                    OP.ZCE2_001 IN AD/IA: APPLPLUGINS      3902241210  JOBNAME: PLUGIN  R CODE: OSUB RESTART BYPASS
 03/07 11.22.27 29                    PROCESSED 2F-OCCURRENCE TOKEN: C76158BA53524889 OP: 001 WSNAME: ZCE2
 03/07 11.29.19 29                    PROCESSED NF-HANDSHAKE EVENT.  PULSE FROM TRACKER ON: MVS/ESA AT DEST: ********
 03/07 11.32.24 36                    BACKUP WAS TAKEN.  DDNAME OF BACKUP CP: EQQCP1DS  DDNAME OF JT: EQQJT04
 03/07 11.32.43 37                    EDP  TYPE DATA HAS BEEN LOGGED
 03/07 11.32.43 20                    JOB TRACKING START  EVENT
 03/07 11.32.43 29                    PROCESSED NF-HANDSHAKE EVENT.  REQUEST BY TRACKER ON: MVS/ESA WITH ID: HWSZ600
 03/07 11.32.43 29                    PROCESSED IJ-SUBMIT SYNCHR.  WS:CPU1  WS SEQ£: 00006  EVDS SEQ£: 00006  REQ£: 098
 03/07 11.32.43 29                    PROCESSED NF-HANDSHAKE EVENT.  CONFIRM BY TRACKER ON: MVS/ESA AT DEST: ********
 03/07 11.32.43 29                    PROCESSED NF-HANDSHAKE EVENT.  PULSE FROM TRACKER ON: MVS/ESA AT DEST: ********
 03/07 11.32.46 29                    DISCARDED 2O-DESTINATION NAME: CWSDDEST WS STATUS: A OP SYSTEM: Z_OS
 03/07 11.32.47 29                    DISCARDED 2O-DESTINATION NAME: NEWTEST4 WS STATUS: O OP SYSTEM:
 03/07 11.32.47 29                    PROCESSED 2O-DESTINATION NAME: CWSGDEST WS STATUS: A OP SYSTEM: Z_OS
 DATE/TIME OF THIS RUN: 110316 18.02

 03/07 11.32.48 29                    PROCESSED 2I-BIND OK      TOKEN: C76690B9DF9C0A47   OP: 001 REMOTE OCC IA: 110224 1111
 03/07 11.32.48 29                    PROCESSED 2S-OCCURRENCE TOKEN: C76690B9DF9C0A47 OP: 001 WSNAME:      JOB NAME:
 03/11 11.48.31 CP   UPDT BY PERTICA  MCP MODIFY              APPL: APPLCPU1          IA: 010311 1111  PRTY: 5
                                      - OPNO:001  TYPE: EX-COMMAND ISSUED
 03/11 11.48.31 25   SCHD BY OPC      JOBNAME: VPJOB1    AD: APPLCPU1          OCC IA: 0103111111 TOKEN: 0000000000000000
 03/11 11.48.31 29                    PROCESSED PC-OCCURRENCE TOKEN: C7742F948FA8B149 CALLER: WSA REQUEST TIMESTAMP: 0111070F11483174
                                                  CLEANUPEVENT  USER  ADID: APPLCPU1         OP: 001 SCHENV:
 03/11 11.48.31 CP   UPDT BY OPC_WSA  OP.CPU1_001 IN APPLCPU1         IS SET TO S   JOBNAME: VPJOB1
 03/11 11.48.31 CP                    OP.CPU1_001 IN AD/IA: APPLCPU1         2903111111  JOBNAME: VPJOB1
 03/11 11.48.31 29                    PROCESSED IJ-SUBMIT JCL     AD/IA: APPLCPU1        0103111111  VPJOB1  (JOB00055)  SEQ£: 000008
 03/11 11.48.31 29                    PROCESSED A1-JOB CARD READ  VPJOB1  (JOB00055)  AT: 11.48.31.82
 03/11 11.48.31 29                    PROCESSED A2-JOB START      VPJOB1  (JOB00055)  AT: 11.48.31.87  ON NODE: ZDIS1013  CM-DATA: NO
 03/11 11.48.31 29                    PROCESSED A3-STEP END       VPJOB1  (JOB00055)  AT: 11.48.31.93  PRSTEP: S1         CODE: 0000
 03/11 11.48.32 29                    PROCESSED A3-STEP END       VPJOB1  (JOB00055)  AT: 11.48.32.05  PRSTEP: S2         CODE: 0000
 03/11 11.48.32 29                    PROCESSED A3-JOB COMPLETE   VPJOB1  (JOB00055)  AT: 11.48.32.05        CODE:  0000 CM-DATA: NO
 03/11 11.48.32 28                    FEEDBACK FOR AD: APPLCPU1         £ OPS UPDATED: 000 FOR DURATION: 000 FOR DEADLINE: 000
 03/11 11.48.32 CP   UPDT BY OPC_JT   OP.CPU1_001 IN APPLCPU1         IS SET TO C   JOBNAME: VPJOB1                 RESTART BYPASS
 03/11 11.48.32 CP                    OP.CPU1_001 IN AD/IA: APPLCPU1         2903111111  JOBNAME: VPJOB1              RESTART BYPASS
 03/11 11.48.32 29                    PROCESSED A3-JOB TERMINATE  VPJOB1  (JOB00055)  AT: 11.48.32.07                    CM-DATA: NO
 03/11 11.49.31 29                    PROCESSED 0W-WKST EVENT. WSID: FTW1 LINK STATUS: U WS STATUS:   FULLY LINKED:   LIMIT:
 03/11 11.50.00 36                    BACKUP WAS TAKEN.  DDNAME OF BACKUP CP: EQQCP2DS  DDNAME OF JT: EQQJT02
 =============> EOF REACHED ON EQQTROUT
```

Figure 6. EQQAUDIT report sample: report pages

```
DATE/TIME OF THIS RUN: 110316 18.02


 ***************************************************
 *                              *RECORDS * EVENTS *
 *     E V E N T  T Y P E    NO *  READ  *SELECTED*
 ***************************************************
 * DAILY PLAN STATUS  RECORD  01 * 000001 * 000000 *
 * DAILY PLAN WORK STATIONS   02 * 000010 * 000000 *
 * DAILY PLAN OCC/OPER/CONDS  03 * 000020 * 000000 *
 * JOB TRACKING START         20 * 000054 * 000054 *
 * MANUAL OPERATIONS          23 * 000132 * 000132 *
 * MODIFY CURRENT PLAN        24 * 000176 * 000176 *
 * OPERATION SCHEDULED        25 * 000049 * 000049 *
 * FEEDBACK DATA              28 * 000004 * 000004 *
 * AUTOMATIC OPERATIONS       29 * 002327 * 002327 *
 * DATA BASE UPDATE           32 * 000263 * 000227 *
 * BACKUP LOG RECORD          36 * 000056 * 000056 *
 * DATA LOG RECORD               * 000056 * 000056 *
 * MULTI-RECORD EVENTS           * 000006 * 000002 *
 ***************************************************


 ************************************************************
 *  MCP PERFORMANCE *  NO OF  *   E L A P S E D   T I M E   *
 *  TYPE OF UPDATE  * UPDATES *  M I N  *  M A X  *  A V G  *
 ************************************************************
 * MCP ADD          *  000001 *   0.01  *   0.01  *   0.01  *
 * MCP MODIFY       *  000066 *   0.01  *   0.22  *   0.01  *
 * MCP CHANG WS     *  000001 *   0.03  *   0.03  *   0.03  *
 * MCP VARY WS      *  000001 *   0.01  *   0.01  *   0.01  *
 ************************************************************

 LONGEST MCP-EVENTS:

 03/09 12.03.00 CP  UPDT BY PERTICA  MCP MODIFY  0.22SEC  APPL: APPLPLUGINS   IA: 110224 1210  PRTY: 5
 03/11 11.47.49 CP  UPDT BY PERTICA  MCP MODIFY  0.07SEC  APPL: APPLCPU1      IA: 110224 1111  PRTY: 5
 03/09 16.39.22 CP  UPDT BY PERTICA  MCP MODIFY  0.05SEC  APPL: APPLPLUGINS   IA: 110224 1210  PRTY: 5

           ***** END OF REPORT *****
```

Figure 7. EQQAUDIT report sample: summary page

Viewing output from the ended-in-error list

SEQQSAMP library member EQQOUTL contains panels and a CLIST that you can use to gain access to the ISPF OUTLIST utility from the ended-in-error list panel in the Modify Current Plan dialog. The OUTLIST utility lets you browse, print, and delete the output list of failed jobs.

The sample provides an additional row command, Q, that you can use on the ended-in-error list panel. This command takes you into ISPF option 3.8. You can view output only on the JES spool of the system where the controller is started.

The ability to view, print, and delete the output for failed jobs directly from the IBM Workload Scheduler for z/OS dialog creates a more integrated environment for those operators responsible for restart and recovery functions.

NetView samples

The following text describes the SEQQSAMP members relating to NetView. The samples illustrate ways that you can use IBM Workload Scheduler for z/OS and NetView functions to automate certain tasks. The samples are indicative of automation tasks that many installations require.

Deadline WTO message

SEQQSAMP member EQQNETW1 contains a sample you can use to stop an online system when IBM Workload Scheduler for z/OS issues a deadline WTO message. This sample REXX EXEC can stop an IMS system in response to the WTO message for the operation representing the IMS system.

The EXEC is written as a general NetView command list and can be started by an operator. The EXEC will require slight modification if you want to use it as a MESSAGE AUTOMATION command list.

Responding to WTO operations

SEQQSAMP member EQQNETW2 contains a sample that you can use to take the action defined in a WTO operation. This sample contains JCL to link-edit and compile a sample PL/I program that will cause NetView to issue a VTAM VARY command. The command is built using the operation text of a WTO operation.

The PL/I program calls the EQQUSINT module to change the status of an operation in the current plan.

Changing operation status from NetView

SEQQSAMP member EQQNETW3 contains a sample that you can use as a base for many automation tasks. This sample uses EQQEVPGM to change the status of a current-plan operation in response to a WTO operation. The sample is coded as a NetView CLIST.

You can insert code to interpret the WTO so that some action is taken by NetView before the call to EQQEVPGM.

z/OS hiperbatch support

If you want to use IBM Workload Scheduler for z/OS to control Hiperbatch activity on your system, consider using the following samples to assist implementation.

Hiperbatch is a z/OS performance enhancement that works with DLF to let batch jobs and started tasks share an in-storage copy of a data set, or data object. You can use IBM Workload Scheduler for z/OS to control connection to the DLF object and to purge the object when IBM Workload Scheduler for z/OS determines that no further operations in the current plan require access to the data. IBM Workload Scheduler for z/OS initiates purge processing if the data object will not be used by the immediate successor operation, or other ready operations.

IBM Workload Scheduler for z/OS initiates purge processing also for operations that have ended in error, unless the *keep on error* value specifies that the resources allocated to the operations must be kept.

The SEQQSAMP member EQQDLFX provides a sample DLF installation exit, which ensures that all objects are retained. The sample EQQPROC is used by IBM Workload Scheduler for z/OS to purge a DLF object when it is no longer needed. EQQPROC invokes program EQQPURGE, which reads JCL from the data set identified by the JCLIN DD statement, and updates the JCL with the name of the object to be purged. Once the JCL is updated, EQQPURGE writes the JCL to the JES internal reader. Sample EQQJCLIN contains the sample JCL for the JCLIN file.

Deleting data sets based on JCL disposition and catalog status

SEQQSAMP member EQQDELDI contains JCL to run the sample program, EQQDELDS, that you can use to delete data sets based on the disposition specified in the JCL and the current status of the data set in the catalog. EQQDELDS is not a function of IBM Workload Scheduler for z/OS. The program is supported by IBM Workload Scheduler for z/OS development to help customers who require this function and who do not want to change existing application JCL. To run this program, modify the JCL statements identified by the characters <==== to meet your installation standards. EQQDELDS deletes any data set that has a disposition of (NEW,CATLG), or (NEW,KEEP) for SMS, if the data set is already present in the catalog. It optionally handles passed data sets.

Note: Data sets are not deleted if they are referenced in a previous step with or in the same step with DISP different from NEW, unless DELLOGIC is set to 2. Data sets defined with indirect or symbolic VOLSER are not deleted by IDCAMS. DELETE only uncatalogs the data set; it does not delete the data set from the SYSRES volume, even if SCRATCH is specified.

You can use EQQDELDS to avoid *not catlgd 2* situations. EQQDELDS cannot run concurrently with subsequent steps of the job in which it is inserted. Therefore, if Smartbatch is active, define EQQDELDS with ACTION=BYPASS in the Smartbatch user control facility.

EQQDELDS supports these types of delete processing:
- DASD data sets on primary volume(s) are deleted using IDCAMS.
- Tape data sets are deleted using IDCAMS NOSCRATCH. This does not cause mount requests for the specified tape volumes.
- DFHSM-migrated data sets are deleted using the ARCHDEL (ARCGIVER) interface. data sets are not moved to primary volumes (recalled) before deletion.

EQQDELDS logs all actions performed in text lines written to the SYSPRINT DD.

A nonzero return code from IDCAMS or ARCHDEL causes EQQDELDS to end.

You can use the EQQUX001 exit to automatically add an EQQDELDS step to all jobs submitted by IBM Workload Scheduler for z/OS.

See also *Limitation on the number of job steps* in *IBM Workload Scheduler for z/OS: Managing the Workload*.

Miscellaneous samples

Besides the samples already described, the SEQQSAMP library also contains samples for:
- Job-tracking capabilities on VM systems (member EQQCVM)
- JCC message-table coding (member EQQJCCTB).

MASS update samples

About this task

The EQQYCBAG member of the EQQSAMP library provides a sample in which the Batch Command Interface Tool (BCIT) is used to unload a group application, and all applications belonging to it, into a sequential file in batchloader control statement format.

The group applications, as well as other applications, can be modified via the batchloader control statements. The sequential file can thereafter be used as input to the batchloader run.

This sample consists of two jobs:

1. The 'unload' job, that uses the batch command interface tool
2. The 'load' job, that uses the batchloader.

Before running the job, you need to customize it with correct values for the job card name, data set names, subsystem name, and so on.

Notices

© (your company name) (year).
Portions of this code are derived from IBM Corp. Sample Programs.
© Copyright IBM Corp. _enter the year or years_.

Trademarks

Terms and conditions for product documentation

Permissions for the use of these publications are granted subject to the following terms and conditions.

Index

Product Number: 5698-T08

Printed in USA