HCL Workload Automation for Z
Planning and Installation
Version 10.2 SPE (Revised July 2025)

Note

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

This edition applies to version 10, release 2, modification level 0 of HCL Workload Automation for Z (program number 19OP1249) and to all subsequent releases and modifications until otherwise indicated in new editions.

Contents

List of Figures

List of Tables

About this publication

HCL Workload Scheduler for Z: Planning and Installation describes the configuration planning and installation tasks of HCL Workload Automation for Z. Installation is the task of making a program ready to do useful work. This task includes adding the installation materials to your system, initializing the program, and applying PTFs to the program. When you install a product, you are carrying out decisions you made in the planning step. Customization, an optional step, gives you the opportunity to tailor the program to the behavior or special needs required for your site.

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

The term *scheduler*, when used in this publication, refers to HCL Workload Automation for Z. The term *DB2®*, when used in this publication, refers to both DATABASE 2 and DB2 Universal Database™.

This publication complements the HCL Workload Automation for Z HCL Workload Automation for Z: Program Directory.

The *Program Directory* is provided with the HCL Workload Automation for Z installation media. It describes all of the installation materials and gives installation instructions specific to the product release level or feature number. If any differences exist between this publication and the *Program Directory*, use the information in the *Program Directory*.

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 the section *Summary of enhancements* in the *Overview* manual.

For information about the APARs that this release addresses, see the HCL Workload Automation for Z: Program Directory.

Who should read this publication

Learn the audience of this publication.

This publication is intended for system programmers who are responsible for software on a z/OS system and plan on installing HCL Workload Automation for Z.

To use this publication effectively, you must be familiar with the following topics:

- Job control language (JCL)
- IBM® System Modification Program Extended (SMP/E)
- z/OS
- JES concepts and facilities
- Writing small code fragments in the Assembler H language
- Interactive System Productivity Facility (ISPF)
- Interactive System Productivity Facility/Program Development Facility (ISPF/PDF)

- Time-Sharing Option (TSO)
- Virtual Storage Access Method (VSAM) (desirable but not essential)

The HCL Workload Automation for Z Application Programming Interface (API) uses advanced program-to-program communication (APPC) services. Defining and configuring the conversation partners requires some knowledge of APPC services.

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 detailed information, see the appendix about accessibility in the *HCL Workload Automation User's Guide and Reference*.

Part I. Planning

This part provides an overview of the HCL Workload Automation environment and describes how to plan for the installation.

You can use HCL Workload Automation for Z to plan, control, and automate the entire production workload in your complex, not just the z/OS® batch subset. It automatically plans, controls, and monitors your production workload to maximize the throughput and optimize resource use, but lets you intervene manually when required.

To avoid security exposures, ensure that you follow all the instructions provided to secure data communication.

Chapter 1. Overview

The following topics introduce HCL Workload Automation for Z and its implementation. For a description of the HCL Workload Automation for Z terminology and functions, see the *Overview* manual.

Hardware requirements

HCL Workload Automation for Z operates on any IBM® hardware configuration supported by z/OS. For the minimum version supported, see the Program Directory.

HCL Workload Automation for Z needs a minimum region of 8 MB below the 16 MB line; at least 32 MB must be available above the 16 MB line. The region value depends strictly on the HCL Workload Automation for Z customization and workload. For HCL Workload Automation for Z to work correctly, you might need to specify a region value of 64 MB, which gives you all the available space below the 16 MB line plus 64 MB above the 16 MB line.

In particular, a region value of 64 MB is strongly recommended when the HCL Workload Automation for Z TCP/IP or APPC server is used for remote interfaces, such as the Dynamic Workload Console, PIF, and ISPF.

Consider to increase the region size specification if the server task will normally run for a long time (several weeks or months). Also make sure that the IEFUSI exit is not limiting the region size to a value less than the one coded in the JCL.

An HCL Workload Automation for Z dialog user needs a region of 3 MB below the 16 MB line; if you want to run EQQAUDIT interactively (option 9.10 of the main menu), this number then increases to 4 MB. HCL Workload Automation for Z report programs need a region of 3 MB below the 16 MB line.

HCL Workload Automation for Z uses less than 1 KB of 24-bit Common Service Area (CSA) storage. The amount of 31-bit Extended Common Service Area (ECSA) used is approximately 30 KB, plus 2 KB per active dialog user.

Software requirements and optional software

Installing and maintaining HCL Workload Automation for Z requires one of the following products:

- z/OS® (program number 5650-ZOS). For the minimum version supported, see the Program Directory.
- IBM® SMP/E for z/OS®. For the minimum version supported, see the Program Directory.

HCL Workload Automation for Z requires the functions provided by a z/OS® control program running on a z/OS® system. The Job Entry Subsystem might be either JES2 or JES3.

Controlling system

The following programs are required on the HCL Workload Automation for Z controlling system:

- HCL Workload Automation for Z Version 10.2 SPE (Revised July 2025) (program number 19OP1249). Both the base product (the tracker) and the controller features are required.

Controlled z/OS® systems

The following program is required on each z/OS system that is controlled by HCL Workload Automation for Z:

- HCL Workload Automation for Z (program number 19OP1249). Only the base product (tracker) is required.

Optional software

The following optional HCL Workload Automation for Z functions require specific IBM programs:

- Tracking resource availability requires the Resource Object Data Manager (RODM) in IBM Z NetView.
- Graphical view of jobs and their dependencies using ISPF panels requires GDDM®.
- For TCP/IP communications, it is required z/OS® Communications Server Version 2.5, or later.
- IBM Z NetView® is required to enable HCL Workload Automation for Z to schedule generic alerts as defined by that NetView® release and to specify an alert receiver ID other than the default receiver.
- User-authority-support functions require z/OS® Security Server RACF® Version 2.5, or later .
- z/OS Version 2.5 DFSMS™ with Hierarchical Storage Management component is required for the catalog management function to recall migrated data sets.
- If you are using Db2 for z/OS, the mirroring feature for the Orchestration Monitor requires that you have:
    - Installed IBM Db2 for z/OS V13.1.503, or later.
    - Installed IBM Db2 Accessories Suite for z/OS for Db2 13 for z/OS V04.02.00, or later.
    - Defined a Workload Manager (WLM) policy for the required Db2 for z/OS environment and that a Db2 procedure for the WLM Application environment is up and running.
- The Dynamic Workload Console reporting feature requires:
    - IBM Db2 for z/OS Version 12, or later.
    - IBM® Semeru Runtime Certified Edition (formerly known as IBM® 64-bit SDK for z/OS®, Java Technology Edition) V17, or later.
- The HCL Workload Automation for Z control Language tool, Workload Automation Programming Language, and Dynamic Workload Console reporting feature require either the IBM Compiler Library for REXX/370 or the IBM Alternate Library for REXX™ on zSeries®, which can be downloaded from https://www.ibm.com/support/pages/node/572469.

Related software

These IBM® licensed programs can be used with HCL Workload Automation for Z to provide comprehensive, integrated DP operations:

- IBM Z NetView® Version 6.2, or later.
- Report Management and Distribution System (RMDS) Version 2.3, or later.
- IBM Z Decision Support for z/OS® Version 1.9, or later.
- System Automation for z/OS® Version 4.2, or later.
- IBM Z Monitoring V1.1, or later
- IBM Tivoli Output Manager Version 3.1, or later.

Parts and their relationships

HCL Workload Automation for Z consists of a base product, the *agent*, and a number of features. You need the base product to track your workload. Hereafter, you see the agent referred to as the *tracker* and the engine referred to as the *Z controller*. One z/OS system in your complex is designated the *controlling* system and runs the *Z controller* feature. Only one controller feature is required, even when you want to start standby controllers on other z/OS systems in a sysplex.

You can also control the workload in other operating environments (UNIX, Windows, IBM i) using the end-to-end scheduling functions provided in HCL Workload Automation for Z and HCL Workload Automation.

The rest of this section describes the tracker and Z controller, their relationship, and how you can configure them.

Tracker

A tracker is required for every z/OS system in an HCL Workload Automation for Z configuration. The tracker handles the submission of jobs and tasks on the system, and keeps track of the status of the workload. In conjunction with standard interfaces to JES and SMF, HCL Workload Automation for Z records the relevant information about the workload by generating *event records*. The event records are captured and stored by the tracker. The tracker then communicates event information to the Z controller for further processing. The log where events are written by the tracker is called the *event data set*.

HCL Workload Automation for Z address spaces are defined as z/OS subsystems. The routines that run during subsystem initialization establish services that enable event information to be generated and stored in common storage (ECSA) even when an address space is not active.

You can optionally install a Data Store for each JES spool in a system. In a simple JES configuration this would mean one Data Store for each tracker. In systems with shared spools (for example, JES2 MAS), there is a Data Store for each spool, and there are fewer Data Stores than trackers.

Controller

The Z controller is the focal point of your HCL Workload Automation for Z configuration. It contains the controlling functions, ISPF dialogs, databases, and plans. The system that the Z controller is started on is called the HCL Workload Automation for Z controlling system. HCL Workload Automation for Z systems that communicate with the controlling system are called controlled or tracker systems. You need to install at least one Z controller for your production systems. This controls the entire HCL Workload Automation for Z configuration, the OPCplex, both local and remote.

You can use the Z controller to provide a single, consistent, control point for submitting and tracking the workload on any operating environment. HCL Workload Automation for Z provides distributed agents and open interfaces you use to integrate the planning, scheduling, and control of work units such as online transactions, file transfers, or batch processing in any operating environment that can communicate with z/OS®.

Server

HCL Workload Automation for Z provides a server you use to access the controller remotely from ISPF dialogs, PIFs, and the Dynamic Workload Console interface. Connections with the server run through Advanced Program-to-Program

Communications (APPC) sessions or Transmission Control Protocol/Internet Protocol (TCP/IP). The server runs in its own address space; however, it is optional if you do not access the controller remotely.

Using a Started Task JCL you start and stop one or more servers either individually, using the Start and Stop operator commands, or automatically with the controller, using a keyword in the OPCOPTS statement. A server must start on the same z/OS image as its controller.

The PIF dialog connection to the controller, whether via server or subsystem interface, is allowed only when the code is at the same level on both sides of the interface.

Graphical user interfaces

A graphical user interface is packaged with the product. You can use it in addition to, or in place of, ISPF:

Dynamic Workload Console

The Web-based user interface for the entire HCL Workload Automation suite of products. It is the strategic user interface for the suite and includes support for the latest functions and enhancements featured in HCL Workload Automation for Z.

The console gets access to the controller by way of the HCL Workload Automation for Z connector component, which is installed by default with the Dynamic Workload Console and is connected to HCL Workload Automation for Z by TCP/IP. For detailed information about how you install the Z connector with the Dynamic Workload Console, see Installing the Dynamic Workload Console on page 218.

The Dynamic Workload Console and HCL Workload Automation for Z connector are components that you install and run on distributed platforms, Windows™, UNIX™, Linux™, and z/OS.

One Z connector can be used to communicate with multiple Z controllers, and can serve multiple machines running Dynamic Workload Console. The graphical user interface and Z connector are installed in the installation directory of the same computer.

For detailed information about the Dynamic Workload Console, see the *Dynamic Workload Console User's Guide*.

The following additional publications are specific to the Dynamic Workload Console:

Product Requirements

Provides you with detailed information about downloading the product installation images.

Dynamic Workload Console Detailed System Requirements

A dynamically maintained document that provides detailed information about the supported platforms, hardware and software prerequisites, and supported client browsers.

Dynamic Workload Console Release Notes

A dynamically maintained document that contains the following topics:

- What is new in the release
- Interoperability tables
- Software limitations and workarounds
- Installation limitations and workarounds
- Internationalization Notes
- Documentation updates
- APARS fixed in the release

Data Store

Data Store is a separate address space. Its function is to collect structured (steps and data sets) and, optionally, unstructured (SYSOUT) information for all submitted jobs.

Data Store is required to use the following Restart and Cleanup functions:

- Restart at the job or step level
- Data set clean up
- JOBLOG retrieval

The controller can be connected to Data Store using XCF, SNA, or TCP/IP.

Output collector

The function of this started task is to collect the logs of jobs and dynamic jobs run on HCL Workload Automation Agents (z-centric) and to send them to the JES spool, where they can be taken and archived by any integrated output management product into a dedicated repository.

Output collector is sent an event (in the form of a record in an event data set) by the controller every time a job or a dynamic job completes or terminates in the z-centric environment. The event contains the information necessary for the output collector to identify the job and the agent that ran it. The output collector then retrieves the job log from the agent (or the dynamic domain manager if the job is dynamic) and copies it to a SYSOUT in JES to make it available to an output management product.

In addition, Output collector attaches a header at the top of the job log with information that classifies the output (occurrence name, occurrence IA, job name, workstation name, operation number, start time, end time) and information about the run (process ID, return code, duration, status, hostname).

Activation of this feature is optional. If you activate it, it automatically collects the logs of all jobs run in the z-centric environment, regardless of whether they complete successfully or terminate in error. If you do not activate it, you can still configure your system to either request logs manually or to receive those of jobs ended in error.

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

For a detailed description of the Output collector, see *Scheduling End-to-end with z-centric Capabilities*.

Configurations

You can configure HCL Workload Automation for Z to control virtually any combination of operating environments. HCL Workload Automation for Z can automatically schedule, submit, and track batch jobs, started tasks, and write-to-operator (WTO) messages. You can also use it to coordinate manual activities in your production workload.

Your configuration can include:

- A controlling system
- Controlled systems:
    - Local and remote controlled z/OS systems, including a parallel sysplex.
    - Standby Z controller systems.
    - Previous HCL Workload Automation for Z releases.
    - Controlled systems running on distributed agents.
    - Other operating environments that do not support the distributed agents.

For an explanation of the connections between HCL Workload Automation for Z systems and some examples of configurations, see Planning your configuration on page 24

Controlling system

A controlling system requires both a tracker and Z controller. If you install only one system, this is the controlling system. The controlling system can communicate with controlled z/OS systems using the Transmission Control Protocol/Internet Protocol (TCP/IP), cross-system coupling facility (XCF), and network communication function (NCF).

Controlled systems

A controlled z/OS system requires a tracker. Communication with the controlling system is through TCP/IP, XCF, or NCF. The tracker writes event records to an event data set, and transfers the records to the controlling system if connected using TCP/IP, XCF, or NCF. NCF uses ACF/VTAM to link HCL Workload Automation for Z systems.

If you use XCF for communication, you can include a *standby Z controller* on one or more controlled systems. A standby Z controller is started in its own address space. It can take over the functions of the Z controller if z/OS fails or if the Z controller itself fails. It cannot perform the functions of a tracker while in standby mode.

The Z controller also controls the workload in the end-to-end scheduling with z-centric capabilities environment.

For detailed information about how to control other operating environments, see *Customization and Tuning*.

Integration with HCL Workload Automation

Integration with HCL Workload Automation is provided by activating the end-to-end scheduling with z-centric capabilities feature. This feature is designed to let you schedule and control workload from the mainframe to distributed systems through z-centric agents, in a very simple architecture of the end-to-end scheduling framework. HCL Workload Automation for Z becomes the single point of control, providing you with all the mainframe capabilities to manage distributed workload. Communication between the z-centric agents and HCL Workload Automation for Z controller is direct, through the HTTP or HTTPS protocol. This feature enables dynamic scheduling of jobs (by connecting a dynamic domain manager to the controller) as well as scheduling of job types with advanced options (file transfer, database, web services, J2EE, and more).

For detailed information about the z-centric end-to-end scheduling environment, see *Scheduling End-to-end with z-centric Capabilities*.

Subtasks

An HCL Workload Automation for Z address space (subsystem) consists of many z/OS subtasks. Some of these subtasks are always attached when the subsystem is started, others are conditionally attached according to initialization parameters specified for the scheduler options (OPCOPTS) statement in the HCL Workload Automation for Z parameter library.

When a Z controller is started in standby mode, only the HCL Workload Automation for Z main task (EQQMAJOR) is started. The subtasks that comprise an active Z controller are attached when a takeover is performed.

Table 1: HCL Workload Automation for Z subtasks on page 19 describes the subtasks.

Table 1. HCL Workload Automation for Z subtasks

Subtask ID	Component code	Description	Activated by	Function
APPC	PP	APPC functions	OPCOPTS APPCTASK(YES)	Starts APPC support
AR	AR	Automatic recovery	OPCOPTS RECOVERY(YES)	Manages failing operations
CPH	CPH	Critical path handler	Always activated	Updates the critical job table
DBF	DF	DB Filler	FEDERATOR parameter of MIRROPTS	Replicates CP data to the database referenced by the Federator
DRT	DX	Data router	Always activated	Routes data to other subtasks or HCL Workload Automation for Z subsystems
EMGR	EM	Event manager	OPCOPTS OPCHOST(YES)	Processes job-tracking events
ERDR	ER	Event reader	OPCOPTS ERDRTASK(*n*)	Reads events from an event data set
EWTR	EW	Event writer	OPCOPTS EWTRTASK(YES)	Writes events to an event data set
EXA	EX	External router	OPCOPTS OPCHOST(YES)	Calls EQQUX009 to route submit requests to a user-defined destination ID
FL	FL	Fetch joblog	OPCOPTS RCLEANUP(YES)	Retrieves JOBLOG information

Table 1. HCL Workload Automation for Z subtasks (continued)

Subtask ID	Component code	Description	Activated by	Function
GEN	GS	General service	OPCOPTS OPCHOST(YES)	Processes HCL Workload Automation for Z dialog requests
HTC	HTC	HTTP client	HTTP parameter of ROUTOPTS	Manages communications with z-centric agents through the HTTP or HTTPS protocol
HTS	HTS	HTTP client	HTTP parameter of ROUTOPTS	Listens for inbound requests from the z-centric agent
ID	ID	TCP/IP Data Store	TCPDEST parameter of FLOPTS	Manages communications with TCP/IP-connected Data Stores
IP	IP	TCP/IP tracker	TCPIP parameter of ROUTOPTS	Manages communications with TCP/IP-connected standard trackers
JCC	JC	Job completion checker	OPCOPTS JCCTASK(YES)	Scans SYSOUT data sets
JLA	JL	JT and DB logs archiver	OPCOPTS OPCHOST(YES)	Copies JT and DB logs to the archive data set (EQQJTARC and EQQDBARC, respectively)
NMM	NM	Normal mode manager	OPCOPTS OPCHOST(YES)	Maintains the current plan
PSU	PS	Pre-SUBMIT tailoring	OPCOPTS RCLEANUP(YES)	Tailors the JCL before submitting it by adding the EQQCLEAN pre-step
RODM	RM	RODM support	OPCOPTS RODMTASK (YES)	Starts RODM support
SUB	SU	Submit task	Always activated	Initiates work (job submit, job release, and WTO and STC operations)
VTAM®	CB	Network communication function (NCF)	OPCOPTS NCFTASK(YES)	Transmits and receives HCL Workload Automation for Z data through a VTAM® link
WSA	WA	Workstation analyzer	OPCOPTS OPCHOST(YES)	Schedules work for processing

Table 1. HCL Workload Automation for Z subtasks (continued)

Subtask ID	Component code	Description	Activated by	Function
Note: The subtask ID is the same identifier used to control the subtask using the z/OS MODIFY command.				

Relationship between HCL Workload Automation for Z and the z/OS system

HCL Workload Automation for Z is a z/OS subsystem, initialized during IPL. Routines run during subsystem initialization establish basic services, such as an event queue in ECSA. HCL Workload Automation for Z uses standard interfaces to SMF and JES to gather relevant information about the workload on the z/OS system.

The functions of the Z controller are available when an address space has been created for it, and the required subtasks have been successfully initialized. The Z controller can run either as a started task or as a batch address space. Normally, the address space is started during the IPL process, that is by a z/OS start command in COMMND*nn*, or by console automation. Alternatively, a z/OS operator can issue a `START` command from the operator console. The z/OS operator can also stop or modify the address space, using the `STOP` and `MODIFY` commands.

A TSO user accesses HCL Workload Automation for Z services using the dialogs. A dialog is a sequence of ISPF panels. Many of the functions supported by the dialogs pass service requests from the TSO user's address space to the Z controller address space for processing.

Before performing any function you request, the dialog function passes the request to the system authorization facility (SAF) router. If RACF®, or a functionally equivalent security product, is installed and active on the z/OS system, the SAF router passes the verification request to RACF® to perform this authority check.

A typical dialog service request is to access one or more records in VSAM files that are maintained and controlled by HCL Workload Automation for Z. Such a request is passed to HCL Workload Automation for Z through the z/OS subsystem interface (SSI). This interface invokes a routine that resides in common storage. This routine must be invoked in APF-authorized mode.

Consider that all long term plan (LTP) and CP batch planning jobs have to be excluded from SMARTBATCH DA (Data Accelerator) processing. When the SMARTBATCH DATA ACCELERATOR is used with the scheduler LTP and CP batch planning jobs, the normal I/O to EQQCKPT is delayed until END OF JOB (or at least END OF JOBSTEP). This interferes with the normal exchange of data between the batch job and the controller started task so that when the batch job signals the controller to check the EQQCKPT to determine whether a new current plan has been created, the required updates to the CKPT have not yet been made. This causes the controller to conclude that no NCP has been created, and no turnover processing is done. As a result, even if the plan jobs run successfully, the NCP is not taken into production by the controller unless a CURRPLAN(NEW) restart is performed.

The Data Store uses the MVS/JES SAPI functions to access sysout data sets, allowing concurrent access to multiple records from a single address space.

Batch optimizer utilities, such as BMC Batch Optimizer Data Optimizer and Mainview Batch Optimizer, prevent correct communication between the scheduler's controller and CP/LTP batch planning jobs. The scheduler's logic depends on an exchange of enqueues and real-time updates of several sequential data sets to pass information back and forth between the controller's STC and the CP/LTP batch planning jobs. These optimizers hold I/O from the batch jobs until END OF STEP or END OF JOB, then preventing the required communication from taking place. When such utilities are allowed to "manage" I/O for the scheduler's CP or LTP batch planning jobs, communication between the jobs and the controller is disrupted. This causes numerous problems that are hard to diagnose. Most commonly, the CURRENT PLAN EXTEND or REPLAN jobs will run to normal completion, and an NCP data set will be successfully created, but the controller will fail to automatically take the new plan into production until it is forced to do so via a CURRPLAN(NEW) restart of the CONTROLLER. Use of BATCHPIPES with these batch planning jobs will result in the same sorts of problems.

Using the Program Directory

The *Program Directory* contains instructions for downloading the product and might include technical information that is more recent than the information provided in this publication.

In addition, the *Program Directory* describes the program temporary fix (PTF) level of the HCL Workload Automation for Z licensed program that you receive.

Sample library

SEQQSAMP is a library included on the distribution CD containing samples of exits, application programs, and the job control language (JCL). You can use the samples for specific installation tasks. Sample library (SEQQSAMP) on page 343 describes the members of the SEQQSAMP library. Familiarize yourself with the contents of the SEQQSAMP library before you begin installation.

The installation process

Table 2: Stages summarizing the HCL Workload Automation for Z installation process on page 22 shows the various stages to install HCL Workload Automation for Z.

After you have installed the product, you might want to include more functions. For details, see *HCL Workload Automation for Z: Customization and Tuning*.

Table 2. Stages summarizing the HCL Workload Automation for Z installation process

Stage	Description	For more information ...
1	Plan your configuration. You might create a diagram of your own HCL Workload Automation for Z configuration to refer to during the installation process.	Planning your configuration on page 24 gives examples of common configurations.
2	Plan your product installation.	Planning your installation on page 48 describes considerations for installing HCL Workload Automation for Z and provides a checklist for the installation tasks.
3	Install the product.	Installing on page 60 describes the installation tasks in detail.

Table 2. Stages summarizing the HCL Workload Automation for Z installation process (continued)

Stage	Description	For more information ...
4	Verify your installation.	Verifying your installation on page 170 describes how you can verify that HCL Workload Automation for Z is correctly installed.

Chapter 2. Planning your configuration

When planning the configuration for your installation, consider the areas explained in the following topics, which describe the connections between the HCL Workload Automation for Z systems and provide some examples of basic configurations.

HCL Workload Automation for Z must recognize when events occur, for example when a started task or job begins to run or terminates, or when a data set has been printed. It uses JES and SMF exits to obtain this information from z/OS and to create *event records* describing the changes in the system.

The event records are stored in a sequential file called the *event data set* identified by the EQQEVDS DD name.

HCL Workload Automation for Z also uses the event data set to write checkpoint information for submission requests. The first record of the data set is used for this purpose, so the EQQEVDS DD name must be specified for all HCL Workload Automation for Z address spaces. The same data set can be used for both submit checkpointing and the event-writer subtask.

Trackers

A tracker must be installed on every z/OS system that you want HCL Workload Automation for Z to control. The tracker on each system writes events to the event data set. A subtask of the tracker, called the *event writer* performs this function. For the current plan to be updated, the event information must be communicated to, and processed by, the Z controller. The events are routed to the Z controller through the connection that links the tracker and the Z controller.

Initialization statements

HCL Workload Automation for Z initialization statements specified in the parameter library describe, among other things, the configuration of your installation.

When a tracker is connected with the Z controller (that is, through an XCF, NCF, or TCP/IP connection) or the tracker and Z controller are running in the same address space, the event writer can be used to forward events directly to the Z controller.

When an event writer is started with the EWSEQNO keyword on the EWTROPTS initialization statement, the event writer logs event information on the event data set and adds the event concurrently to the data router queue. The event is *not* read back from the event data set each time, as it is by an event reader subtask. In this configuration, events are only read back from DASD if they need to be resent to the Z controller during restart processing (for example when the communication link to the Z controller becomes active after an outage).

For more information about the ERDROPTS and EWTROPTS initialization statements, see *Customization and Tuning*. For detailed information about allocating event data sets, see also Event data sets (EQQEVDS, EQQEVDnn, and EQQHTTP0) on page 128.

Communication

The data router subtask is responsible for communicating the event to the Z controller event manager subtask, either by XCF, NCF, TCP/IP or by adding directly to the queue when the tracker and Z controller are started in the same address space. This eliminates the need for a separate event-reader function, saves time, and saves I/O operations.

The EWSEQNO value is not used to build a DD name, as happens with the event reader subtask. The event writer uses the EQQEVDS DD name to identify the event data set.

If a connection is lost between a tracker and the Z controller, the event writer continues to record events. When the connection is restored, the event data set is processed from the last event received by the Z controller before the outage.

Note: Controllers scheduling work (for a given z/OS image) must have unique subsystem names.

How to connect HCL Workload Automation for Z systems

HCL Workload Automation for Z systems can be connected by using any of the following methods.

- TCP/IP link
- XCF communication links
- VTAM® link

The controller uses any of these methods to transmit work to a tracker system. The tracker system uses the same connection to transmit events back to the Z controller.

Distributed agents communicate with the controller by using TCP/IP services.

TCP/IP

HCL Workload Automation for Z uses Transmission Control Protocol/Internet Protocol (TCP/IP) to connect a tracker to the controller using a TCP/IP link.

The Z controller transmits work to the tracker through TCP/IP, and the same connection is used to pass back event information. The scheduler uses TCP/IP also to connect a distributed agent to the server. The TCP/IP connection between the server and the clients is established by the server.

z/OS cross-system coupling facility

HCL Workload Automation for Z uses the z/OS cross-system coupling facility (XCF) to connect HCL Workload Automation for Z systems using XCF communication links.

When one or more trackers are connected to the Z controller through XCF communication links, the HCL Workload Automation for Z systems form an XCF group. The systems use XCF group, monitoring, and signaling services to communicate. The Z controller submits work and control information to the trackers using XCF signaling services. The trackers use XCF services to transmit events back to the Z controller.

XCF connections let HCL Workload Automation for Z support a hot standby Z controller and automatic-workload-restart functions.

VTAM® (network communication function)

HCL Workload Automation for Z uses the network communication function (NCF) to connect a tracker to the Z controller using a VTAM link. The Z controller transmits work to the tracker through NCF, and the same connection is used to pass back event information.

Workstation destination

The various physical and logical locations where tasks are performed at your installation are represented in HCL Workload Automation for Z by workstations. Each workstation groups related activities. Every operation in the application description database and the current plan is associated with a workstation. You define workstations in the workstation description database.

The destination field is one attribute of a workstation description. It identifies the system in your configuration to which the operations scheduled for this workstation should be submitted. The field can contain an XCF member name, a VTAM LU name, an IP address or host name and port, or a user-defined destination.

If the destination field is not blank, the same name must also be present in the SNA, TCPIP, USER, XCF, HTTP, or HTTPS keyword of the ROUTOPTS statement, depending on the connection method.

The destination field can also remain blank. A blank destination field means that operations at this workstation will be submitted by the Z controller.

The operation-initiation exit, EQQUX009, manages the routing of the workload to user-defined destinations.

Workload restart

You can use workload restart (WLR) to restart and reroute work in your HCL Workload Automation for Z configuration. WLR tracks the status of workstations. It can be invoked when a workstation becomes inactive; that is, when the Z controller cannot communicate with the tracker at the destination that the workstation represents.

If an operation is *restartable*, it can be started again after a workstation failure. If an operation is *reroutable*, it can be moved to an alternative workstation for running when its workstation is no longer active.

For WLR purposes, the status of a workstation can be either active or inactive. An inactive workstation has a status of offline, failed, or unknown. The actions that WLR performs depend on the new status of the workstation and on the values you specify on the WSFAILURE and WSOFFLINE keywords of the JTOPTS initialization statement. The inactive status that a workstation can have depends on the type of connection between the tracker and the Z controller. The connection type and the new workstation status determine whether workload restart actions can be invoked automatically. You can use the full capabilities of WLR on systems that are connected by XCF.

Note: JES also has restart functions, which can be used when the system is restarted after a failure. JES can restart jobs that were active when the failure occurred. To prevent jobs from being started twice, ensure that both JES and WLR do not perform restart actions for jobs on the failing system.

JES considerations

The JES type and configuration in your installation has implications on your HCL Workload Automation for Z configuration.

Consider the following situations in a JES type configuration:

1. On systems where JES2 is installed, an HCL Workload Automation for Z Tracker must be installed on each system in the JES2 Multi-Access Spool (MAS) complex.
2. If you do not install HCL Workload Automation for Z on all systems in a JES3 complex, ensure that:
   - A tracker is installed on the global.
   - Jobs are submitted, whether by HCL Workload Automation for Z or outside HCL Workload Automation for Z, to a system where a tracker is installed. Use the //*MAIN SYSTEM=*sysid* statement in the JCL, or start job classes used by these jobs only on those systems where a tracker is installed.
   - If you track print operations, output is printed only on those systems where a tracker is installed.

Basic configuration examples

The examples in this topic shows the configurations using the various connection methods. They are based on a single-image z/OS environment. For examples of more complex configurations, see Configuration examples.

The examples in this topic show:

- All HCL Workload Automation for Z address spaces as Version 2 subsystems.
- Sample initialization statements that you can use to create the configuration. Only initialization statements that specifically relate to the configuration are included.
- The HCL Workload Automation for Z components that are required, the flow of automatic work submission, and event collection in various system combinations.

TCP/IP connected

Figure 1: A z/OS system with a TCP/IP connection on page 28 shows two HCL Workload Automation for Z address spaces with a TCP/IP connection on a z/OS system.

You represent this system to the scheduler by defining a computer workstation with a destination field that specifies the destination name of the tracker. The Z controller transmits JCL, release commands, WTO messages, and cleanup requests across the TCP/IP link. The tracker receives data across the TCP/IP link and performs the following actions:

- Submits JCL for batch jobs to the JES internal reader
- Writes the JCL for started tasks into the EQQSTC data set and issues `START procname` z/OS commands
- Issues JES release commands for jobs in HOLD status
- Submits the cleanup job.

The event-tracking routines create event records to describe activities that occur on the system. These records are added to the tracker event writer queue in ECSA. The tracker processes the queue, transmits the records to the Z controller across the TCP/IP link, and writes the events into the event data set. The IP task in the Z controller receives the event records, and the current plan is updated.

Note: You must specify EQQEVDS for a controller, even if an event writer is not started in the controller address space. The EQQEVDS data set is used for submit checkpointing. It can be the same data set that is used by an event-writer function. Use a unique EQQEVDS for each address space of the scheduler.

Example

Figure 1. A z/OS system with a TCP/IP connection

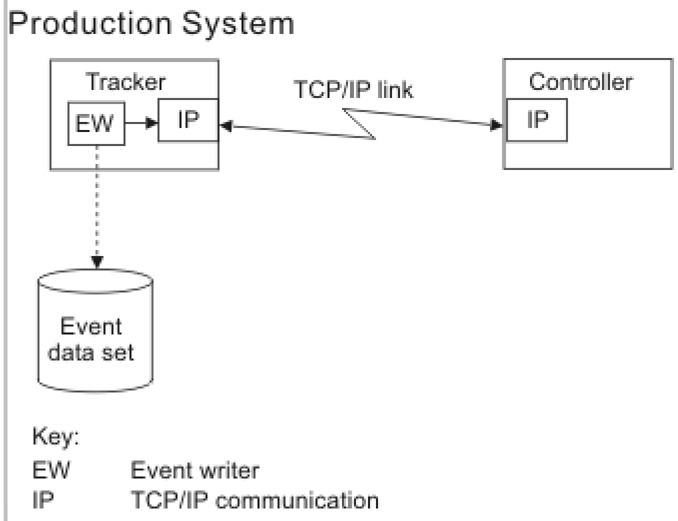


Table 3: Example EQQPARM members for Figure 4 on page 28 shows the initialization statements you can use to create the configuration in Figure 1: A z/OS system with a TCP/IP connection on page 28.

Table 3. Example EQQPARM members for Figure 4

Members for the Z controller	Members for the tracker

OPCECNT
`OPCOPTS  OPCHOST(YES)`
`         ERDRTASK(0)`
 	
`ROUTOPTS TCPIP(DEST1:'1.111.111.111'/4444)`
 	
`TCPOPTS  TCPIPJOBNAME('TCPIP')`
`         HOSTNAME('9.12.134.1')`
`         TRKPORTNUMBER(8888)`
	TRKA
	`OPCOPTS OPCHOST(NO)`
	`        ERDRTASK(0)`
	`        EWTRTASK(YES)`
	`TRROPTS HOSTCON(TCP)`
	`        TCPHOSTNAME('9.12.134.1')`
	`        TCPPORTNUMBER(8888)`
	`TCPOPTS TCPIPJOBNAME('TCPIP')`
	`        HOSTNAME('1.111.111.111')`
	`        TRKPORTNUMBER(4444)`
	
	STDEWTREWTROPTS EWSEQNO(01)

Note: In this example, the name of the destination is DEST1. The destination name is defined also in the destination field of the workstation.

XCF connected

Figure 2: A z/OS system with an XCF connection on page 29 shows two HCL Workload Automation for Z address spaces with an XCF connection in a z/OS monoplex.

You represent this system to HCL Workload Automation for Z by defining a computer workstation with a destination field that specifies the XCF member name of the tracker. The Z controller uses XCF services to transport JCL, release commands, WTO messages, and cleanup requests to members in the sysplex. The tracker receives data from XCF and performs the following actions:

- Submits JCL for batch jobs to the JES internal reader
- Writes the JCL for started tasks into the EQQSTC data set and issues `START procname` z/OS® commands
- Issues JES release commands for jobs in HOLD status
- Submits the cleanup job.

The event-tracking routines create event records to describe activities that occur on the system. These records are added to the tracker event writer queue in ECSA. The tracker processes the queue, transports the records to the Z controller across the XCF link, and writes the events into the event data set. The data router subtask in the Z controller receives the event records from XCF, and the current plan is updated.

Example

Figure 2. A z/OS system with an XCF connection

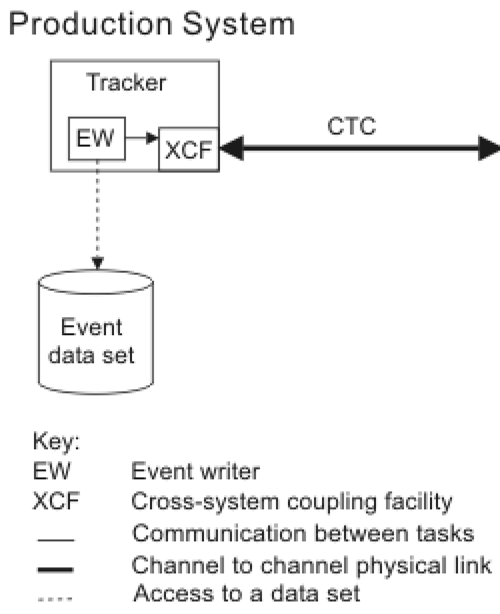


Table 4: Example EQQPARM members for Figure 5 on page 30 shows the initialization statements you can use to create the configuration in Figure 2: A z/OS system with an XCF connection on page 29.

Table 4. Example EQQPARM members for Figure 5

Members for the Z controller	Members for the tracker

OPCECNT
`OPCOPTS  OPCHOST(YES)`
`         ERDRTASK(0)`
`ROUTOPTS XCF(OPCTRK)`
`XCFOPTS  MEMBER(OPCCNT)`
`         GROUP(PLEXSYSA)`
	TRKA
	`OPCOPTS OPCHOST(NO)`
	`        ERDRTASK(0)`
	`        EWTRTASK(YES)`
	`        EWTRPARM(STDEWTR)`
	`TRROPTS HOSTCON(XCF)`
	`XCFOPTS MEMBER(OPCTRK)`
	`        GROUP(PLEXSYSA)`

	STDEWTR `EWTROPTS EWSEQNO(01)`

Note: In this example, the name of the monoplex is PLEXSYSA. The members in that group are:

OPCCNT

The controller

OPCTRK

The tracker

The tracker member name is defined in the destination field of the workstation.

VTAM® connected

Figure 3: A z/OS system with a VTAM connection on page 31 shows two HCL Workload Automation for Z address spaces with a VTAM® connection on a z/OS system.

You represent this system to HCL Workload Automation for Z by defining a computer workstation with a destination field that specifies the LU name of the tracker. The Z controller transmits JCL, release commands, WTO messages, and cleanup requests across the LU-LU link using the NCF component. The tracker receives data across the VTAM® link and performs the following actions:

- Submits JCL for batch jobs to the JES internal reader
- Writes the JCL for started tasks into the EQQSTC data set and issues `START` *`procname`* z/OS commands
- Issues JES release commands for jobs in HOLD status
- Submits the cleanup job.

The event-tracking routines create event records to describe activities that occur on the system. These records are added to the tracker event writer queue in ECSA. The tracker processes the queue, transmits the records to the Z controller across the VTAM® link, and writes the events into the event data set. The VTAM® subtask in the Z controller receives the event records, and the current plan is updated.

Note: You must specify EQQEVDS for a controller, even if an event writer is not started in the controller address space. The EQQEVDS data set is used for submit checkpointing. It can be the same data set that is used by an event-writer function. Use a unique EQQEVDS for each address space of the scheduler.

Example

Figure 3. A z/OS system with a VTAM® connection

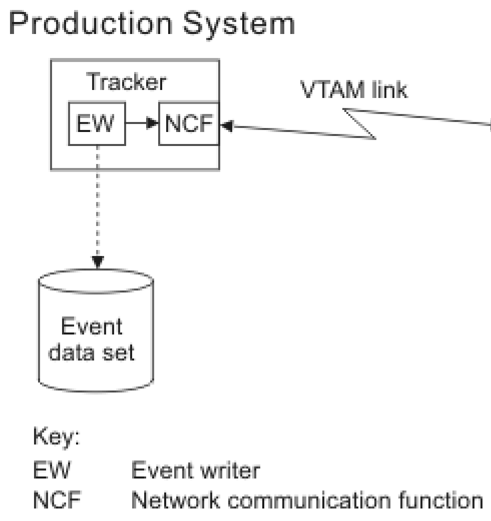


Table 5: Example EQQPARM members for Figure 3 on page 31 shows the initialization statements you can use to create the configuration in Figure 3: A z/OS system with a VTAM connection on page 31.

Table 5. Example EQQPARM members for Figure 3

Members for the Z controller	Members for the tracker

OPCECNT
`OPCOPTS  OPCHOST(YES)`
`         ERDRTASK(0)`
`         NCFTASK(YES)`
`         NCFAPPL(CNTSYS)`
`ROUTOPTS SNA(TRKSYS)`
	TRKA
	`OPCOPTS OPCHOST(NO)`
	`        ERDRTASK(0)`
	`        EWTRTASK(YES)`
	`        EWTRPARM(STDEWTR)`
	`        NCFTASK(YES)`
	`        NCFAPPL(TRKSYS)`
	`TRROPTS HOSTCON(SNA) SNAHOST(CNTSYS)`

	STDEWTR

Table 5. Example EQQPARM members for Figure 3 (continued)

Members for the Z controller	Members for the tracker
	`EWTROPTS EWSEQNO(01)`

Note: In this example, the LU name of the Z controller is CNTSYS and the tracker uses TRKSYS. The tracker LU is defined in the destination field of the workstation.

Tracker and controller in a single address space

Figure 4: A tracker and controller configured in a single address space on page 32 shows one HCL Workload Automation for Z address space performing the function of both the tracker and the Z controller. To optimize availability, do not use this configuration in your production environment. However, at least one of your HCL Workload Automation for Z test environments will probably use this configuration.

You represent this system to HCL Workload Automation for Z by defining a computer workstation with a blank destination field. The submit subtask performs the following actions:

- Submits JCL for batch jobs to the JES internal reader
- Writes the JCL for started tasks into the EQQSTC data set and issues `START` *`procname`* z/OS commands
- Issues JES release commands for jobs in HOLD status

The event-tracking routines create event records to describe activities that occur on the system. These records are added to the subsystem event writer queue in ECSA. The event writer subtask processes the events and:

- Adds the event to the data router queue, and the current plan is updated
- Writes the events into the event data set.

Example

Figure 4. A tracker and controller configured in a single address space

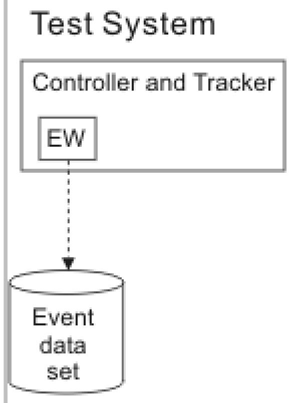


Table 6: Example EQQPARM members for Figure 6 on page 33 shows initialization statements to create the configuration in Figure 4: A tracker and controller configured in a single address space on page 32.

Table 6. Example EQQPARM members for Figure 6

EQQPARM members for the address space	

OPCECNT
`OPCOPTS  OPCHOST(YES)`
`         ERDRTASK(0)`
`         EWTRTASK(YES)`
`         EWTRPARM(STDEWTR)`
	STDEWTR
	`EWTROPTS EWSEQNO(01)`

Configuration examples contains HCL Workload Automation for Z configuration examples for more complex environments.

Basic data store configuration examples

You need to install a Data Store for each spool tracked by HCL Workload Automation for Z in the configuration.

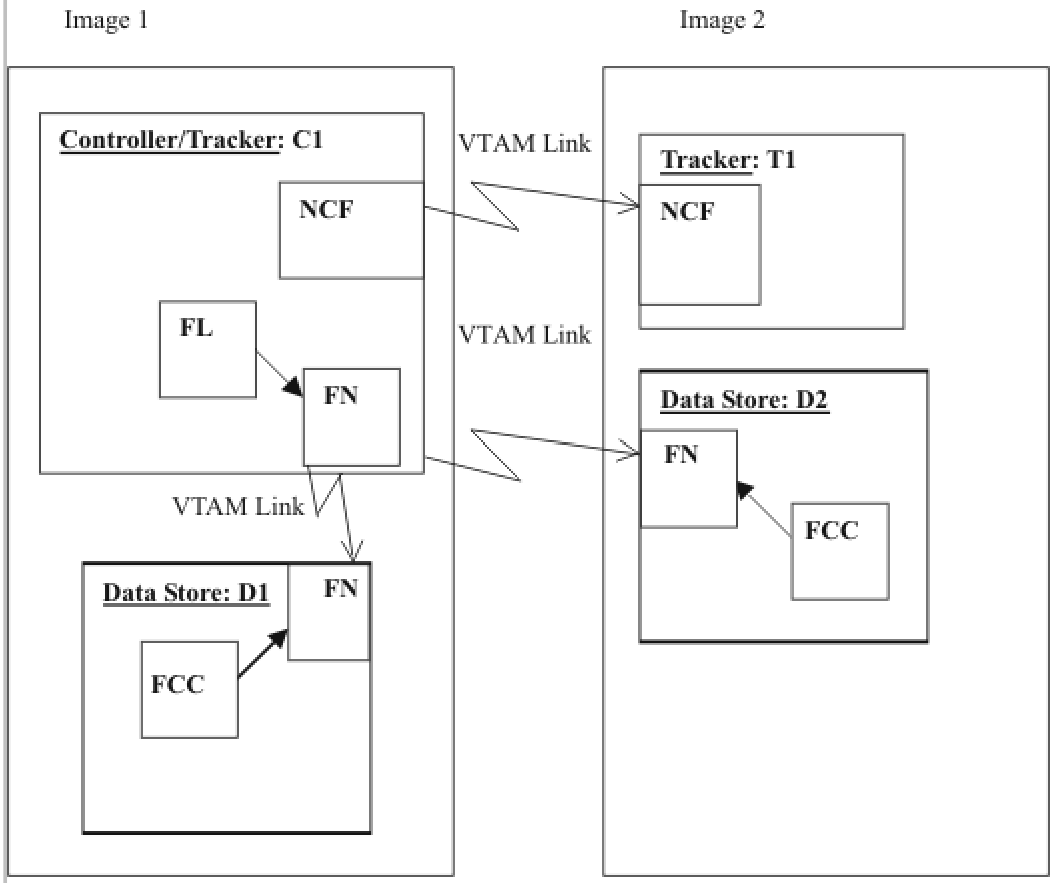
If you have a shared spool, for example, JES2 MAS, you can have a single Data Store for multiple trackers. Three kinds of controller-Data Store connections are supported: SNA, XCF, and TCP/IP. The Data Store type must be defined either as SNA, XCF, or TCP/IP. The same controller can connect, at the same time, with more than one Data Store, using a different connection type for each Data Store. Note that you need 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.

SNA-only connection

Figure 5: Controller and tracker in same address space with tracker connected through SNA on page 34 shows a JES2 with two images. In Image 1, the controller and tracker are in the same address space. Image 2 contains a tracker. The spool is not shared. Two Data Stores are required, one for Image 1 and one for Image 2. All connections are VTAM links.

Example

Figure 5. Controller and tracker in same address space with tracker connected through SNA



Key:

FCC

Data Store Communication task

FL

Fetch Job Log Task

FN

Data Store SNA handler task

NCF

Network Communication function

Table 7: Example members for Figure 7 on page 35 shows the initialization statements you can use to create the configuration in Figure 5: Controller and tracker in same address space with tracker connected through SNA on page 34.

Table 7. Example members for Figure 7

Controller member	Tracker member
C1	T1

`OPCOPTS`
`RCLEANUP(YES)`
`NCFTASK(YES)`
`NCFAPPL(LU00C1T)`
 	
`FLOPTS`
`CTLLUNAM(LU00C1D)`
`SNADEST(********.LU000D1,`
`LU000T1.LU000D2)`
 	
`ROUTOPTS SNA(LU000T1)`
	`OPCOPTS`
	`NCFTASK(YES)`
	`NCFAPPL (LU000T1)`
	
	`TRROPTS`
	`HOSTCON(SNA)`
	`SNAHOST(LU00C1T)`

D1	D2

`DSTOPTS`
`DSTLUNAM(LU000D1)`
`CTLLUNAM(LU00C1D)`
	`DSTOPTS`
	`HOSTCON(SNA)`
	`DSTLUNAM(LU000D2)`
	`CTLLUNAM(LU00C1D)`

Data Store members	

Note: In this example, the LU names for the communication partners are the following:

LU00C1D

Controller C1, when communicating with a Data Store.

LU000D1

Data Store D1.

LU000D2

Data Store D2.

LU00C1T

Controller C1, when communicating with tracker T1.

LU000T1

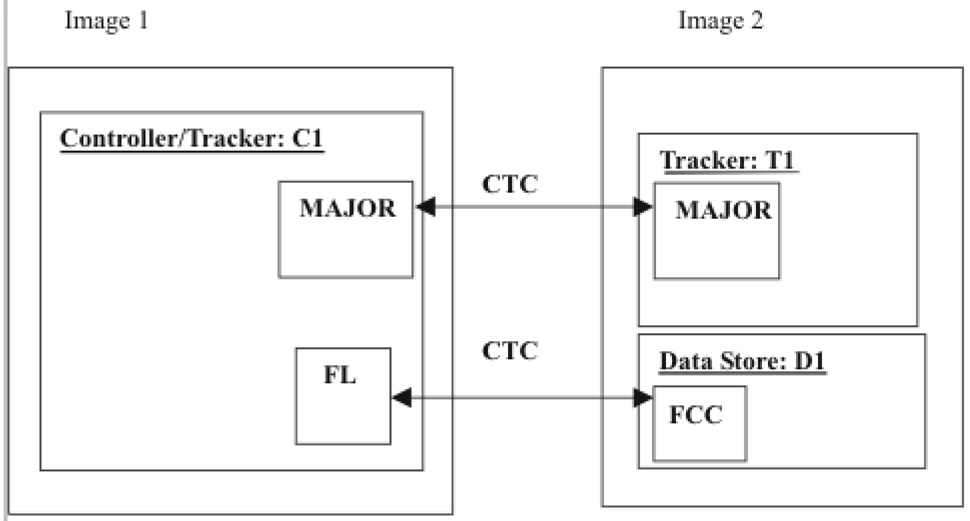
Tracker T1.

XCF-only connection

Figure 6: Controller, tracker, and Data Store connected through XCF on page 36 shows a JES2 MAS (shared spool) with two images. In Image 1, the controller and a tracker are in the same address space and connected via XCF. Image 2 contains another tracker. You need only one Data Store, which is installed in Image 2. The controller will request the Job Log from the Data Store using the FL subtask.

Example

Figure 6. Controller, tracker, and Data Store connected through XCF



Key:

FL

Fetch Job Log task

FCC

Data Store Communication task

MAJOR

Controller/tracker main task

Table 8: Example members for Figure 8 on page 36 shows the initialization statements you can use to create the configuration in Figure 6: Controller, tracker, and Data Store connected through XCF on page 36.

Table 8. Example members for Figure 8

Controller member	Tracker member

C1
`OPCOPTS`
` RCLEANUP(YES)`
` NCFTASK(NO)`
 	
`ROUTOPTS`
` XCF(XCFMEMT1)`
 	
`XCFOPTS`
` GROUP(XCFGRUCT)`
` MEMBER(XCFMEMCT)`
 	
`FLOPTS`
` DSTGROUP(XCFGRUCD)`
	T1
	`OPCOPTS`
	` NCFTASK(NO)`
	
	`TRROPTS`
	` HOSTCON(XCF)`
	
	`XCFOPTS`
	` GROUP(XCFGRUCT)`
	` MEMBER(XCFMEMT1)`

Table 8. Example members for Figure 8 (continued)

Controller member	Tracker member

`CTLMEM(XCFMEMCD)`
`XCFDEST(********.XCFMEMD1,`
` XCFMEMT1.XCFMEMD1)`
 	
Data Store member
 	
D1
 	
`DSTOPTS`
` HOSTCON(XCF)`
` DSTGROUP(XCFGRUCD)`
` DSTMEM(XCFMEMD1)`	

Note: In this example, the XCF groups for the communication partners are the following:

XCFGRUCD

The XCF group for the communication between controller and Data Store. The members in the group are:

XCFMEMCD

The controller.

XCFMEMD1

The Data Store.

XCFGRUCT

The XCF group for the communication between controller and tracker. The members in the group are:

XCFMEMCT

The controller.

XCFMEMT1

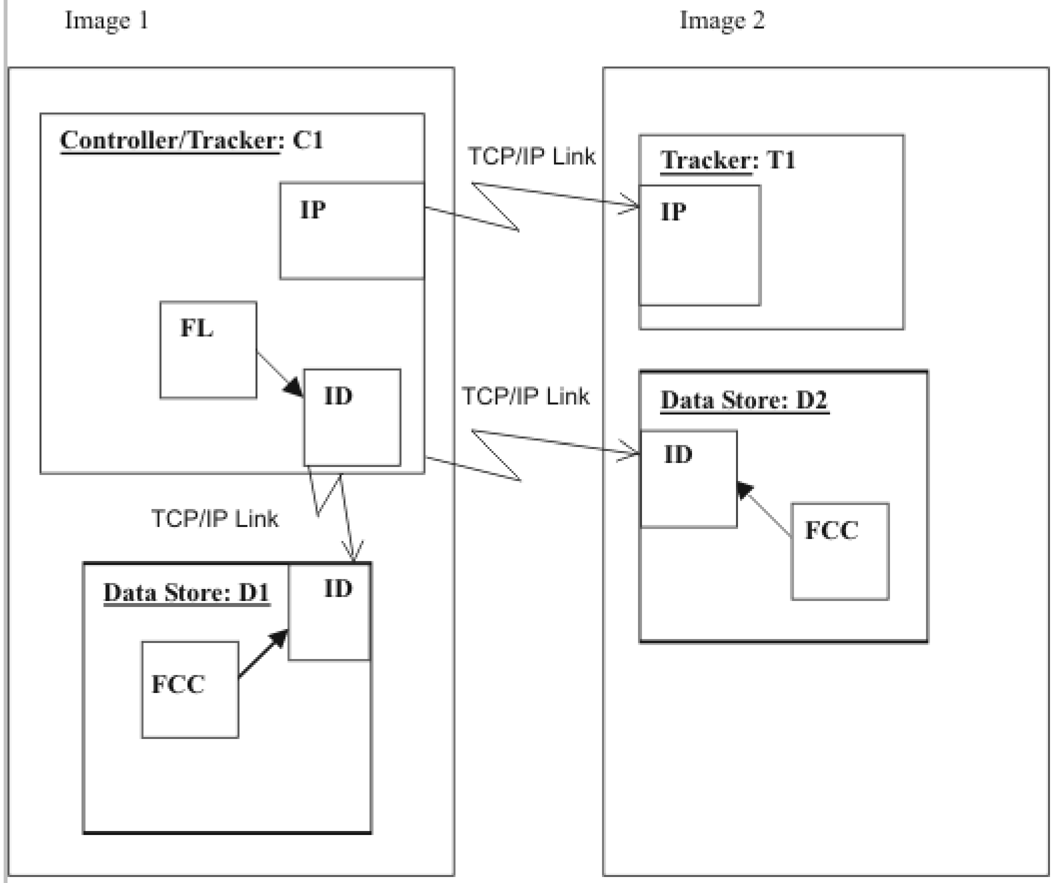
The tracker.

TCP/IP-only connection

Figure 7: Controller and tracker in same address space with tracker connected through TCP/IP on page 38 shows a JES2 with two images. In Image 1, the controller and tracker are in the same address space. Image 2 contains a tracker. The spool is not shared. Two Data Stores are required, one for Image 1 and one for Image 2. All connections are TCP/IP links.

Example

Figure 7. Controller and tracker in same address space with tracker connected through TCP/IP



Key:

FCC

Data Store Communication task

FL

Fetch Job Log Task

ID

Task for Data Store-to-controller TCP/IP communication

IP

Task for tracker-to-controller TCP/IP communication

Table 9: Example members for Figure 9 on page 39 shows the initialization statements you can use to create the configuration in Figure 7: Controller and tracker in same address space with tracker connected through TCP/IP on page 38.

Table 9. Example members for Figure 9

Controller member	Tracker member

C1
`OPCOPTS`
` RCLEANUP(YES)`
 	
`FLOPTS`
`  TCPDEST(********.'9.12.134.1',`
`           '9.12.134.9')`
 	
`ROUTOPTS TCPIP(TRK1:'9.12.134.9')`
	T1
	`TRROPTS`
	` HOSTCON(TCP)`
	` TCPHOSTNAME('9.12.134.1')`

Data Store members	

D1
`DSTOPTS`
` HOSTCON(TCP)`
` CTLHOSTNAME('9.12.134.1')`
	D2
	`DSTOPTS`
	` HOSTCON(TCP)`
	` CTLHOSTNAME('9.12.134.1')`

Note: In this example, the name of the tracker destination is TRK1. The destination name is defined also in the destination field of the workstation. The TCP/IP address of image 1 is 9.12.134.1 and the TCP/IP address of image 2 is 9.12.134.9.

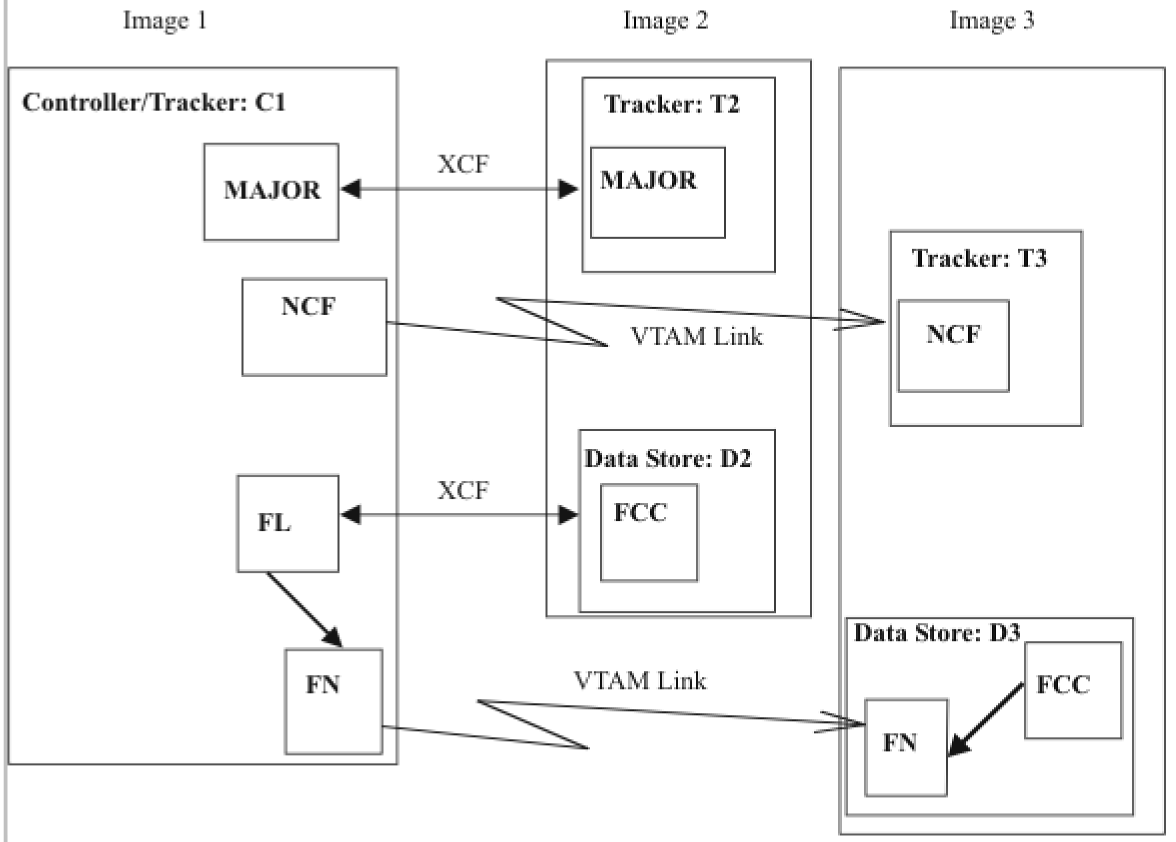
Mixed SNA and XCF connection

Figure 8: A mixed SNA and XCF connection on page 40 shows a mixed SNA and XCF connection. In Image 1, the controller and tracker are in the same address space. In Image 2, the tracker is connected using XCF. In Image 3, the remote tracker is connected using SNA with a VTAM® link. The spool is only shared between Image 1 and Image 2 (JES2 MAS). You must have two Data Stores, one installed in Image 2 and one in Image 3.

Note that the controller and tracker in Image 1 must have two different LU names. For each XCF connection, there must be a different XCF group name.

Example

Figure 8. A mixed SNA and XCF connection



Key:

FCC

Data Store Communication task

FL

Fetch Job Log task

FN

Data Store SNA handler task

MAJOR

Controller/tracker main task

NCF

Network Communication function

Table 10: Example members for Figure 10 on page 41 shows the initialization statements you can use to create the configuration in Figure 8: A mixed SNA and XCF connection on page 40.

Table 10. Example members for Figure 10

Controller member	

C1
`OPCOPTS`
` RCLEANUP(YES)`
` NCFTASK(YES)`
` NCFAPPL(LU00C1T)`
 	
`ROUTOPTS`
` SNA(LU000T3)`
` XCF(XCFMEMT2)`
 	
`XCFOPTS`
` GROUP(XCFGRUCT)`
` MEMBER(XCFMEMCT)`
 	
`FLOPTS`
` DSTGROUP(XCFGRUCD)`
` CTLMEM(XCFMEMCD)`
` XCFDEST(********.XCFMEMD2,`
` XCFMEMT2.XCFMEMD2)`
` CTLLUNAM(LU00C1D)`
` SNADEST(LU000T3.LU000D3)`	

Tracker members	

T2
`OPCOPTS`
` NCFTASK(NO)`
 	
`TRROPTS`
` HOSTCON(XCF)`
 	
`XCFOPTS`
` GROUP(XCFGRUCT)`
` MEMBER(XCFMEMT2)`
	T3
	`OPCOPTS`
	` NCFTASK(YES)`
	` NCFAPPL(LU000T3)`
	
	`TRROPTS`
	` HOSTCON(SNA)`
	` SNAHOST(LU00C1T)`

Data Store members	

D2
`DSTOPTS`
` HOSTCON(XCF)`
` DSTGROUP(XCFGRUCD)`
` DSTMEM(XCFMEMD2)`
	D3
	`DSTOPTS`
	` DSTLUNAM(LU000D3)`
	` CTLLUNAM(LU00C1D)`

Note: In this example, the XCF groups or the LU names for the communication partners are the following:

XCFGRUCD

The XCF group for the communication between controller and Data Store. The members in the group are:

XCFMEMCD

Controller C1.

XCFMEMD2

Data Store D2.

XCFGRUCT

The XCF group for the communication between controller and tracker. The members in the group are:

XCFMEMCT

Controller C1.

XCFMEMT2

Tracker T2.

LU00C1D

Controller C1, when communicating with D3.

LU000D3

Data Store D3.

LU00C1T

Controller C1, when communicating with T3.

LU000T3

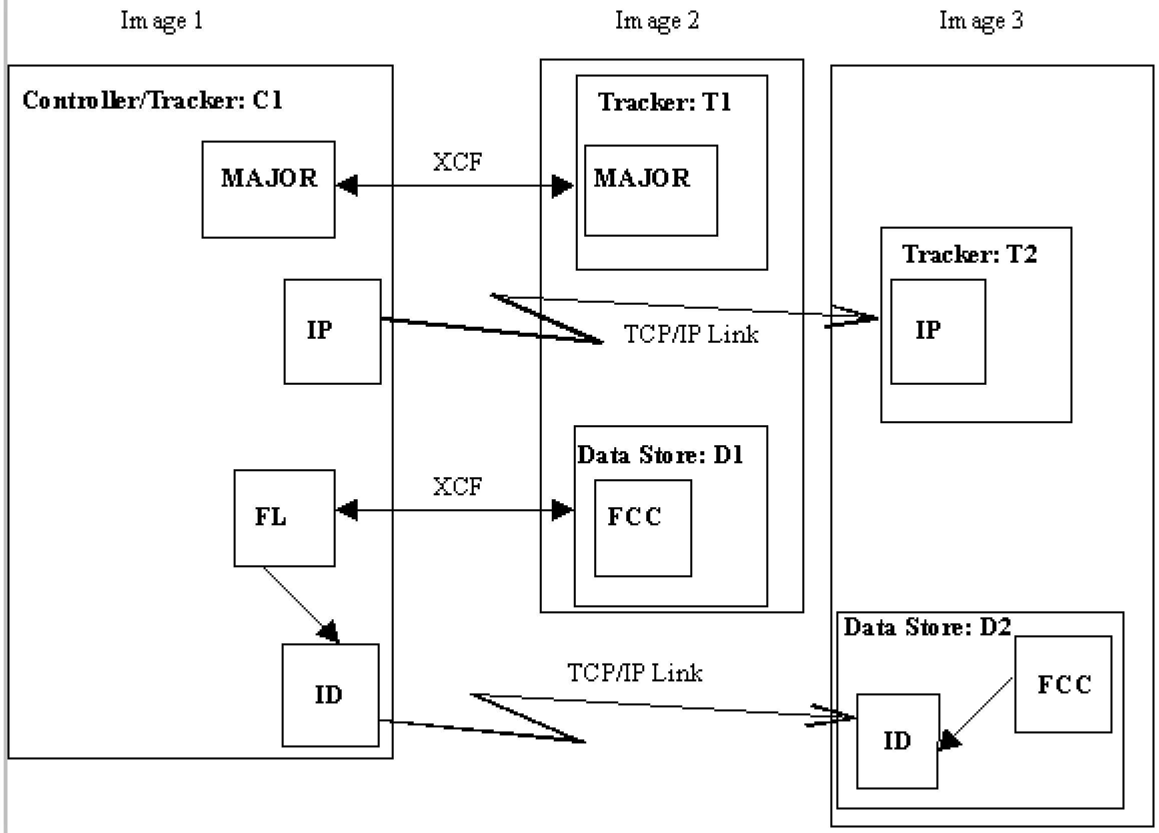
Tracker T3.

Mixed TCP/IP and XCF connection

Figure 9: A mixed TCP/IP and XCF connection on page 43 shows a mixed TCP/IP and XCF connection. In Image 1, the controller and tracker are in the same address space. In Image 2, the tracker is connected using XCF. In Image 3, the remote tracker is connected using TCP/IP. The spool is shared only between Image 1 and Image 2 (JES2 MAS). You must have two Data Stores, one installed in Image 2 and one in Image 3.

Example

Figure 9. A mixed TCP/IP and XCF connection



Key:

FCC

Data Store Communication task

FL

Fetch Job Log task

FN

Data Store SNA handler task

MAJOR

Controller/tracker main task

ID

Task for Data Store-to-controller TCP/IP communication

IP

Task for Tracker-to-controller TCP/IP communication

Table 11: Example members for Figure 11 on page 44 shows the initialization statements you can use to create the configuration in Figure 9: A mixed TCP/IP and XCF connection on page 43.

Table 11. Example members for Figure 11

Controller member	

C1
`OPCOPTS`
` RCLEANUP(YES)`
 	
` FLOPTS`
` TCPDEST(********.'1.22.333.4','1.22.333.4'`
 	
` DSTGROUP(XCFGRUCD)`
` CTLMEMT(XCFMEMCD)`
` XCFDEST(********.XCFMEMD2,XCFMEMT2.XCFMEMD2)`
 	
`ROUTOPTS`
` TCPIP(TRK1:'1.22.333.4')`
` XCF(XCFMEMCT)`	

Tracker members	

T1
`OPCOPTS`
` NCFTASK(NO)`
 	
`TRROPTS`
` HOSTCON(XCF)`
 	
`XCFOPTS`
` GROUP(XCFGRUCT)`
` MEMBER(XCFMEMT2)`
	T2
	`OPCOPTS`
	` NCFTASK(NO)`
	
	`TRROPTS`
	` HOSTCON(TCP)`
	` TCPHOSTNAME('1.22.333.4')`

Data Store members	

D1
`DSTOPTS`
` HOSTCON(XCF)`
` DSTGROUP(XCFGRUCD)`
` DSTMEM(XCFMEMD2)`
	D2
	`DSTOPTS`
	` HOSTCON(TCP)`
	` CTLHOSTNAME('1.22.333.4')`

Configuring a backup controller for disaster recovery

HCL Workload Automation for Z supports the recovery of a system failure between two remote sites. A disaster recovery process ensures that the business supported by the data center is always kept viable by switching from a local site where the failure occurs, to a remote site.

The controller in charge of planning, controlling, and monitoring the workload sends all the data and plan updates to a *backup* controller running in a different sysplex. In this way, the backup controller (also known as a *remote hot standby controller)* is kept up-to-date and can act as the *primary* controller when a planned or unplanned switch occurs.

You can configure a backup controller to replace the primary controller in the event of a system or connection failure. Both controllers must have the same configuration. The backup controller is continuously connected with the primary controller through TCP/IP and kept updated with all the required data. When the backup controller takes over from the primary controller, the tracker, if running, switches its connection from the primary to the backup controller.

To configure a backup controller and set its communication with the primary controller, define the following initialization statements. For detailed information about the statements, see *Customization and Tuning*.

BKPTOPTS

To define the local attributes for the TCP/IP communication between the primary and backup controller. Define this statement on both controllers.

OPCOPTS

To set the OPCHOST parameter, which defines the role of the subsystem. Define this statement on both controllers.

TRROPTS

To define the routing options from a z/OS tracker that is connected to a primary controller, and possibly also to a backup controller.

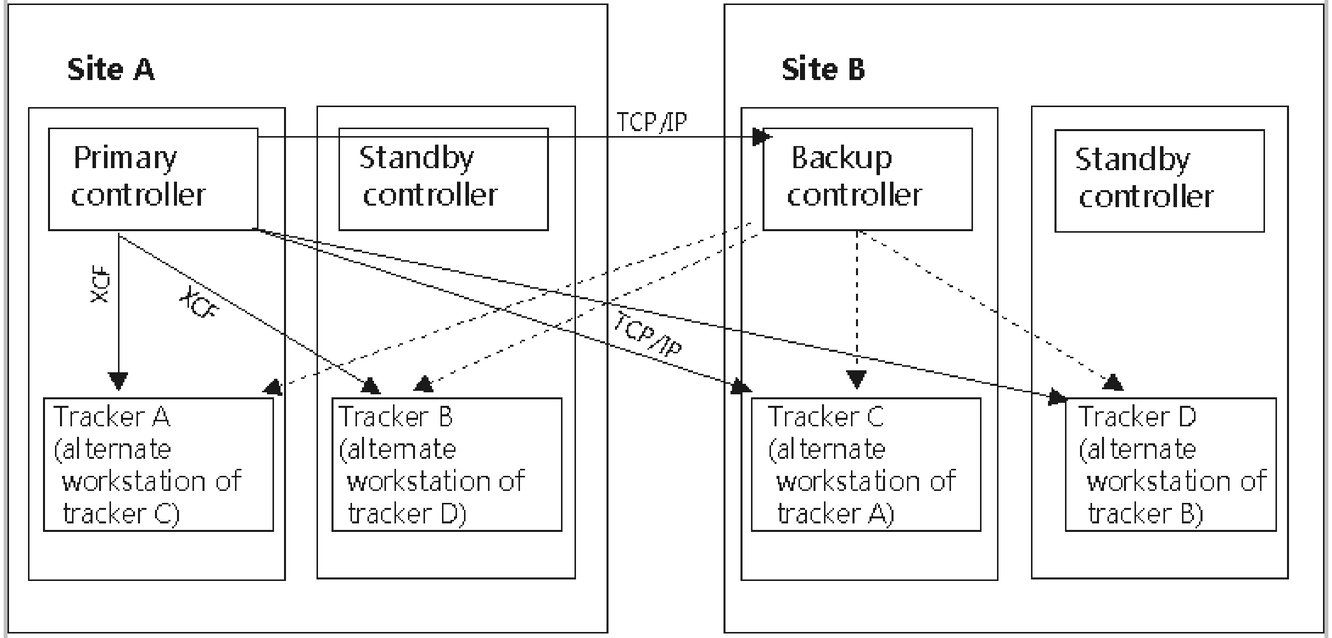
Note: To connect to a backup controller, a tracker can use only TCP/IP. In this case, to connect to a primary controller the same tracker can use only XCF or TCP/IP connection protocol.

The configuration to include a backup controller supports the following features:

- z/OS trackers
- HCL Workload Automation agents (also known as z-centric agents)
- Dynamic domain managers
- Cross dependencies

Figure 10: Configuring trackers, primary controller, and backup controller on page 46 shows two remote sites, with two images each.

Figure 10. Configuring trackers, primary controller, and backup controller


Site A
Site B
Primary controller
Standby controller
Backup controller
Standby controller
TCP/IP
XCF
XCF
TCP/IP
Tracker A (alternate workstation of tracker C)
Tracker B (alternate workstation of tracker D)
Tracker C (alternate workstation of tracker A)
Tracker D (alternate workstation of tracker B)

In this configuration:

- The primary controller is connected to trackers A and B through XCF links. It is connected to the backup controller, trackers C and D through TCP/IP.
- The backup controller connects to trackers A, B, C, and D through TCP/IP links when you issue the modify command `/F procname,BKTAKEOVER` and the backup controller becomes the controlling system.
- On the primary controller you set the following statements:

```
OPCOPTS  OPCHOST(YES)
BKPTOPTS TCPIPJOBNAME('TCPIP')
         HOSTNAME('1.11.111.111')
         LOCPORTNUMBER(7543)
         PEERHOSTNAME('2.22.222.222')
         PEERPORTNUMBER(6456)
ROUTOPTS XCF(HWA2)                      1
```

- On the backup controller you set the following statements:

```
OPCOPTS  OPCHOST(BACKUP)
BKPTOPTS TCPIPJOBNAME('TCPIP')
         HOSTNAME('2.22.222.222')
         LOCPORTNUMBER(6456)
         PEERHOSTNAME('1.11.111.111')
         PEERPORTNUMBER(7543)
ROUTOPTS TCPIP(HWA2:'3.33.333.333'/4348)1
```

- On trackers A and B you set the following statements:

```
TRROPTS  HOSTCON(XCF)
         BKPHOSTNAME('2.22.222.222')
         BKPPORTNUMBER(924)
XCFOPTS GROUP(GRUXCFX)
        MEMBER(HWA2)
TCPOPTS  HOSTNAME('3.33.333.333')
         TRKPORTNUMBER(4348)
```

- On trackers C and D you have set the following statements:

```
TRROPTS  HOSTCON(TCP)
         TCPHOSTNAME('1.11.111.111')
         TCPPORTNUMBER(424)
         BKPHOSTNAME('2.22.222.222')
         BKPPORTNUMBER(924)
TCPOPTS  HOSTNAME('3.33.333.333')
         TRKPORTNUMBER(4348)
```

1

On the primary and backup controller, the destination name in the ROUTOPTS statement must be the same (`HWA2`, in this example).

Chapter 3. Planning your installation

During the planning stage of your HCL Workload Automation for Z project, consider carefully how you want to install the scheduler to control your production workload. The installation consists of installing the tracker and Z controller in combinations to suit your processing environment, and connecting them using one or more of the communication methods described in Planning your configuration on page 24. Later, you can customize your HCL Workload Automation for Z systems to include more functions.

Before you start the installation tasks, ensure that you have all the resources you need to complete your installation. For a detailed description of the installation tasks, see Installing on page 60.

Configuring for availability

It is recommended that you install the tracker and Z controller as separate subsystems on an HCL Workload Automation for Z controlling system. The tracker can then continue to collect events even when the Z controller is stopped. Events are created by SMF and JES exits, and added to a queue in the z/OS extended common service area (ECSA). If the event writer is not active while work is running in the system, the queue might fill up, and new events might be lost. HCL Workload Automation for Z cannot recover these lost events.

You can improve HCL Workload Automation for Z availability using the z/OS® Automatic Restart Manager (ARM). Automatic restart management can reduce the impact of an unexpected error on HCL Workload Automation for Z because it can restart it automatically, without operator intervention.

To use Automatic Restart Manager, set the ARM parameter in the OPCOPTS statement to YES. For details about the ARM parameter, see *Customization and Tuning*.

Hot standby

If you connect your HCL Workload Automation for Z systems by using the z/OS cross-system coupling facility (XCF), you can include one or more standby Z controllers in your configuration. A standby system can take over the functions of the Z controller if the Z controller fails or if the z/OS system that it was active on fails. You can create a standby Z controller on one or more HCL Workload Automation for Z controlled systems within the XCF group. Each standby system must have access to the same resources as the Z controller. These resources include data sets and VTAM® cross-domain resources. However, if EQQMLOG is allocated as a data set, it cannot be shared between the Z controller and standby Z controller. The standby system is started the same way as the other HCL Workload Automation for Z address spaces, but is not activated unless a failure occurs or unless it is directed to take over through a z/OS operator modify command. If you use one or more servers to remotely access the controller, note that the server must always run on the same system as the controller.

Starting an event writer with an event reader function

Starting an event writer twith an event-reader function improves performance, because events are not written to the event data set and then read back again, which requires an event-reader task to continually check the event data set for new events. Instead, the event writer writes events to the event data set and forwards them directly to the Z controller through a TCP/IP , XCF, or NCF link.

Using two message log (MLOG) data sets

All the major components and tasks of HCL Workload Automation for Z log messages to either SYSOUT or to a data set, based on your configuration choice.

When the messages are logged to a data set (EQQMLOG), the message log data set remains under the control of the HCL Workload Automation for Z controller for as long as the controller is active. If you want to save, or clear, the contents of the data set, you must first stop the controller. Also, if the controller is not stopped for a long period of time, and EQQMLOG runs out of space, the messages are written into the system log.

To avoid these problems, including the increased amount of time required to find specific records in an oversized data set, you can configure HCL Workload Automation for Z to use two message log data sets, EQQMLOG and EQQMLOG2.

The two MLOGs are used alternatively: while one logs messages, the other remains inactive. Then, when the active data set reaches a pre-set level of completion, its contents are copied into a GDG data set and it goes idle, while the other data set starts logging. When the same level of completion is reached in the now active data set, the switch is repeated. Every time the data sets are shifted, message:

```
EQQZ402I SWITCH FROM ddn1 TO ddn2
```

is issued on SYSLOG and written in the just switched MLOG.

The messages copied from EQQMLOG and EQQMLOG2 are available in the GDG data set.

To configure HCL Workload Automation for Z to use two MLOGS, specify Y in the `MLOG SWITCH` field of the EQQJOBSC installation aid panel on page 69. To activate this feature, you must also specify in the `Switchlimit` field the number of records written in the MLOG file that you want to trigger the switch mechanism. When the file reaches this level of capacity, its contents are saved in the GDG file, and the oncoming messages are written on the alternate file.

The number of records in `Switchlimit` must be greater than 0 for the feature to be activated.

Specifying Y in `MLOG SWITCH` results in the creation of sample JCLs that:

- Add DD EQQMLOG2 in the controller started task JCL (samples EQQCONO and EQQCON) and initial parameters (samples EQQCONOP and EQQCONP)
- Add DD EQQMLOG2 in the tracker started task JCL and initial parameters (samples EQQTRA and EQQTRAP)
- Allocate the EQQMLOG2 data set like EQQMLOG (sample EQQPCS02)
- Create the GDG data set where the outgoing MLOG file is archived (samples EQQSMLOG and EQQREPRO)
- Allocate the GDG root to archive the MLOGS (sample EQQPCS12)

You must then make the EQQSMLOG procedure (copied from the EQQJOBS customized samples) accessible to the controller, for example by adding it in the `user.proclib`.

After the product is installed, you can configure the MLOGPROCNAME and SWITCHMLOGLIM keywords of the OPCOPTS initialization statement to set or change the feature (for detailed information about OPCOPTS, see *Customization and Tuning*).

Particular attention should be paid to calibrating the number of records specified in `Switchlimit` (or SWITCHMLOGLIM), the point being that an exceedingly low value triggers the switching function with unnecessary frequency while at the other end you risk filling the MLOG to its capacity and generate a B37 abend. As a best practice, you enter a number that at least doubles the number of records logged at the controller startup or during the workload peak (when many `ETT add` messages are sent).

For example, if you use a 3390 disk, a track is 56664 bytes. One cylinder includes 15 tracks, which are 849960 bytes. If you allocate an MLOG of 1 cylinder with 1 extent, the number of available bytes is 1699920. Since the MLOG record length is 125 bytes, the maximum number of available records is 13599 (1699920/125). Under these conditions, the appropriate number of records for `Switchlimit` or SWITCHMLOGLIM should be a little lower, perhaps 13000. If you then find that a greater number of messages is issued at short intervals during peak workload or at controller startup, this implies that you should increase the value and allocate more space for EQQMLOG.

Another advantage of using the MLOG switching function is that it prevents the loss of the messages that are written to EQQMLOG when abend B37 occurs as the MLOG file becomes full. In fact with the switching function on, if the current MLOG file fills up due to an inadequate number of records defined in `Switchlimit` or SWITCHMLOGLIM, when B37 is issued, a switch MLOG action is forced, all records are copied to the GDG data set, and control is passed to the alternate MLOG.

When this feature is installed, you can:

- Run the modify command, `/F subsys, SWITCHMLOG`, to force the switch to the alternate data set (EQQMLOG or EQQMLOG1), regardless of the number of currently logged messages, and starts the record counter from 0 again.
- See which of the two MLOG data sets is in current use on the BROWSING GENERAL CURRENT PLAN INFORMATION dialog (EQQSGCPP; fast path 6.6).

See also Message log data set (EQQMLOG) on page 134.

Checklist for installing HCL Workload Automation for Z

Checklist for installing HCL Workload Automation for Z on page 50 provides you a guide through the installation tasks for a tracker, Z controller, standby Z controller, or the ISPF dialogs.

Note: Always install the tracker first on the controlling system or on a system where a standby Z controller will be installed.

In the checklist, the task numbers are arranged in a recommended order but are not meant to imply a required order. You perform the tasks suited to your own configuration.

The **Applies to** column indicates for which HCL Workload Automation for Z address space you should perform that particular task. You might not need to perform every task outlined in the list. Skip those tasks or actions that do not apply to your installation.

A check mark (✓) in the **IPL** column means that an IPL of the z/OS system is needed for the change to take effect. It does not indicate how many IPLs are needed. You can install HCL Workload Automation for Z with only one IPL of the z/OS system by performing all the required steps before a scheduled IPL.

Table 12. Checklist for installing HCL Workload Automation for Z

Step	Description	Applies to	IPL
Step 1. Loading tracker software on page 62	***Load tracker software***. Perform the following actions on each system in your HCL Workload Automation for Z configuration: • Run SMP/E to receive tracker software. • Apply tracker maintenance.	• Tracker	
Step 2. Loading controller software on page 63	***Load Z controller software***. Perform the following actions on each system where you are installing a Z controller, standby Z controller, or dialogs: • Run SMP/E to receive Z controller software. • Apply Z controller maintenance.	• Z controller • Standby Z controller • Dialogs	
This step is not applicable to this version.	***Load national language support (NLS) software for the Z controller.*** This step is not applicable to this version.		
Step 4. Using the EQQJOBS installation aid on page 63	***Run the EQQJOBS installation aid***. You can run EQQJOBS as soon as the tracker software is loaded. It helps you install HCL Workload Automation for Z: • Set up EQQJOBS. • Create the sample job JCL. Do this to generate tailored samples from the EQQJOBS dialog. • Generate batch job skeletons. Use EQQJOBS to generate skeletons for the ISPF dialogs. • Optionally generate the Data Store samples if you want to install the Data Store.	• Tracker • Z controller • Standby Z controller • Dialogs	
Step 5. Adding SMF and JES exits for event tracking on page 87	***Add SMF and JES event tracking exits***. Perform this task on every z/OS system in your HCL Workload Automation for Z configuration.	• Tracker	✓

Table 12. Checklist for installing HCL Workload Automation for Z (continued)

Step	Description	Applies to	IPL
	Note: If you place exits in a link-pack-area (LPA) library, you must perform an IPL of the z/OS system with the CLPA option.		
Step 6. Updating SYS1.PARMLIB on page 90	***Update SYS1.PARMLIB***. On each system where you are installing the product, perform the actions that are applicable to your installation: • Define HCL Workload Automation for Z subsystems (IEFSSN*nn*). This is required for each system where the product is installed. • Authorize the HCL Workload Automation for Z load module library (IEAAPF*nn*). Do this if you install the product in a separate load module library. • Update SMF parameters (SMFPRM*nn*). Do this when installing a tracker. • Update dump-content definitions. Consider this on each system where you are installing the product. • Update the z/OS link-library definition (LNKLST*nn*) on each system where you are installing the product. • Update XCF initialization options (COUPLE*nn*). Review this section if you use XCF connections. • Modify TSO parameters (IKJTSO*nn*). Do this when installing a Z controller, a standby Z controller, or the ISPF dialogs. • Update PPT for performance (SCHED*nn*) on each system where you are installing the product. • Define the DLF exit for Hiperbatch™ support (COFDLF*nn*). Do this if you use Hiperbatch™ support. • Choose whether to start HCL Workload Automation for Z automatically (COMMND*nn*).	• Tracker • Z controller • Standby controller • Dialogs	✓

Table 12. Checklist for installing HCL Workload Automation for Z (continued)

Step	Description	Applies to	IPL
	Consider this on each system where you are installing the product. • Update APPC options (APPCPM*nn*). Consider this action if you intend to use the HCL Workload Automation for Z API or server. Define VTAM® resources before you update SYS1.PARMLIB. Coordinate this action with task 18 or 19.		
Step 7. Setting up the RACF environment on page 101	***Set up the RACF® environment***. Perform these actions on each system in your HCL Workload Automation for Z configuration: • Update RACF® for HCL Workload Automation for Z started tasks (ICHRIN03) on all HCL Workload Automation for Z started tasks on each system. • Update RACF® for a Z controller or standby Z controller.	• Tracker • Z controller • Standby controller • Dialogs	✓
At this point, if you placed exit modules in LPA, you can IPL with CLPA. No other options for HCL Workload Automation for Z require an IPL.			
Step 8. Securing communications on page 109	***Set up the SSL environment*** Perform these actions to activate a secure communication in a TCP/IP network: • Create the SSL work directory. • Create as many private keys, certificates, and trusted certification authority (CA) chains as you plan to use in your network. • Configure the scheduler, by specifying the TCPOPTS statement for each component of your network.	• Tracker • Z controller • Standby controller • Data Store server • User address space	
Step 9. Allocating data sets on page 112	***Allocate data sets***. Perform these actions if you are installing a tracker or a Z controller:	• Tracker • Z controller • Data Store server	

Table 12. Checklist for installing HCL Workload Automation for Z (continued)

Step	Description	Applies to	IPL
	• Review the section on allocating HCL Workload Automation for Z data sets. Do this before you allocate data sets. • Allocate VSAM data sets for a Z controller. Perform this action to create new data sets for a Z controller. • Allocate non-VSAM data sets. Perform this action for each HCL Workload Automation for Z address space. • Optionally, allocate the VSAM Data Store data sets if you want to use the Data Store.		
Implementing support for started-task operations on page 141	***Update SYS1.PROCLIB***. Perform these actions for each HCL Workload Automation for Z address space. • Create a JCL procedure for each address space on all z/OS systems where you are installing HCL Workload Automation for Z. • If you use HCL Workload Automation for Z to schedule started-task operations, ensure that the started-task-submit data set (EQQSTC) is in the JES PROCLIB concatenation and in the master scheduler start procedure. • If you use Restart and Cleanup, copy the EQQCLEAN sample procedure to a data set that is referenced in the JES PROCLIB concatenation.	• Tracker • Z controller • Standby controller	
Step 11. Defining the initialization statements on page 146	***Define initialization statements***. Perform this task for each HCL Workload Automation for Z address space: • Define initialization statements. Create members in the parameter library (EQQPARM) for each address space.	• Tracker • Z controller • Standby controller	
If you are not using NCF, XCF, or TCP/IP connections you can now start a tracker and continue with the verification task.			

Table 12. Checklist for installing HCL Workload Automation for Z (continued)

Step	Description	Applies to	IPL
The ISPF environment on page 192	***Set up the ISPF environment***. Perform these actions on the system where you are installing the ISPF dialogs. • Set up the HCL Workload Automation for Z CLIST library. • Set up the ISPF tables. • Allocate ISPF and HCL Workload Automation for Z data sets to your TSO session. • Invoke the HCL Workload Automation for Z dialog.	• Dialogs	
If you are not using NCF, XCF, or TCP/IP connections, the API or server, you can now start a Z controller and continue with the verification task.			
Step 13. Using XCF for communication on page 153	***Using XCF for local communications***. Even if you have already specified XCF initialization statements in Task 12 and updated the COUPLE*nn* member in Task 7, consider these actions if you use XCF: • Update XCF initialization options. Ensure that XCF initialization options are suitable for your HCL Workload Automation for Z configuration. • Add initialization statement options for XCF. Specify XCF runtime options in the parameter library for all started tasks.	• Tracker • Z controller • Standby controller	
Step 14. Activating the network communication function on page 155	***Activate NCF for VTAM® connections***. Perform these actions for each address space that is connected through NCF. Ensure that a standby Z controller has the same tracker connections as the Z controller and that each tracker can connect to all standby Z controllers:	• Tracker • Z controller • Standby controller	

Table 12. Checklist for installing HCL Workload Automation for Z (continued)

Step	Description	Applies to	IPL
	• Add NCF network definitions. Define VTAM® applications on each system where a started task uses NCF. • Add NCF session parameters on each z/OS system where HCL Workload Automation for Z is installed. • Update the COS table. Consider this action if you do not want to use the VTAM® COS default entry. • Activate network resources. Check that all the required VTAM® resources are active. • Add NCF initialization options. Include initialization statement options in the parameter library for all started tasks that use NCF.		
Step 15. Using TCP/IP for communication on page 158	***Activate TCP/IP connections*** Perform these actions for each address space that is connected via TCP/IP. Ensure that a standby Z controller has the same tracker connections as the Z controller and that each tracker can connect to all standby Z controllers: • Add TCP/IP network definitions. Define IP address for the controller and tracker. • Add TCP/IP initialization options. Include initialization statement options in the parameter library for all started tasks that use TCP/IP. • For TCP/IP, the HCL Workload Automation for Z server can manage up to 500 concurrent connection requests in a queue. In the PROFILE.TCPIP configuration file, set the SOMAXCONN statement to a value not greater than 500.	• Tracker • Z controller • Standby controller	
Step 16. Activating support for the API on page 159	***Activate support for the HCL Workload Automation for Z API***.	• Tracker • Z controller • Standby controller	

Table 12. Checklist for installing HCL Workload Automation for Z (continued)

Step	Description	Applies to	IPL
	To use the API, perform these actions for each HCL Workload Automation for Z address space that you want to send requests to: • Define VTAM® resources. Define a local LU for HCL Workload Automation for Z, logon modes, and cross-domain resources, as required. • Update APPC options. Update the APPCPM*nn* member of SYS1.PARMLIB. • Activate HCL Workload Automation for Z support for APPC. In the parameter library, include APPCTASK(YES) on the OPCOPTS statement.		
Step 17. Activating support for the product dialog and programming interface using the server on page 162	***Activate support for the HCL Workload Automation for Z Server to use APPC or TCP/IP communications.*** **Note:** Include this task when activating an HCL Workload Automation for Z server. To activate the server, perform these actions in the order shown: 1. Define VTAM® resources. Define a local LU for HCL Workload Automation for Z, logon modes, and cross-domain resources, as required. 2. Update APPC options. Update the APPCPM*nn* member of SYS1.PARMLIB. 3. Activate HCL Workload Automation for Z support for APPC. In the parameter library, include SERVERS on the OPCOPTS statement.	• Server • Z controller • Standby controller	
Step 18. Activating support for end-to-end scheduling with z-centric capabilities on page 166	***Activate support for end-to-end scheduling with z-centric capabilities***	• Tracker • Z controller • Standby controller	

Table 12. Checklist for installing HCL Workload Automation for Z (continued)

Step	Description	Applies to	IPL
	Note: Include this task when you intend to use HCL Workload Automation for Z to schedule jobs on distributed z-centric agents. • Define the ROUTOPTS initialization parameters for the controller. • Define the HTTPOPTS initialization parameters for the tracker.		
Step 19. Activating support for Dynamic Workload Console on page 167	***Activate Support for the Dynamic Workload Console*** **Note:** Include this task when activating a server and intending to use the Dynamic Workload Console. To activate the server, perform these actions: • Install the Connector • In the parameter library, include SERVERS on the OPCOPTS statement.	• Server • Z controller • Standby controller	
Step 20. Activating support for the Java utilities on page 168	***Activate Support for the Java™ utilities*** Perform this task if you want to use one of the following features: • Dynamic Workload Console reporting. • Event-driven workload automation for data set triggering, with centralized deploy process.	• Z controller	

Part II. Installing and migrating HCL Workload Automation for Z

Chapter 4. Installing

#unique_40_Connect_42_itasks summarizes the tasks to install HCL Workload Automation for Z and identifies the address space type for each of them. Depending on your configuration, you might not need to perform every task indicated in the table. Skip those sections that are not relevant to your installation.

Before beginning the installation tasks, determine the HCL Workload Automation for Z configuration you want to create and the functions you want to use (for details, see Planning your configuration on page 24).

Note: To generate the skeletons that are appropriate to the current version, you are required to run EQQJOBS. For details, see Step 4. Using the EQQJOBS installation aid on page 63.

Table 13. HCL Workload Automation for Z installation tasks

Installation task	Perform for					
	tracker	Controller	Server	Standby Z controller	Data Store	Dialogs
Step 1. Loading tracker software on page 62	✓	✓		✓	✓	✓
Step 2. Loading controller software on page 63		✓		✓		✓
Step 3. Loading national language support software on page 63	N/A	N/A	N/A	N/A	N/A	N/A
Step 4. Using the EQQJOBS installation aid on page 63		✓			✓	✓
Step 5. Adding SMF and JES exits for event tracking on page 87	✓					
Step 6. Updating SYS1.PARMLIB on page 90	✓	✓		✓		✓
Step 7. Setting up the RACF environment on page 101	✓	✓		✓		
Step 8. Securing communications on page 109	✓	✓	✓	✓	✓	✓
Step 9. Allocating data sets on page 112				✓	✓	
• Allocating the VSAM data sets on page 112		✓				
• Allocating non-VSAM data sets on page 121	✓	✓				

Table 13. HCL Workload Automation for Z installation tasks (continued)

Installation task	Perform for					
	tracker	Controller	Server	Standby Z controller	Data Store	Dialogs
• Allocating the files and directories on page 138			✓			
Step 10. Creating JCL procedures for address spaces on page 140	✓	✓		✓	✓	
Step 11. Defining the initialization statements on page 146	✓	✓		✓	✓	
Step 12. Setting up the ISPF environment on page 146						✓
Step 13. Using XCF for communication on page 153	✓	✓		✓	✓	
Step 14. Activating the network communication function on page 155	✓	✓		✓	✓	
Step 15. Using TCP/IP for communication on page 158	✓	✓		✓		
Step 16. Activating support for the API on page 159		✓		✓		
Step 17. Activating support for the product dialog and programming interface using the server on page 162		✓		✓		
Step 19. Activating support for Dynamic Workload Console on page 167		✓		✓		
Step 20. Activating support for the Java utilities on page 168		✓				
Verifying your installation on page 170						
• Verifying installation of a tracker on page 170	✓					
• Verifying installation of a controller and dialogs on page 176		✓				
• Verifying installation of a standby controller on page 180				✓		

Table 13. HCL Workload Automation for Z installation tasks (continued)

Installation task	Perform for					
	tracker	Controller	Server	Standby Z controller	Data Store	Dialogs
• Verifying installation of the Restart and Cleanup function on page 180		✓		✓	✓	
• Verifying configuration on page 182	✓	✓		✓		

Step 1. Loading tracker software

You must load tracker software on each z/OS system in your configuration. Process the software distribution media by using the facilities of System Modification Program Extended (SMP/E). This creates or updates the necessary software libraries on your system. Table 14: tracker libraries loaded by SMP/E on page 62 describes the distribution and target libraries that are created or updated by SMP/E.

Table 14. tracker libraries loaded by SMP/E

SMP/E DD name		
Distribution	Target	Description
AEQQPNL0	SEQQPNL0	Panels for the EQQJOBS installation aid
AEQQMOD0 (object)	SEQQLMD0 (load)	tracker modules
AEQQMSG0	SEQQMSG0	Messages
AEQQMAC0	SEQQMAC0	Assembler macros
AEQQCLIB	SEQQCLIB	HCL Workload Automation for Z CLISTs
AEQQSAMP	SEQQSAMP	Sample exits, programs, and JCL
AEQQSKL0	SEQQSKL0	JCL skeletons, input to EQQJOBS
AEQQTBL0	SEQQTBL0	ISPF tables
AEQQDATA	SEQQDATA	Sample HCL Workload Automation for Z databases

It is recommended that you place all the HCL Workload Automation for Z load modules in a separate library. Use the same library for both the tracker and the Z controller. Create the library before you run the SMP/E jobs.

Alternatively, you can place the HCL Workload Automation for Z load modules in one of your existing load-module libraries, for example SYS1.LINKLIB. The remaining data sets loaded by SMP/E are new data sets that you must create before running the SMP/E jobs. For the JCL and instructions to load the software, see the *HCL Workload Scheduler for Z: Program Directory*.

Step 2. Loading controller software

To load the Z controller software, process the software distribution media by using SMP/E. This creates or updates the necessary disk-resident libraries on your system. For a description of the data set that is created or updated by SMP/E, see Table 15: Controller libraries loaded by SMP/E on page 63.

Table 15. Controller libraries loaded by SMP/E

SMP/E DD name		
Distribution	**Target**	**Description**
AEQQMOD0 (object)	SEQQLMD0 (load)	Controller modules

Recommendation: Install the Z controller in the same library that you are using for the tracker.

Step 3. Loading national language support software

Loading national language support (NLS) software is not applicable for HCL Workload Automation for Z

Step 4. Using the EQQJOBS installation aid

EQQJOBS is a CLIST-driven ISPF dialog that can help you install HCL Workload Automation for Z. You can set up EQQJOBS as soon as the tracker software has been installed. EQQJOBS assists in the installation of the tracker and Z controller by building batch-job JCL that is tailored to your requirements. You can then use the JCL to perform the following actions:

- Install the tracking exits
- Set up RACF® security
- Create data sets
- Create started-task JCL
- Perform planning functions

Set up EQQJOBS now, so that it is ready to use when needed.

Setting up the EQQJOBS installation aid

EQQJOBS reads skeleton JCL from the SEQQSAMP or SEQQSKL0 libraries, tailors the JCL, and then writes the tailored JCL to an output library that you specify. The output library must be a partitioned data set with record length 80 and record format FB, and must be allocated before you run EQQJOBS. The components of EQQJOBS reside in these libraries:

SEQQCLIB

CLIST to drive the dialog

SEQQPNL0

EQQJOBS panels

SEQQSAMP

Sample JCL

SEQQSKL0

HCL Workload Automation for Z batch-job skeletons.

To be able to run EQQJOBS, you can either:

- Allocate the following libraries to the DD statements in your TSO session:
    - SEQQCLIB to SYSPROC
    - SEQQPNL0 to ISPPLIB
    - SEQQSKL0 and SEQQSAMP to ISPSLIB.

  To invoke EQQJOBS, enter the TSO command `EQQJOBS` from an ISPF environment.
- From the SEQQCLIB library, invoke the EQQ#JOBS CLIST (where the symbol # depends from the code page you are using, this is related to code page 1047) by issuing the `EXEC` command.

  In this way, the required libraries are automatically allocated, and EQQJOBS is automatically invoked.

The following panel is displayed:

Example

Figure 11. EQQJOBS0 - EQQJOBS application menu

```
  EQQJOBS0 ------------ EQQJOBS APPLICATION MENU   -------------------------------
Select option ===>
                                                           Userid   - SYSPROG
                                                           Time     - 15:04
   1 - Create sample job JCL                               Terminal - 3278


   2 - Generate batch-job skeletons


   3 - Generate Data Store samples


   4 - Generate WAPL samples


   X - Exit from the EQQJOBS dialog
```

The following sections describe:

- Option 1, Creating the sample job JCL on page 65
- Option 2, Generating batch-job skeletons on page 75
- Option 3, Generating Data Store samples on page 80
- Option 4, Creating the Workload Automation Programming Language samples on page 84

Note:

1. To ensure that all files are correctly allocated, perform first option 2 then option 1.
2. If you need to run EQQJOBS again, ensure that you saved your customized batch-job skeletons because they will be generated again and overwritten.

Creating the sample job JCL

To ensure that all the files are correctly allocated, before creating the sample job JCL you must have generated the batch-job skeletons, as described in Generating batch-job skeletons on page 75.

To create the sample job JCL:

1. From the EQQJOBS application menu, select option 1. The following panel is displayed:

   Figure 12. EQQJOBS3 - Create sample job JCL

```
EQQJOBS3 ------------------- CREATE SAMPLE JOB JCL ----------------------------
Command ===>

The data set names specified on this panel should be fully qualified
names without any enclosing apostrophes.

Enter the name of the output library:

Sample job JCL        ===> CCOPC.OPCA.INSTJCL______________________________
BAtch-job skeletons   ===> ________________________________________________

Job statement information:

===> //SYSPROG1 JOB (111111,2222),'OPCESA BATCH',CLASS=B,MSGCLASS=H,_______
===> //         MSGLEVEL=(1,1),NOTIFY=SYSPROG______________________________
===> //*___________________________________________________________________
===> //*___________________________________________________________________

The following data set names are used by one or more of the generated job

Message library name  ===> OPC.SEQQMSG0____________________________________
Data library name     ===> DEV.DATA________________________________________
Parameter library     ===> CCOPC.OPCA.PARM_________________________________
Checkpoint data set    ===> CCOPC.OPCA.CKPT_________________________________
Press ENTER to continue
```

   The data set names that you specify on this panel must be completely qualified and not be enclosed within quotes.

   Sample job JCL

   Required. Name of the library to which you want that the generated JCL samples are written. The library must be allocated before you generate the batch JCL samples. Ensure that the library that you specify has sufficient directory blocks to store all the sample members that are generated by EQQJOBS (for details, see Table 16: Sample JCL generated by the EQQJOBS dialog on page 72.)

   Batch-job skeletons

   Required. Name of the library where the JCL skeletons are to be stored (either a new library that you create or an existing JCL skeletons library). Before you submit the batch jobs, allocate this library to the ISPSLIB DD statement in the TSO session of dialog users. For a description about how to set up the dialog, see Step 12. Setting up the ISPF environment on page 146.

Job statement information

Required. The JOB statement that follows standard JCL syntax and your installation standards.

Message library name

Required. Name of the library that contains the HCL Workload Automation for Z messages (SMP/E target DDNAME SEQQMSG0).

Data library name

Required. Name of the library that will contain the SSL certificates provided with HCL Workload Automation for Z. For detailed information about these certificates, see *Scheduling End-to-end with z-centric Capabilities*.

Parameter library

Required. Name of the library that will contain the initialization statements. This library will be allocated by the EQQPCS01 batch job.

Checkpoint data set

Required. Name of the checkpoint data set. This library will be allocated by the EQQPCS01 batch job.

2. Press Enter. The following panel is displayed:

Figure 13. EQQJOBS4 - Create sample job JCL

```
EQQJOBS4 ------------------- CREATE SAMPLE JOB JCL ----------------------------
 Command ===>

 Enter the following required job stream parameters:

 non-VSAM dsn prefix   ===> CCOPC.OPCA________________
 VSAM dsn prefix       ===> CCOPC.OPCAV_______________
 Unit name             ===> 3390____      Default unit name
 Primary volume serial ===> PROD01        Primary volume serial for VSAM
 Backup  volume serial ===> PROD02        Secondary volume serial for VSAM
 SYSOUT class          ===> *             SYSOUT class for reports


 The following information is optional:

 STEPLIB dsname        ===>
OPC.SEQQLMD0______________________________________
 VSAMCAT dsname        ===>
________________________________________________
 VSAM password         ===> ________
 Dsn prefix of old
  VSAM files           ===> CCOPC.OPCAV_______________
  non-VSAM files       ===> CCOPC.OPCA________________
 Samples with cloning support generated:   N   (Y/N)

 Static symbol used    ===> SYSCLONE without enclosing ‘&’ and period

 Press ENTER to continue
```

non-VSAM dsn prefix

Required. The qualifiers that prefix the non-VSAM data set names. HCL Workload Automation for Z adds a low-level qualifier to the prefix to uniquely identify the non-VSAM data sets. For example, it adds EV for the event data set. In this example, the full name is CCOPC.OPCA.EV.

Note: HCL Workload Automation for Z does not use the prefix for the parameter library or checkpoint data set. You specified the names of these data sets in the previous panel.

VSAM dsn prefix

Required. The qualifiers that prefix the VSAM data set names. HCL Workload Automation for Z adds a low-level qualifier to the prefix to uniquely identify each HCL Workload Automation for Z VSAM data set. For example, it adds AD for the application description data set. In this example, the complete name is CCOPC.OPCAV.AD.

Unit name

Required. The device name that is valid at your installation; it can be a device type, for example 3380, or a group name, for example PROD or TEST.

Primary volume serial

Required. The volume to be used by sample job EQQPCS01 to allocate the primary data sets. Some HCL Workload Automation for Z logical files are implemented as two physical data sets, a primary and an alternate; for example, the current plan data set. To minimize the potential impact of errors on a particular device, allocate the primary and alternate data sets on different physical devices.

SYSOUT Class

Required. The SYSOUT class that you want to use for the reports that are generated by the sample jobs.

STEPLIB dsname

Optional. Name of the HCL Workload Automation for Z load module library, if the load modules are not in a data set included in an active LNKLST member.

VSAMCAT dsname

Optional. Name of a catalog in which VSAM data sets are to be defined, if they are not to be defined in the master catalog.

VSAM password

Optional. The password for the VSAM catalog, if the catalog is password-protected.

VSAM files

Optional. The qualifiers that prefix your existing HCL Workload Automation for Z VSAM data set names. These are used to create the data set-conversion sample JCL.

non-VSAM files

Optional. The qualifiers that prefix your existing HCL Workload Automation for Z non-VSAM data set names. These are used to create the data set-conversion sample JCL.

Samples with cloning support generated

Optional. Enter Y if you want the SYSCLONE variable resolved.

Note:

a. Generated JCLs do not contain a period before &SYSCLONE.
b. &SYSCLONE variable is intended to be substituted in the scheduler started tasks. It is not substituted in the generated JCLs that run as batch jobs. To obtain the variable substitution, run the JCL as cataloged procedure.

3. Click Enter. The following panel is displayed:

Figure 14. EQQJOBS9 - Create sample job JCL

```
EQQJOBS9 ------------------- CREATE SAMPLE JOB JCL ----------------------------
Command ===>

  JAVA UTILITIES ENABLEMENT:   Y        (Y= Yes ,N= No)
   Installation Directory      ===> /usr/lpp/TWS/V9R5M0______________________
                               ===> ________________________________________
                               ===> ________________________________________
   Java Directory              ===> /usr/lpp/java/J5.6______________________
                               ===> ________________________________________
                               ===> ________________________________________
   Work Directory              ===> /var/TWS/inst___________________________
                               ===> ________________________________________
                               ===> ________________________________________
   User ID                     ===> UID_____
   Group ID                    ===> GID_____

   JZOS Batch Launcher
      PDSE Library             ===> ___________________________________________
      Load Module Name         ===> JVMLDMnn

Press ENTER to continue
```

JAVA UTILITIES ENABLEMENT

Specify Y to enable one or all the following features:

- Dynamic Workload Console reporting.
- Event-driven workload automation feature for data set triggering.
- Jobs submission on z-centric and dynamic agents.

Installation Directory

The directory, with its complete path, where the product specific HFS or ZFS files are stored. This path corresponds to the path where the binary files are located, omitting the `/bin` subdirectory.

Java™ Directory

The HFS or ZFS directory where the Java™ Software Development Kit (SDK) for z/OS® is installed.

Work Directory

The directory where the subsystem specific HFS or ZFS files are stored. Each subsystem supporting the JAVA utility must have its own work directory.

User ID

The UNIX™ System Services user ID.

Group ID

The UNIX™ System Services group ID.

JZOS Batch Launcher PDSE Library

The PDSE that contains the JZOS Batch Launcher JVMLDM module.

JZOS Batch Launcher Load Module Name

Specify JVMLDM*nn*, meaning the JZOS Batch Launcher load module name used with the 64-bit SDK for z/OS® supported version.

4. Click Enter. The following panel is displayed:

Figure 15. EQQJOBSC - Create sample job JCL

```
EQQJOBSC ------------------- CREATE SAMPLE JOB JCL ----------------------------
Command ===>

  OUTPUT COLLECTOR                    _        (Y= Yes ,N= No)
   Class                         ===> ________
   Writer                        ===> ________

  MLOG SWITCH                         _        (Y= Yes ,N= No)
   Switchlimit                   ===> _______  (0 , nnnn)

  EXTENDED AUDITING              ===> _        (Y= Yes ,N= No)

  STEP AWARENES                         Y        (Y= Yes ,N= No)
   Store on primary controller     ===> Y        (Y= Yes ,N= No) forced to N if
                                                  Step Awareness is N
Press ENTER to continue
```

OUTPUT COLLECTOR

If you plan to use the Output collector started task to collect job logs from HCL Workload Automation Agents, specify Y to generate the related samples.

Class

Name of the JES CLASS where the job logs are to be copied by Output collector. If you do not specify this value, no JESCLASS keyword is recorded in the generated EQQOUCP sample; you can define it later in the OUCOPTS initialization statement (for details about OUCOPTS, see *Customization and Tuning*).

Writer

Name of the user ID associated with the WRITER task used to copy the job logs to the JES SYSOUT. If you do not specify this value, no WRITER keyword is recorded in the generated EQQOUCP sample; you can define it later in the OUCOPTS initialization statement (for details about OUCOPTS, see *Customization and Tuning*).

MLOG SWITCH

Enter Y if you plan to use the message log (MLOG) switching function, which automatically switches to a second MLOG file when the first one reaches the number of records specified in `Switchlimit`. Specify Y to generate the related samples.

See also Using two message log (MLOG) data sets on page 49.

Switchlimit

The number of records that must be present in the MLOG file to activate the switching function. When this number is reached, the switching function activates an alternate MLOG file and archives the records of the first MLOG file into a GDG data set. You can also use the OPCOPTS SWITCHMLOGLIM parameter to specify or modify this number after the product is installed.

Note that unless you specify a positive number of records, the function is not activated, even if you specified Y in the MLOG SWITCH field. The maximum possible value is 999 999 999 records.

See also Using two message log (MLOG) data sets on page 49.

EXTENDED AUDITING

Specify YES to activate the extended auditing feature. This value is meaningful only if you have set AMOUNT(EXTENDED) in the AUDIT statement of the controller (for details, see *Customization and Tuning*).

By activating the extended auditing feature, the EQQPCS14 sample creates the EQQDB*nn*, EQQDBARC, and EQQDBOUT data sets. EQQDB*nn* and EQQDBARC are added to the controller started-task. EQQDBARC and EQQDBOUT are added to the DP batch job.

STEP AWARENESS

Specify YES to have the tracker generate a list of the step events related to the jobs you run, and send it to the primary controller. The primary controller logs the step list, without storing it. If a backup controller exists, the primary controller sends the list to the backup controller, where the information is stored.

Store on primary controller

Specify YES to store the list of step events also on the primary controller. This option can affect the controller performance.

5. Click Enter. The following panel is displayed:

Figure 16. EQQJOBSD - Create sample job JCL

```
EQQJOBSD ------------------- CREATE SAMPLE JOB JCL ----------------------------
Command ===>

  CDP LOG INTEGRATION ENABLEMENT:    Y  (Y= Yes ,N= No)
   HFS Path for log files      ===>  /u/cdplog______________________________
   Switch CDP log limit        ===>  12000  (1, nnnnnnnn)
    (Switch limit is an integer representing the number of records to be
      written in current LOG before to switch to the second, default is 1000)

  BKPT configuration           N         (Y= Yes ,N= No)
   Backup data set name        ===> TWSZ.INST.BKPT__________________________

  BKPTOPTS statement procedure names:
   DUMP Procedures for:
   NCP, NCX, NXD Files         ===> ________       CX, CP1, XD1 Files      ===> ________
   LTP File                    ===> ________       CX, CP2, XD2 Files      ===> ________

   RESTORE Procedures:
   NCP, NCX, NXD Files         ===> ________       CX, CP1, XD1 Files      ===> ________
   LTP FIle                    ===> ________       CX, CP2, XD2 Files      ===> ________

 Press ENTER to create sample job JCL.
```

CDP LOG INTEGRATION ENABLEMENT

Enter Y to enable the logging of data in the EQQCDP1 and EQQCDP2 log files, which are read by IBM Common Data Provider for z Systems (CDP) if you configure it to integrate with HCL Workload Automation for Z.

HFS Path for log files

The complete path to the EQQCDP1 and EQQCDP2 log files (up to 40 alphanumeric characters). It must be existent in the HFS file system.

Switch CDP log limit

How many records must be written in the CDP log, before the CDP log switching function is started. The CDP log switching function stops the running CDP log file, and starts the alternate CDP log file. The alternate CDP log file records upcoming messages until the log limit value is reached again, and the process is repeated.

The default is 1000. Valid values are between 1 and 999 999 999.

BKPT configuration

Specify Y to enable the backup controller configuration.

Backup data set name

Required. The name of the backup data set.

DUMP Procedures for NCP, NCX, NXD Files

Name of the dump procedure for NCP, NCX, and NXD files. If you do not specify any value, the default EQQSENCP is used.

DUMP Procedures for CX, CP1, XD1 Files

Name of the dump procedure for CX, CP1, XD1, and ST files. If you do not specify any value, the default EQQSECP1 is used.

DUMP Procedures for CX, CP2, XD2 File

Name of the dump procedure for CX, CP2, XD2, and ST files. If you do not specify any value, the default EQQSECP2 is used.

DUMP Procedures for LTP Files

Name of the dump procedure for the LTP file. If you do not specify any value, the default EQQSENLT is used.

RESTORE Procedures for NCP, NCX, NXD Files

Name of the restore procedure for NCP, NCX, and NXD files. If you do not specify any value, the default EQQRENCP is used.

RESTORE Procedures for CX, CP1, XD1 Files

Name of the restore procedure for CX, CP1, XD1, and ST files. If you do not specify any value, the default EQQRECP1 is used.

RESTORE Procedures for CX, CP2, XD2 Files

Name of the restore procedure for CX, CP2, XD2, and ST files. If you do not specify any value, the default EQQRECP2 is used.

RESTORE Procedures for LTP File

Name of the restore procedure for the LTP file. If you do not specify any value, the default EQQRESLT is used.

6. After you have entered the information on panel EQQJOBSD, click Enter. The dialog process generates several members in the output library that you specified. These members, which are described in Table 16: Sample JCL generated by the EQQJOBS dialog on page 72, will be used at various stages of the installation.

Table 16. Sample JCL generated by the EQQJOBS dialog

Member	Description of job
EQQ9SM01	Updates the RACF® router table (ICHRFR01).
EQQ9SMDE	Updates the RACF® class-descriptor table (ICHRRCDE).
EQQAUDIB	Sample to invoke EQQAUDIT in batch mode outside of the dialog. **Note:** EQQAUDIB is used only if you specified a value for the **EQQTROUT *dsname*** and the **EQQAUDIT *output dsn*** fields in the EQQJOBSA panel.
EQQBENCR	Sample JCL to run the utility that encrypts the Windows™ passwords set in the USRPSW parameter of the USRREC statements.
EQQCONOP	Sample initial parameters for the controller only.
EQQCONO	Sample started task procedure for the controller only.
EQQCONP	Sample initial parameters for a controller and tracker in same address space.
EQQCON	Sample started task procedure for a controller and tracker in same address space.
EQQDBENC	Contains the JCL to encrypt the password in the DBOPT statement.
EQQDBOPT	Sample DBOPT statement.
EQQDPCOP	JCL and usage notes for copy VSAM function.
EQQFLWAT	Sample JCL to call **filewatch** utility to monitor HFS or ZFS file changes.
EQQICNVS	Migrates VSAM files.
EQQJES2	Assembles and link-edits the JES2 EXIT7.

Table 16. Sample JCL generated by the EQQJOBS dialog (continued)

Member	Description of job
EQQJER2U	Restores the EXIT7 as a JES2 usermod.
EQQJER2V	Restores the EXIT51 as a JES2 usermod.
EQQJER3U	Restores the EQQUX191 and EQQUX291 as JES3 usermods.
EQQJER3Z	Restore the EQQUX291 as JES3 usermod.
EQQJES21	Assembles and link-edits the JES2 EXIT51.
EQQJES2U	Installs the JES2 EXIT7 usermod.
EQQJES2V	Installs the JES2 EXIT51 usermod.
EQQJES3	Assembles and link-edits a JES3 exit.
EQQJES3U	Installs the JES3 usermod.
EQQJES3Z	Installs the EQQUX291 as JES3 usermod.
EQQOUC	Sample started task procedure for the Output collector.
EQQOUCH	Sample header template for job logs.
EQQOUCP	Sample initial parameters for the Output collector.
EQQPCS01	Allocates unique data sets within the sysplex.
EQQPCS02	Allocates non-unique data sets.
EQQPCS03	Generates a job that allocates VSAM copy data set.
EQQPCS07	Allocates VSAM data sets for Restart and Cleanup.
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 on page 136 and EQQOUCKP on page 136) 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 to send and restore procedures.
EQQPCS14	Allocates the data sets for the extended auditing.
EQQRECP1	Restore procedure for CX, CP1, XD1, ST files.
EQQRECP2	Restore procedure for CX, CP2, XD2, ST files.

Table 16. Sample JCL generated by the EQQJOBS dialog (continued)

Member	Description of job
EQQRENCP	Restore procedure for NCP, NCX, NXD files.
EQQRESLT	Restore procedure for LTP files.
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.
EQQRAD	Restore procedure for the Application Description.
EQQRJS1	Restore procedure for the primary JCL repository.
EQQRJS2	Restore procedure for the secondary JCL repository.
EQQROI	Restore procedure for the Operator Instructions.
EQQRRD	Restore procedure for the Resource Description.
EQQRSI	Restore procedure for the Side Information file.
EQQRWS	Restore procedure for the Workstation Description.
EQQSAD	Send procedure for the Application Description.
EQQSAMPI	Copies sample databases from the sample library to VSAM data sets.
EQQSECP1	Dump procedure for CX, CP1, XD1, ST files.
EQQSECP2	Dump procedure for CX, CP2, XD2, ST files.
EQQSENCP	Dump procedure for NCP, NCX, NXD files.
EQQSENLTP	Dump procedure for LTP files.
EQQSERP	Sample initial parameters for a Server.
EQQSER	Sample started task procedure for a Server.
EQQSJS1	Send procedure for the primary JCL repository.
EQQSJS2	Send procedure for the secondary JCL repository.
EQQSMF	Updates SMF exits for HCL Workload Automation for Z.
EQQSMLOG	Sample procedure that creates the GDG data set where the outgoing MLOG file is archived when the MLOG switching function takes effect. Uses the EQQREPRO input parameter.
EQQSOI	Send procedure for the Operator Instructions.
EQQSRD	Send procedure for the Resource Description.
EQQSSI	Send procedure for the Side Information file.
EQQSWS	Send procedure for the Workstation Description.

Table 16. Sample JCL generated by the EQQJOBS dialog (continued)

Member	Description of job
EQQTRA	Sample started task procedure for a tracker.
EQQTRAP	Sample initial parameters for a tracker.
EQQTROPT	Sample TRGOPT statement.

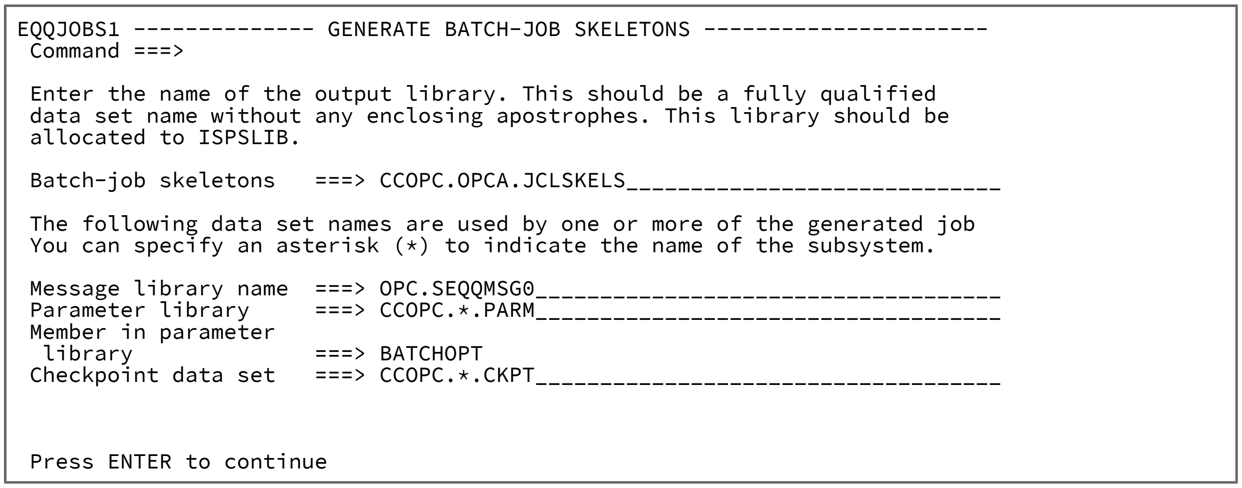
Generating batch-job skeletons

To ensure that all files are correctly allocated, you must generate the batch-job skeletons before creating the sample job JCL.

Several Z controller functions, such as daily planning, are performed by batch jobs that are submitted from the HCL Workload Automation for Z dialog. To generate the skeleton JCL for these jobs:

1. From the EQQJOBS main menu, select option 2. The following panel is displayed:

   Figure 17. EQQJOBS1 - Generate HCL Workload Automation for Z batch-job skeletons



```
EQQJOBS1 -------------- GENERATE BATCH-JOB SKELETONS ----------------------
 Command ===>

 Enter the name of the output library. This should be a fully qualified
 data set name without any enclosing apostrophes. This library should be
 allocated to ISPSLIB.

 Batch-job skeletons   ===> CCOPC.OPCA.JCLSKELS_____________________________

 The following data set names are used by one or more of the generated job
 You can specify an asterisk (*) to indicate the name of the subsystem.

 Message library name  ===> OPC.SEQQMSG0____________________________________
 Parameter library     ===> CCOPC.*.PARM____________________________________
 Member in parameter
  library              ===> BATCHOPT
 Checkpoint data set   ===> CCOPC.*.CKPT____________________________________


 Press ENTER to continue
```

Batch-job skeletons

Required. Name of the library where the JCL skeletons are to be stored. Before you use the HCL Workload Automation for Z dialog to submit batch jobs, allocate this library to the ISPSLIB DD statement in the TSO session of dialog users.

For detailed information about how you set up the dialog, see The ISPF environment on page 192. You can create a new library for the skeleton JCL members or put them in an existing skeleton-JCL library.

In the following fields, you can enter &XOPCNM. as one of the qualifiers for the data set names. This is an ISPF variable that is stored in the profile and is the same variable that you specify in option 0.1 (SUBSYSTEM NAME) in the HCL Workload Automation for Z dialogs. When a skeleton is then used by a dialog of the scheduler, &XOPCNM. is substituted with the name of the scheduler subsystem that is being used.

Ensure that &XOPCNM. ends with a period if it is not the low-level qualifier. For example, you could enter `CCOPC.&XOPCNM..PARMLIB` but `CCOPC.&XOPCNM.PARMLIB` results in a JCL error.

If you enter an asterisk (*) as a data set qualifier, the generated skeletons will contain &XOPCNM. in place of the asterisk.

Message library name

Required. Name of the library that contains the HCL Workload Automation for Z messages (SMP/E target DD name SEQQMSG0).

Parameter library

Required. Name of the library where to store the initialization statements.

Member in parameter library

Required. Name of a member in the parameter library where to store the BATCHOPT initialization statement. The HCL Workload Automation for Z batch jobs will use this member. If you have not already created the BATCHOPT statement, you can still generate the batch skeletons, but remember to create a member with the same name when you create the initialization statements.

Checkpoint data set

Required. Name of the checkpoint data set.

2. Click Enter. The following panel is displayed:

Figure 18. EQQJOBS2 - Generate HCL Workload Automation for Z batch-job skeletons

```
EQQJOBS2 -------------- GENERATE BATCH-JOB SKELETONS ----------------------
 Command ===>

 Enter the following required job stream parameters:
 Non-VSAM dsn prefix ===> CCOPC.*__________________


 VSAM dsn prefix     ===> CCOPC.*V_________________
 Unit name           ===> 3390____      Default unit name
 Unit name (temp ds) ===> SYSDA___      Unit name for temporary data sets
 Unit name (sort ds) ===> SYSDA___      Unit name for sort work data sets
 SYSOUT class        ===> *             SYSOUT class for reports

 The following information is optional:

 STEPLIB dsname        ===> OPC.SEQQLMD0__________________________________
 STEPCAT dsname        ===> ______________________________________________
 EQQMLOG dsname        ===> CCOPC.*.MLOGBAT
---------------------------------

 The following information is REQUIRED WITH DBCS support:

 KJSRTBL dsname        ===> ______________________________________________

 Press ENTER to generate OPC batch-job skeletons
```

Non-VSAM dsn prefix

Required. Enter the qualifiers that prefix the non-VSAM data set names. HCL Workload Automation for Z adds a low-level qualifier to the prefix to uniquely identify the non-VSAM data sets. For example, it adds JTARC for the job-tracking archive data set. If the subsystem name is OPCA, the data set name will be CCOPC.OPCA.JTARC when the skeleton is used by the dialogs.

VSAM dsn prefix

Required. Enter the qualifiers to prefix the VSAM data set names. HCL Workload Automation for Z adds a low-level qualifier to the prefix to uniquely identify each HCL Workload Automation for Z VSAM data set. For example, it adds WS for the workstation description data set. If the subsystem name is OPCA, the data set name will be CCOPC.OPCAV.WS when the skeleton is used by the dialogs.

Unit name

Required. Enter a device name that is valid at your installation. This can be a device type, for example 3380, or a group name, for example PROD or TEST.

Unit name (temp ds)

Required. Enter a device name that can be used for temporary data sets.

Unit name (sort ds)

Required. Enter a device name that can be used for sort-work data sets.

SYSOUT class

Required. Specify the SYSOUT class that you want to use for the reports that are generated by the batch jobs.

STEPLIB dsname

Optional. Name of the HCL Workload Automation for Z load module library if the load modules are not in a data set included in an active LNKLST member.

STEPCAT dsname

Optional. Name of a private catalog if one or more data sets cannot be reached through the master catalog. To customize the EQQAUDNS skeleton clist with the appropriate loadlib that should be referenced when audit/debug is invoked, you must specify the dsname.

EQQMLOG dsname

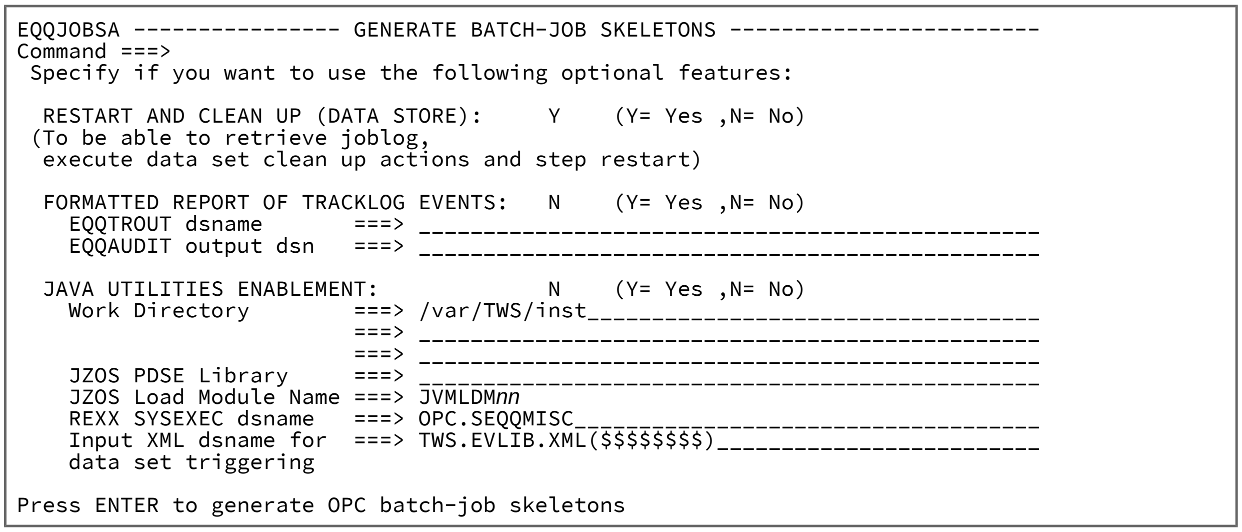
Optional. Name of a message log data set if messages are not sent to SYSOUT. This must not be the same data set that is used by a tracker, controller, or standby controller.

KJSRTBL dsname

Required if you use the Japanese language feature. Name of the data set to be used when sorting fields containing DBCS data.

3. Click Enter. The following panel is displayed:

Figure 19. EQQJOBSA - Generate HCL Workload Automation for Z batch-job skeletons



```
EQQJOBSA ---------------- GENERATE BATCH-JOB SKELETONS ------------------------
Command ===>
 Specify if you want to use the following optional features:

  RESTART AND CLEAN UP (DATA STORE):      Y    (Y= Yes ,N= No)
 (To be able to retrieve joblog,
  execute data set clean up actions and step restart)

  FORMATTED REPORT OF TRACKLOG EVENTS:   N    (Y= Yes ,N= No)
    EQQTROUT dsname       ===> ________________________________________________
    EQQAUDIT output dsn   ===> ________________________________________________

  JAVA UTILITIES ENABLEMENT:             N    (Y= Yes ,N= No)
    Work Directory        ===> /var/TWS/inst___________________________________
                          ===> ________________________________________________
                          ===> ________________________________________________
    JZOS PDSE Library     ===> ________________________________________________
    JZOS Load Module Name ===> JVMLDMnn
    REXX SYSEXEC dsname   ===> OPC.SEQQMISC____________________________________
    Input XML dsname for  ===> TWS.EVLIB.XML($$$$$$$$)_________________________
    data set triggering

Press ENTER to generate OPC batch-job skeletons
```

RESTART AND CLEAN UP (Data Store)

Specify Y if you want to use the Restart and Cleanup feature.

FORMATTED REPORT OF TRACKING EVENTS

Specify Y if you want to use the feature that produces a formatted report of the tracklog events.

EQQTROUT dsname

Optional. Name of the data set in which DP Extend and Replan writes tracklog events. Leave blank if you want the corresponding DD card for these jobs to specify DUMMY as in previous releases. Type out if you plan to use sample EQQAUDIB (see Table 16: Sample JCL generated by the EQQJOBS dialog on page 72).

EQQAUDIT output dsn

Name of the data set where the EQQAUDIT output is to be written. Required if FORMATTED REPORT OF TRACKLOG EVENTS is set to Y.

Work Directory

Directory where the subsystem specific HFS or ZFS files are stored. Each subsystem supporting the JAVA utility must have its own work directory.

JZOS PDSE Library

The PDSE containing the JZOS Batch Launcher JVMLDM module.

JZOS Load Module Name

Specify JVMLDM*nn*, meaning the JZOS Batch Launcher load module name used with the 64-bit SDK for z/OS® supported version.

REXX™ SYSEXEC dsname

The installation SEQQMISC library containing the REXX™ programs EQQRXARC and EQQRXTRG.

Input XML dsname for data set triggering

Name of the input data set containing the event rules in XML format used to produce the data set triggering configuration files that will be sent to trackers. The default data set is provided in the panel.

Note: In controller MLOG dsn, EQQTROUT dsname, and EQQAUDIT output dsn you can use an asterisk (*) for the subsystem name. It will be replaced with the current subsystem name when the dialog is invoked.

4. After you have entered the information on panel EQQJOBSA, click Enter. The dialog generates the batch-job skeleton members.

   After completing this procedure, you can proceed with the creation of the sample job JCL, as described in Creating the sample job JCL on page 65.

If you are not sure at this stage what some of the values will be, it does not matter. You can rerun the dialog as many times as you want to regenerate the skeletons. You can also edit the generated skeletons manually.

Table 17: Controller skeleton JCL generated by the EQQJOBS dialog on page 79 shows the JCL skeleton members generated by EQQJOBS:

Table 17. Controller skeleton JCL generated by the EQQJOBS dialog

Member	Batch job description
EQQADCDS	Application cross-reference of conditional dependencies.
EQQADCOS	Calculate and print run dates of an application.
EQQADDES	Application cross-reference of external dependencies.
EQQADPRS	Application print program.
EQQADXRS	Application cross-reference program.
EQQADX1S	Application cross-reference of selected fields.
EQQAMUPS	Application description mass update.
EQQAPARS	Procedure to gather diagnostic information.
EQQAUDIS	Extract and format job tracking events (batch invocation).
EQQAUDNS	Extract and format job tracking events (interactive invocation) **Note:** Ensure to copy this member from the library where it was created by EQQJOBS into a procedure library. This step is required since this member must be invoked interactively.
EQQDBARS	Daily Planning - Historical run data archiver for Dynamic Workload Console reporting feature.

Table 17. Controller skeleton JCL generated by the EQQJOBS dialog (continued)

Member	Batch job description
EQQDPEXS	Daily planning - plan next period.
EQQDPPRS	Daily planning - print current period results.
EQQDPRCS	Daily planning - replan current period.
EQQDPSJS	Daily planning -DBCS sort step.
EQQDPSTS	Daily planning - normal sort step.
EQQDPTRS	Daily planning - plan a trial period.
EQQJVPRS	Print JCL variable tables.
EQQLEXTS	Long-term planning - extend the long-term plan.
EQQLMOAS	Long-term planning - modify all occurrences.
EQQLMOOS	Long-term planning - modify one occurrence.
EQQLPRAS	Long-term planning - print all occurrences.
EQQLPRTS	Long-term planning - print one occurrence.
EQQLTRES	Long-term planning - create the long-term plan.
EQQLTRYS	Long-term planning - trial.
EQQOIBAS	Operator instructions - batch program.
EQQOIBLS	Operator instructions - batch input from a sequential data set.
EQQTPRPS	Print periods.
EQQTPRTS	Print calendars.
EQQTRBLS	Event driven workload automation - Create configuration files for data set triggering
EQQWPRTS	Print workstation descriptions.

Generating Data Store samples

To create the Data Store samples, perform the following steps:

1. From the EQQJOBS application menu, select option 3. The following panel is displayed:

   Figure 20. EQQJOBS5 - Create Data Store samples

```
EQQJOBS5 ----------------- CREATE DATA STORE SAMPLES --------------------
Command ===>

The data set names specified on this panel should be fully qualified
names without any enclosing apostrophes.

Enter the name of the output library:

Sample job JCL        ===> CCOPC.OPCA.JCLDS__________

Job statement information:

===> //SYSPROG1 JOB (111111,2222),'OPCESA BATCH',CLASS=B,MSGCLASS=H,__
===> //         MSGLEVEL=(1,1),NOTIFY=SYSPROG______________________________
===> ______________________________________________________________________
===> ______________________________________________________________________

The following data set names are used by one or more of the generated jobs.
Message library name  ===> OPC.SEQQMSG0_______________________________
Parameter library     ===> CCOPC.OPCEDS.PARM__________________________


Press ENTER to continue
```

   Sample job JCL

   Required. Name of the library to which you want the generated Data Store samples written. The library must be allocated before you generate the Data Store samples. Ensure that the library that you specify has sufficient directory blocks to store all the sample members that are generated by EQQJOBS (see Table 18: Data Store samples generated by the EQQJOBS dialog on page 84).

   Job statement information

   Required. The JOB statement that follows standard JCL syntax and your installation standards.

   Message library name

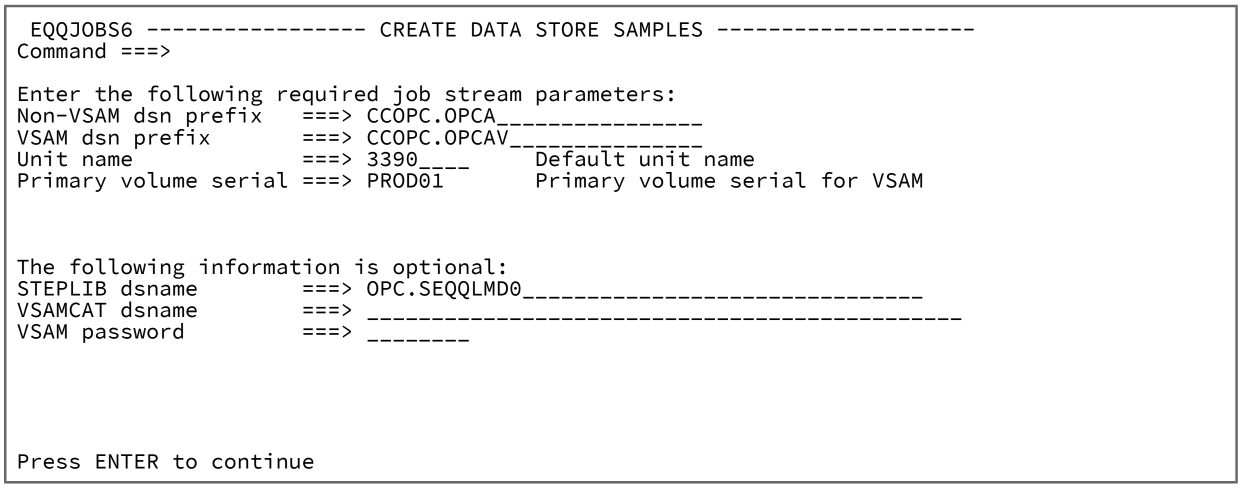
   Required. Name of the library where to store the scheduler messages (SMP/E target DDNAME SEQQMSG0).

   Parameter library

   Required. Name of the library where to store the initialization statements. Use a name different from that of the controller and tracker parameter library. Ensure that this library is allocated.

2. Click Enter. The following panel is displayed:

Figure 21. EQQJOBS6 - Create Data Store samples



```
 EQQJOBS6 ----------------- CREATE DATA STORE SAMPLES --------------------
Command ===>

Enter the following required job stream parameters:
Non-VSAM dsn prefix   ===> CCOPC.OPCA_______________
VSAM dsn prefix       ===> CCOPC.OPCAV______________
Unit name             ===> 3390____     Default unit name
Primary volume serial ===> PROD01       Primary volume serial for VSAM



The following information is optional:
STEPLIB dsname        ===> OPC.SEQQLMD0________________________________
VSAMCAT dsname        ===> ______________________________________________
VSAM password         ===> ________




Press ENTER to continue
```

Non-VSAM dsn prefix

Required. The qualifiers that prefix the non-VSAM data set names. The scheduler adds a low-level qualifier to the prefix to uniquely identify each Data Store non-VSAM data set.

VSAM dsn prefix

Required. The qualifiers that prefix the VSAM data set names. The scheduler adds a low-level qualifier to the prefix to uniquely identify each Data Store VSAM data set.

Unit name

Required. A device name that is valid at your installation. This could be a device type, for example 3390, or a group name, for example PROD or TEST.

Primary volume serial

Required. The volume that will be used by sample job EQQPCS04 to allocate the primary data sets.

STEPLIB dsname

Optional. Name of the scheduler load module library, if the load modules are not in a data set included in an active LNKLST member.

VSAMCAT dsname

Optional. Name of a catalog in which VSAM data sets are to be defined, if they are not to be defined in the master catalog.

VSAM password

Optional. The password for the VSAM catalog, if the catalog is password-protected.

3. Click Enter. The following panel is displayed:

Figure 22. EQQJOBS7 - Create Data Store samples

```
EQQJOBS7 ----------------- CREATE DATA STORE SAMPLES ---------------------
 Command ===>

 Enter the parameters to build DSTOPTS and DSTUTIL options samples:

 Reserved destination          ===> OPCX____
 Connection type               ===> SNA (SNA/XCF/TCP)
   SNA Data Store luname       ===> I9PC45AA (only for SNA connection)
   SNA Controller luname       ===> I9PC45RA (only for SNA connection)
   Xcf Group                   ===> ________ (only for XCF connection)
   Xcf Data Store member       ===> ________ (only for XCF connection)
   Xcf FL task member          ===> ________ (only for XCF connection)
   TCP Controller host name:                 (only for TCP connection)
   ===> ____________________________________________________
   TCP Controller port number  ===> _____    (only for TCP connection)
 Jobdata ret. period           ===> 2_ (number of days)
 JobLog retrieval              ===> Y (Y/N)
   Max n. lines to store       ===> 0______
   JobLog  ret. period         ===> 5_ (number of days)

 Press ENTER to create sample job JCL
```

Reserved destination

Required. The Data Store reserved output destination. It must correspond to the DSTDEST parameter set in RCLOPTS option.

Connection Type

Required. The connection method used to establish the communication between FN/FL tasks and Data Store. It can be TCP, SNA, or XCF.

SNA Data Store luname

The Data Store VTAM® application name, if SNA connection type has been set.

SNA Controller luname

The controller FN task VTAM® application name, if SNA connection type has been set.

Xcf Group

Name of XCF group, if XCF connection type has been chosen.

Xcf Data Store member

Name of Data Store XCF member, if XCF connection type has been chosen.

Xcf FL task member

Name of the FL task XCF member, if XCF connection type has been chosen.

Job data retention period

The Data Store structured information retention period. It consists of the interval in days used by the online cleanup and is necessary to be able to use the Restart and Cleanup feature.

Joblog retrieval

Specify if the joblog retrieval must be enabled. This means that the Data Store will save the unstructured data in the joblog.

Max n. of lines to store

The maximum number of user sysout lines to be stored. The range is 0 to 10000.

Joblog retention period

The Data Store unstructured information retention period. It consists of the interval in days used by the online cleanup and is necessary to enable the Joblog Browse function.

TCP Controller host name

The controller TCP/IP host name, if you set TCP in Connection type.

TCP Controller port number

The controller TCP/IP port number, if you set TCP in Connection type.

4. Click Enter after you have entered the information on panel EQQJOBS7. The dialog now generates several members in the output library that you specified. These members, which are described in Table 18: Data Store samples generated by the EQQJOBS dialog on page 84, are used at various stages of the installation:

Table 18. Data Store samples generated by the EQQJOBS dialog

Member	Sample Description
EQQCLEAN	Sample procedure invoking EQQCLEAN program
EQQDSCL	Batch Clean Up sample
EQQDSCLP	Batch Clean up sample parameters
EQQDSEX	Batch Export sample
EQQDSEXP	Batch Export sample parameters
EQQDSIM	Batch Import sample
EQQDSIMP	Batch Import sample parameters
EQQDSRG	Batch sample reorg
EQQDSRI	Batch Recovery index
EQQDSRIP	Batch Recovery index parameters
EQQDST	Sample procedure to start Data Store
EQQDSTP	Parameters for sample procedure to start Data Store
EQQPCS04	Allocate VSAM data sets for Data Store

Creating the Workload Automation Programming Language samples

To create the Workload Automation Programming Language samples, perform the following steps:

1. From the EQQJOBS application menu, select option 4. The following panel is displayed:

   Figure 23. EQQJOBSE - Create WAPL samples

```
EQQJOBSE ---------------------- CREATE WAPL SAMPLES ----------------------------
Command ===>

The data set names specified on this panel must be fully qualified
without any enclosing apostrophes.

Enter the name of the output library:

Sample job JCL        ===> TWS.INST.EQQJOBS________________________________

The following data set names are used by one or more of the generated job
WAPL data maps        ===> TWS.INST.SEQQWAPL_______________________________
MISC library name     ===> TWS.INST.SEQQMISC_______________________________
Message library name  ===> TWS.INST.SEQQMSG0_______________________________
Steplib               ===> ________________________________________________
REXX libraries        ===> ________________________________________________
ISPF Messages         ===> ISP.SISPMENU____________________________________
ISPF Panels           ===> ISP.SISPMENU____________________________________
ISPF Skeletons        ===> ISP.SISPMENU____________________________________
ISPF Tables           ===> ISP.SISPMENU____________________________________
SYSOUT Class of Internal Reader   ===> _
CSSMTP External Writer            ===> ____________________________________

Subsys  ===> TWSR___   Language  ===> EN__   Version  ===> ________________

Press ENTER to continue
```

   The data set names that you specify on this panel must be completely qualified and not be enclosed within quotes.

   Sample job JCL

   Required. The complete name of the library where to store the generated Workload Automation Programming Language samples. The library must be allocated before you generate the Workload Automation Programming Language samples. Ensure that the library has enough directory blocks to store all the sample members that are generated by EQQJOBS (for details, see Table 19: Workload Automation Programming Language samples generated by the EQQJOBS dialog on page 86).

   WAPL data maps

   Required. The installation SEQQWAPL library containing the Workload Automation Programming Language data maps.

   MISC library

   Required. The installation SEQQMISC library containing the REXX Workload Automation Programming Language programs.

   Message library name

   Required. Name of the library where to store the HCL Workload Automation for Z messages (SMP/E target DDNAME SEQQMSG0).

   Steplib

   Optional. Name of the HCL Workload Automation for Z load module library, if the load modules are not in a data set included in an active LNKLST member.

REXX Libraries

Optional. Name of the IBM Compiler Library for REXX/370. This is either the compiler run time library (SEAGLPA) if you have the REXX compiler installed, or the REXX alternative library (SEAGALT) if you do not have the compiler installed. You must specify the name of the REXX library only if neither of these libraries are included in an active LNKLST member.

Note: Workload Automation Programming Language processing is significantly faster with the SEAGLPA library.

ISPF Messages

The ISPF messages library.

ISPF Panels

The ISPF panels library.

ISPF Skeletons

The ISPF skeletons library.

ISPF Tables

The ISPF tables library.

SYSOUT Class of Internal Reader

Class of the internal reader for SMTP emails.

CSSMTP External Writer

Writer for SMTP emails through z/OS.

Subsys

The HCL Workload Automation for Z subsystem.

Language

The Workload Automation Programming Language language. Only English (EN) is supported.

Version

The HCL Workload Automation for Z version.

Table 19. Workload Automation Programming Language samples generated by the EQQJOBS dialog

Member	Sample Description
EQQILSON	Workload Automation Programming Language procedure to load the output file into ISPF.
EQQYXJPL	Procedure to run the EQQWAPL load module.
EQQYXJPX	Workload Automation Programming Language for z/OS batch procedure.

Table 19. Workload Automation Programming Language samples generated by the EQQJOBS dialog (continued)

Member	Sample Description
EQQWCMD1	Use only on a started-task (STC) workstation to run Workload Automation Programming Language commands for anHCL Workload Automation for Z system configured with OPCOPTS RCLEANUP(YES).
EQQWCMD2	Use only on a started-task (STC) workstation to run Workload Automation Programming Language commands for anHCL Workload Automation for Z system configured with OPCOPTS RCLEANUP(NO).
EQQWTSO1	Sets up the Workload Automation Programming Language environment for TSO, runs Workload Automation Programming Language commands, processes the result, and resets the environment.
EQQWTSO2	This sample assumes that the Workload Automation Programming Language environment is already set up in the TSO LOGON procedure or startup REXX/CLIST. It runs commands and then processes the results.
EQQWTSO3	Sets up the Workload Automation Programming Language environment for TSO, to be called by another REXX script that sets Workload Automation Programming Language commands and then processes the results.
EQQWTSO4	This sample assumes that the Workload Automation Programming Language environment is already setup in the TSO LOGON procedure or startup REXX/CLIST. It is called by another REXX that sets Workload Automation Programming Language commands and then processes the results.
EQQWTSX3	Calls another REXX script that sets up the Workload Automation Programming Language environment to run the commands specified here.
EQQWTSX4	Calls another REXX script that assumes that the Workload Automation Programming Languageenvironment has already been set up in the TSO LOGON procedure or startup CLIST. Then it runs the commands specified here.

Step 5. Adding SMF and JES exits for event tracking

Perform this task if you are installing a tracker.

HCL Workload Automation for Z tracks the progress of jobs and started tasks through the z/OS system by using JES and SMF exit points. Add all these exits on each z/OS system where you will start HCL Workload Automation for Z.

To simplify the installation of HCL Workload Automation for Z event tracking, several sample event-tracking exits can be found in your sample library, SEQQSAMP. To assemble and install exits, you can use the sample JCL provided to install the exits as SMP/E *usermods* or alternatively you can assemble and link-edit the exits yourself. For JES exits, apply *usermods* in the CSI where JES is included: this is the best method. It has the advantage that SMP automatically reassembles the exits if maintenance is applied to the JES control blocks that HCL Workload Automation for Z is dependent on.

If you install a new release of HCL Workload Automation for Z in a new CSI, and the JES usermod is already installed in the same CSI as a previous release, follow these steps:

1. Apply any necessary tolerance PTFs so that the previous release can run with the new exit code.
2. Change the DDDEFs for JES so that they point to the SEQQSAMP and SEQQMAC0 libraries of the *new* release.
3. APPLY REDO the JES usermod. This reassembles the exits with the new code.

The sample exits all use the EQQEXIT macro to create event-generating code. For more information on the EQQEXIT macro, see Invoking the EQQEXIT macro on page 357.

Table 20: Sample exits for HCL Workload Automation for Z on page 88 describes the samples that you can use to generate and install the exits. The sample exit, skeleton JCL, and *usermod* entries identify members of the SEQQSAMP library. The event types in the table are prefixed with A for JES2 or B for JES3, when they are created by the exit. (SeeVerifying tracking events on page 172 for more information about event types.)

Table 20. Sample exits for HCL Workload Automation for Z

Exit name	Exit type	Sample exit	Sample JCL/ usermod	Event supported	Event type
IEFACTRT	SMF	EQQACTR1	EQQSMF	Job and step completion	3J,3S
IEFUJI	SMF	EQQUJI1	EQQSMF	Job start	2
IEFU83	SMF	EQQU831	EQQSMF	End of print group, purge (JES3 only), data set triggering support, and automatic change support	4,5,S,T
EXIT7	JES2	EQQXIT74	EQQJES2/ EQQJES2U	JCT I/O exit for JES2, purge	1,3P,5
EXIT51	JES2	EQQXIT51	EQQJES21/ EQQJES2V	JES2 QMOD phase change exit	1
IATUX19	JES3	EQQUX191	EQQJES3/ EQQJES3U	Output processing complete	3P
IATUX29	JES3	EQQUX291	EQQJES3/ EQQJES3U	On job queue	1

SMF only

The EQQU831 sample generates type 4 and type 5 events and also generates resource availability events when a data set is closed after read or update processing. For more information, see Implementing support for data set triggering on page 100.

You must tailor the sample JCL to the requirements of your installation. If you have already run EQQJOBS, tailored versions of the JCL will already exist in the EQQJOBS output library. Alternatively, you can copy any of the members from the SEQQSAMP library to one of your own libraries and manually tailor the JCL.

For detailed information about how to activate SMF exits, see Updating SMF parameters on page 94 and the documentation for SMF.

JES2 only

The load module names are the same as the exit names, except for JES2. The load module of the JES2 exits, which are EXIT7 and EXIT51, are called OPCAXIT7 and TWSXIT51, and their entry points are called OPCAENT7 and TWSENT51, respectively.

If your z/OS system is a JES2 system, include these records in the JES2 initialization member:

JES2 Initialization Statements

Example

```
LOAD(OPCAXIT7) /*
Load HCL Workload Automation for Z exit mod */
EXIT(7) ROUTINES=OPCAENT7,STATUS=ENABLED /*
Define EXIT7 entry point */
```

Example

```
LOAD(TWSXIT51) /*
Load HCL Workload Automation for Z exit mod */
EXIT(51) ROUTINES=TWSENT51,STATUS=ENABLED /*
Define EXIT51 entry point */
```

To dynamically install the JES2 exits for HCL Workload Automation for Z, use these commands once the modules are available in the LNKLST:

```
$ADD LOADMOD(OPCAXIT7),STORAGE=PVT
```

```
$T EXIT(7),ROUTINES=OPCAENT7,
 STATUS=ENABLED
```

```
$ADD LOADMOD(TWSXIT51),STORAGE=PVT
```

```
$T EXIT(51),ROUTINES=TWSENT51,
 STATUS=ENABLED
```

To put a new version of an exit (that was previously installed) in place, use these commands once the modules are available in the LNKLST:

```
$TLOADMOD(OPCAXIT7),REFRESH
$TLOADMOD(TWSXIT51),REFRESH
```

For more information on JES2 initialization statements, see *JES2 Initialization and Tuning Reference*.

JES3 only

To activate the exits for a JES3 system, you can link them to a library that is concatenated ahead of SYS1.JES3LIB. Alternatively, you can replace the existing exits in SYS1.JES3LIB with the HCL Workload Automation for Z-supplied IATUX19 and IATUX29 exits. For more information, see *JES3 Initialization and Tuning Reference*. If you get RC=4 and the warning ASMA303W Multiple address resolutions may result when you assemble IATUX19 running the EQQJES3/EQQJES3U sample, you can ignore the message. If version ASMA90 of the compiler reports errors, and the RMODE=ANY statement is defined, remove the RMODE=ANY statement from the sample exit.

Step 6. Updating SYS1.PARMLIB

The following topics describe the updates to SYS1.PARMLIB for your environment:

- Defining subsystems on page 90
- Calculating MAXECSA values on page 92
- Authorizing the load-module library on page 94
- Updating SMF parameters on page 94
- Updating z/OS dump options on page 97
- Updating the z/OS link-library definition on page 97
- Updating XCF initialization options on page 98
- Modifying TSO parameters on page 98
- Performance considerations on page 100
- Defining the DLF exit for Hiperbatch support on page 100
- Starting the product automatically on page 100
- Updating APPC options on page 100
- Implementing support for data set triggering on page 100

Defining subsystems

When you define the subsystem names of the HCL Workload Automation for Z controllers and trackers, consider the following:

- The Subsystem/STC name of HCL Workload Automation for Z controllers is unique within the PLEX. If two different controllers (regardless of their location) are configured to track work on the same z/OS® system, they must have different Subsystem/STC names.
- Because subsystem names on a given LPAR must be unique, and because all HCL Workload Automation for Z trackers and controllers started tasks must have the same name as their associated subsystems, all started tasks on any given LPAR must have unique names. That is, inside a z/OS image, controllers and trackers must have unique Subsystem/STC names.
- Trackers running on different LPARs but connected to the same controller can have the same Subsystem/STC name. In this case, system variables like &SYSNAME can be used with the condition that each tracker uses different HCL Workload Automation for Z data sets. The tracker name cannot be the same as the name of a controller.

You must define the name of every new HCL Workload Automation for Z subsystem in the active subsystem-name-table member of SYS1.PARMLIB. Install at least two HCL Workload Automation for Z controlling systems, one for testing and one for your production environment.

Note: It is recommended that you install the tracker and the Z controller in separate address spaces on the controlling system.

To define the subsystems, update the active IEFSSN*nn* member in SYS1.PARMLIB. Include records as in the following example:

Example

```
Subsystem definition record
SUBSYS SUBNAME(subsystem name) INITRTN(module name) INITPARM ('maxecsa,suffix')
```

subsystem name

The name assigned to an HCL Workload Automation for Z subsystem. The name must be from 2 to 4 characters. All the subsystem names, as defined in the SYS1.PARMLIB member IEFSSN*nn*, must be unique within a GRS complex with the exception of a standby controller. Also, the subsystem names of the controllers must be unique within your OPCplex/OPC network, both local and remote systems. HCL Workload Automation for Z requires the started task name or job name used for an HCL Workload Automation for Z address space to exactly match the name of the associated subsystem.

module name

The name of the subsystem initialization module, EQQINIT2.

maxecsa

Defines the maximum amount of extended common service area (ECSA) that is used to queue job-tracking events. The value is expressed in kilobytes (1 KB equals 1024 bytes). The default is 4, which means that a maximum of 4 KB (4096 bytes) of ECSA storage is needed to queue job-tracking events. The maximum value allowed for MAXECSA is 2816.

suffix

The module name suffix for the EQQSSCM module that EQQINIT2 loads into common storage. EQQSSCM is the subsystem communication module. The suffix must be a single character. Because the name of the module shipped with HCL Workload Automation for Z is EQQINIT2, specify a suffix value of 2. If you do not provide a suffix, EQQINIT2 attempts to load module name EQQSSCM2. You can also specify a subsystem communication module name in the SSCMNAME keyword of the OPCOPTS initialization statement to load an updated version of the module before a scheduled IPL. For details, see *Customization and Tuning*.

For more information about the EQQSSCM modules, Updating the z/OS link-library definition on page 97.

The following example illustrates a record that you can include in the SYS1.PARMLIB IEFSSN*nn* member. The record defines an HCL Workload Automation for Z subsystem called OPCA. This represents a tracker. Its initialization module is EQQINIT2.

The amount of ECSA that is allocated, 101104 bytes, is enough for 1136 job-tracking events. Because suffix value 2 is specified, EQQINIT2 loads module EQQSSCM2.

Example

```
/*Subsystem definition example*/
SUBSYS SUBNAME(OPCA) INITRTN(EQQINIT2) INITPARM ('100,2')
```

Note: In addition to updating the IEFSSN*nn* member, you can temporarily use the SETSSI commands to update subsystem information. For example, to delete a subsystem issue the following command:

```
setssi delete,s=XXXX,force
```

To add a new subsystem, issue the following command:

```
setssi add,s=XXXX,i=EQQINITX,p='100,2'
```

Calculating MAXECSA values

HCL Workload Automation for Z allocates ECSA storage for job-tracking events in blocks of 1424 bytes. Each block is equivalent to 16 events. Table 21: Examples of MAXECSA storage values on page 92 gives examples of the storage needed for, the storage actually allocated, and the events accommodated for several **maxecsa** values. The number of events created for each job or started task in your environment is influenced by the definitions in the EWTROPTS initialization statement. Every job or started task creates a minimum of six events. If the job or started task generates output and PRINTEVENTS(ALL) or PRINTEVENTS(END) is specified, an event is created when each output group is purged. If STEPEVENTS(ALL) is specified, an event is created for every step in the job or started task.

If you want to calculate values that are not shown in Table 21: Examples of MAXECSA storage values on page 92 for a given MAXECSA value, use this method:

- Space requested = MAXECSA * 1024
- Blocks = space requested / 1424 (round down to a whole number)
- Space allocated = blocks * 1424
- Events accommodated = blocks * 16

Table 21. Examples of MAXECSA storage values

MAXECSA value	Amount of MAXECSA space requested	Blocks of ECSA space allocated (bytes)	Number of events accommodated
0	0	0 (0)	0
4	4096	2 (2848)	32
8	8192	5 (7120)	80
16	16384	11 (15664)	176

Table 21. Examples of MAXECSA storage values (continued)

MAXECSA value	Amount of MAXECSA space requested	Blocks of ECSA space allocated (bytes)	Number of events accommodated
36	36864	25 (35600)	400
72	73728	51 (72624)	816
100	102400	71 (101104)	1136
200	204800	143 (203632)	2288
400	409600	287 (408688)	4592
500	512000	359 (511216)	5744

Note:

1. Allocate enough ECSA storage so that job-tracking events are not lost when the HCL Workload Automation for Z event-writer subtask is not active. When the event writer is active, the number of queued events in ECSA is almost always 0. Allocate enough ECSA for the maximum amount of time you expect the event writer to be inactive.

   For example, after the IPL of a z/OS system, job-tracking events can occur before the tracker address space has become active. If you expect a maximum of 50 events to occur during this time, you should set a MAXECSA value of 8, as shown in Table 21: Examples of MAXECSA storage values on page 92. When the event writer becomes active, the queued events are processed and removed from ECSA.

   If events are lost, message EQQZ035E is stored to the message log. For a description of this message, see *Messages and Codes*.

2. ECSA storage for job-tracking events is required only if the started task includes an event-writer subtask. On a controlling system, you can have one address space running only an event writer subtask, and another one running the Z controller functions and the remaining tracker functions. In this situation, you must specify a MAXECSA value of 0 for the subsystem that contains the Z controller functions.
3. To submit only trackers that have OPCOPTS EWTRTASK(NO) set, the MAXECSA value must also be 0. Do not define subsystems for trackers that will never be started, because this causes waste of ECSA storage. Even if performance monitors might show that a tracker has more ECSA than the value specified by the MAXECSA parameter, this extra ECSA belongs to other subsystems, and the total amount of ECSA used

by HCL Workload Automation for Z for trackers never exceeds the total number of subsystems defined multiplied by their MAXECSA value.

4. All ECSA storage is allocated above the 16 MB line.

Authorizing the load-module library

You must update the active authorized-program-facility member (IEAAPF*nn*, or PROG*nn*) to authorize the load-module library. Each record, except the last, ends with a comma. For the following example, assume that you have installed HCL Workload Automation for Z load modules in the data set OPC.SEQQLMD0 and that this data set is on volume ABC123. To authorize this library, insert this record before the last entry in the IEAAPF*nn*:

```
OPC.SEQQLMD0     ABC123,
```

or update the PROG*nn* member.

Note: Libraries that are defined in the IEAAPF*nn* or PROG*nn* member are authorized only if they remain on the volume specified. If DFHSM is used in your system, change DFHSM parameters so that the new authorized library is not migrated by DFHSM.

Updating SMF parameters

The SMFPRM*nn* member defines parameters for the System Management Facilities (SMF). You must verify that the active SMF parameter member, SMFPRM*nn*, specifies that all SMF exits used by HCL Workload Automation for Z event tracking are activated, and that the required SMF records are being collected. If this is not the case, you must update the active SMF parameter member. Event tracking requires these SMF exits:

IEFUJI

Job initiation exit

IEFACTRT

Job-end and step-end exits

IEFU83

Record write exit. It is optional, and required only for data set triggering, automatic time change, and print event functions.

HCL Workload Automation for Z uses the following SMF record types:

6

For PRINT (A4 and B4) events, used only for tracking work on PRINT workstations

14

Only for data set triggering with SRREAD=YES

15

For data set triggering with SRREAD=YES or SRREAD=NO

18

Only if you want to monitor renaming data sets.

26

For all job tracking

30

For all job tracking

64

Only for data set triggering with VSAM data sets

90

Only if you want automatic daylight savings time change

HCL Workload Automation for Z requires more SMF records to be collected if you install the SMF IEFU83 exit with SRREAD set to YES on the EQQEXIT invocation. Specify this if you want special resource availability events automatically generated when a data set is closed after being opened for:

- Read processing
- Output processing
- Either read or output processing

These SMF records are needed:

- Type 14 records are required for non-VSAM data sets opened for INPUT or RDRBACK processing.
- Type 15 records are required for non-VSAM data sets opened for output.
- Type 64 records are required for VSAM data sets.
- Type 90 records support daylight savings time automatically (optional).

You can specify that the SMF records used by the exit are not written to the SMF log. If your installation does not currently collect SMF records 14, 15, or 64, but you want resource availability events automatically generated, change the EQQU831 sample so that these records are not written to the SMF log.

To avoid data set triggering, and thus to improve performance, specify SRREAD=NO in the IEFU83 SMF exit on invocation of the EQQEXIT macro. The SRREAD=NO parameter prevents data set triggering for only SMF record type 14.

Active exits are defined by the EXITS parameter of the SYS and SUBSYS keywords. An example of these keywords is:

Example

```
/*SYS and SUBSYS keywords*/
SYS(TYPE(6,14,15,18,26,30,60,62,64,90),EXITS(IEFU83,IEFACTRT,IEFUJI))
 SUBSYS(STC,EXITS(IEFUJI,IEFACTRT,IEFU83))
 SUBSYS(JESn,EXITS(IEFUJI,IEFACTRT,IEFU83))
```

Note:

1. JES*n* is either JES2 or JES3. This parameter does not refer to JES itself, but to batch jobs handled by JES. So do not suppress exit invocation. Ensure that you do not specify TYPE6=NO and TYPE26=NO on the JOBCLASS and STCCLASS statements of the JES2 initialization parameters.
2. You might find it useful during installation to code two SMFPRM*nn* members, one with the exits active and the other with the exits inactive. You can then use the `SET SMF=`*nn* z/OS command to switch your current SMF parameters to the new member. By switching back, using the `SET SMF=`*nn* command, you avoid the need to re-IPL, if you encounter a problem.
3. Exits for SUBSYS STC are required by HCL Workload Automation for Z.

Use the PROG*nn* parmlib member to specify installation exits and control their use. Using PROG*nn*, you can associate multiple exit routines with installation exits at IPL, or while the system is running. Use PROG*nn* in addition to SMFPRM*nn* to specify exits, whether or not you want to take advantage of these functions.

The following example shows how you can specify SMF exits in a PROGxx parmlib member. If you specify this in SMFPRMnn:

```
SYS(...EXITS(IEFU83,IEFACTRT,IEFUJI))
```

you would add this to get the equivalent processing in PROGnn:

```
EXIT ADD EXITNAME(SYS.IEFU83) MODNAME(IEFU83)
EXIT ADD EXITNAME(SYS.IEFACTRT) MODNAME(IEFACTRT)
EXIT ADD EXITNAME(SYS.IEFUJI) MODNAME(IEFUJI)
```

When you associate new exit routines with SMF exits through PROGnn or the SETPROG command, you must use the following naming conventions:

- For exits listed on the EXITS keyword of the SYS statement in SMFPRMnn, each exit will have the name SYS.*xxxx* (where *xxxx* is one of the exits listed).
- For exits listed on the EXITS keyword of the SUBSYS statement of SMFPRMnn, each exit will have the name SYS*zzzzz.xxxx* (where *zzzz* is the name of the subsystem and *xxxx* is one of the exits listed).

If you define two members in SYS1.PARMLIB with two different names, for example, PROG03 in which there is the statement `EXIT ADD EXITNAME(SYS.1 EFACTRT) MODNAME(EQQACTR1)`, you can switch to the version EQQACTR1 without re-ipling by issuing the command:`/SET PROG=03`

If you are using FTP, you must add the following statement to the SMFPRMxx member:

```
 SUBSYS(OMVS,EXITS(IEFUJI,IEFU83))
```

Also, these statements must be added to the PROG*nn* member, making sure that you replace MODNAME with the module name that was used when the exits were link-edited:

```
 EXIT ADD EXITNAME(SYSOMVS.IEFU83) MODNAME(EQQU831)
 EXIT ADD EXITNAME(SYSOMVS.IEFUJI) MODNAME(EQQUJI1)
```

For information on using PROG*nn* to control the use of exits and exit routines, see *z/OS Initialization and Tuning Reference*

Updating z/OS dump options

The sample JCL procedure for an HCL Workload Automation for Z address space includes a DD statement and a dump data set is allocated by the EQQPCS02 JCL created by EQQJOBS. SYSMDUMP is the dump format preferred by the service organization.

Ensure that the dump options for SYSMDUMP include RGN, LSQA, TRT, CSA, and GRSQ on systems where an HCL Workload Automation for Z address space will run. To display the current SYSMDUMP options, issue the z/OS command `DISPLAY DUMP,OPTIONS`. You can use the `CHNGDUMP` command to alter the SYSMDUMP options. Note that this command changes the parameters only until the next IPL is performed.

To dump an HCL Workload Automation for Z address space by using the z/OS `DUMP` command, the SDUMP options must specify RGN, LSQA, TRT, CSA, and GRSQ. Consider defining these options as your system default.

When dumping the controller address space, if JTOPTS CRITJOBS(YES) was specified or taken as the default, you must also dump the data spaces; this is important if you need to collect information when a critical path problem occurs. Add `DSPNAME=('zzzz',*)` to the DUMP command, where *zzzz* is the controller subsystem name.

Updating the z/OS link-library definition

If you installed HCL Workload Automation for Z in a separate load-module library, it is recommended that you define this library in the active LNKLST*nn* member. Alternatively, you can define the load-module library on the STEPLIB DD statement of the started-task JCL and TSO logon procedures of HCL Workload Automation for Z dialog users.

If you installed load modules in the data set OPC.SEQQLMD0 and this data set is cataloged in the master catalog, insert this record before the last entry in the LNKLST*nn* member to add this library to the link library concatenation:

Example

```
Adding LINKLIB
OPC.SEQQLMD0
```

If you choose not to define the HCL Workload Automation for Z load-module library in the LNKLST*nn* member, you *must*:

- Copy the tracker modules, EQQINIT2 and EQQSSCM2, to a library in the z/OS link-library concatenation. EQQINIT2 is used by the master-scheduler-initialization function when the z/OS system is being IPLed. EQQINIT2 then loads EQQSSCM2 into common storage. EQQSSCM2 is about 23KB and is placed above the 16MB line. Remember to copy the modules again whenever they are updated by HCL Workload Automation for Z maintenance. This is especially important for the EQQSSCM2 module, which must be at the same update level as the rest of the HCL Workload Automation for Z code.
- Define the HCL Workload Automation for Z load-module library on a STEPLIB DD statement in the started-task JCL.
- Define the HCL Workload Automation for Z load-module library on a STEPLIB DD statement in the TSO logon procedure of all HCL Workload Automation for Z dialog users.
- Load the dialog module, EQQMINO2, from an APF-authorized library. If you define the HCL Workload Automation for Z load-module library on a TSO STEPLIB DD statement, and any of the other libraries defined on this DD statement

are not authorized, you must copy EQQMINO2 to another library in the LNKLST concatenation so that it is loaded APF authorized. You must also remember to copy the module again whenever it is updated by HCL Workload Automation for Z maintenance.

Updating XCF initialization options

This section is useful if you use XCF for communication.

XCF initialization options are specified in the COUPLE*nn* member of SYS1.PARMLIB. If you have not specified your own COUPLE*nn* member, the system uses the default member, COUPLE00. The IBM-supplied COUPLE00 member causes the system to be IPLed in XCF-LOCAL mode. This mode is *not* supported by HCL Workload Automation for Z. So ensure that your system uses a COUPLE*nn* member that does not IPL the system in XCF-LOCAL mode. The COUPLE*nn* member must include the PCOUPLE keyword of the COUPLE statement. If this is omitted, XCF is initialized in XCF-LOCAL mode. For HCL Workload Automation for Z purposes, you can use the default values for the remaining XCF options.

```
COUPLEnn example
COUPLE   SYSPLEX(PLEX1)               /* SYSPLEX name            */
         PCOUPLE(PLEX1.COUPLE1)       /* Primary couple data set */
         ACOUPLE(PLEX2.COUPLE2)       /* Alternate couple data set*/
         MAXMSG(2000)                 /* No of 1k message buffers */
CLASSDEF CLASS(DEFAULT)               /* Default transport class */
         CLASSLEN(956)                /* Message length          */
         GROUP(OPCGRP,OPCDS)          /* OPC Group names         */
PATHIN   DEVICE(cccc,dddd)
PATHOUT  DEVICE(aaaa,bbbb)
PATHIN   STRNAME(str1,str2) CLASS(DEFAULT)
PATHOUT  STRNAME(str1,str2) CLASS(DEFAULT)
```

Issue the console command "D XCF,CLASSDEF,CLASS=ALL" to see if you already have a DEFAULT class (the name of this class might be something other than DEFAULT) having CLASSLEN(956), which is the default value. If there is such a class, you just need to add the TWS specific GROUP names (OPCGRP,OPCDS) to the CLASSDEF statement for that CLASS, as shown in the example above.

Note: By specifying MAXMSG(2000) on the COUPLE statement, as shown above, all transport classes will use this value unless a different value is specified at the CLASSDEF level. MAXMSG(2000) is the default value.

If XCF is used to connect the Data Store to the controller, a specific XCF group must be defined, and it must be different from the one used to connect the controller to the z/OS® tracker. These two separate XCF groups can use the same XCF transport class.

Note: You can change XCF options while the system is active by using the SETXCF operator command.

For more information about XCF, see *z/OS® MVS™ Setting up a Sysplex*.

Modifying TSO parameters

You must define the EQQMINO2 module to TSO on each system where you install the scheduler dialogs. You must also authorize the HCL Workload Automation for Z TSO commands on every system where you install HCL Workload Automation for Z. If you do not authorize the TSO commands, the commands will only work on the system where the Z controller is installed.

To request services from the subsystem for a TSO user, the HCL Workload Automation for Z dialog invokes the EQQMINO2 module using the TSO service facility. EQQMINO2 is the dialog interface module. It must run as an APF-authorized program. To achieve this, define EQQMINO2 to TSO. If you are installing the scheduler dialogs, include EQQMINO2 in the list of programs defined by the AUTHTSF statement in the IKJTSO*nn* member of SYS1.PARMLIB. This statement defines programs to be authorized when invoked using the TSO service facility, as shown in the following example:

Example

```
IKJTSOnn AUTHTSF example
AUTHTSF NAMES(IKJEFF76 +
              IEBCOPY  +
              EQQMINO2)
```

If you prefer, you can put EQQMINO2 in CSECT IKJEFTAP instead of IKJTSO*nn*. For more information about using IKJEFTAP, see *TSO/E Customization*.

HCL Workload Automation for Z supports the BACKUP, BULKDISC, JSUACT, OPINFO, OPSTAT, SRSTAT, and WSSTAT TSO commands. Update the IKJTSO*nn* member on each system where you are installing HCL Workload Automation for Z to define these commands as authorized commands. To do this, add them to the list of commands defined by the NAMES keyword of the AUTHCMD statement, as shown in the following example

Example

```
IKJTSOnn AUTHCMD example
AUTHCMD NAMES(BACKUP +
              BULKDISC +
              JSUACT +
              OPINFO +
              OPSTAT +
              SRSTAT +
              WSSTAT)
```

If the default entry in the ISPF TSO command table ISPTCM is set for unauthorized TSO commands, then ISPTCM must be updated. The ISPTCM can be updated using the ISPMTCM macro. Define the BACKUP, BULKDISC, JSUACT, OPINFO, OPSTAT, SRSTAT, and WSSTAT commands like this:

Example

```
ISPTCM example
ISPMTCM  FLAG=62,ENTNAME=BACKUP
ISPMTCM  FLAG=62,ENTNAME=BULKDISC
ISPMTCM  FLAG=62,ENTNAME=JSUACT
ISPMTCM  FLAG=62,ENTNAME=OPINFO
ISPMTCM  FLAG=62,ENTNAME=OPSTAT
ISPMTCM  FLAG=62,ENTNAME=SRSTAT
ISPMTCM  FLAG=62,ENTNAME=WSSTAT
```

No update is needed to ISPTCM if the default entry is set up for authorized TSO commands. For more information about the ISPMTCM macro statements, see *ISPF Planning and Customization*.

Performance considerations

The tracker and the controller address spaces must be nonswappable. To do this, include the definition of their top load module, EQQMAJOR, in the program properties table (PPT). This PPT entry example is defined in a SCHED*nn* member of SYS1.PARMLIB:

```
SCHEDnn example
PPT PGMNAME(EQQMAJOR) NOSWAP
```

The EQQMAJOR program must run in storage key 8, the default value.

To ensure prompt processing by HCL Workload Automation for Z and to avoid delays in the handling of event records, the tracker subsystem performance rating (that is, its dispatching priority) should match that of the JES subsystem.

Defining the DLF exit for Hiperbatch™ support

If you want to include Hiperbatch™ support for HCL Workload Automation for Z controlled jobs, specify the DLF exit name in the COFDLF*nn* member of SYS1.PARMLIB. A DLF exit sample is supplied with the SEQQSAMP library. The exit must reside in an authorized library in the LNKLST concatenation. This example of a COFDLF*nn* member defines a DLF exit called OPCDLF:

```
COFDLFnn example
CLASS MAXEXPB(nnnn) PCTRETB(nnn) CONEXIT(OPCDLF)
```

For more information about invoking Hiperbatch™ support in HCL Workload Automation for Z, see *Customization and Tuning*.

Starting the product automatically

The COMMND*nn* member of SYS1.PARMLIB list z/OS commands automatically issued during system initialization. To avoid delays in starting HCL Workload Automation for Z when the z/OS system is started, consider including the names of your HCL Workload Automation for Z started tasks in this member. For information on how to include start commands for your HCL Workload Automation for Z address spaces, see *MVS™ Initialization and Tuning Reference*.

Updating APPC options

If you want to use the API, or the server, to communicate with HCL Workload Automation for Z, you must update APPC options. For a detailed description of what you need to do, see Step 16. Activating support for the API on page 159, or Step 17. Activating support for the product dialog and programming interface using the server on page 162.

Implementing support for data set triggering

Use the HCL Workload Automation for Z data set triggering function to start dependent processing or schedule unplannable work by automatically generating special resource availability events when a data set is closed after being opened for:

- Read processing
- Output processing
- Either read or output processing.

HCL Workload Automation for Z uses the SMF exit IEFU83 to generate a resource availability event when IEFU83 is called for SMF record types 14, 15, or 64. The data set activity SMF records are generated when a data set is closed or processed by EOV. HCL Workload Automation for Z will generate resource availability events only when the data set is closed. When a VSAM data set is closed, two SMF 64 records are created, one each for the DATA and INDEX components. When resource availability events are requested for VSAM data sets, the event will be created when the DATA component is closed, HCL Workload Automation for Z will not generate an event when the INDEX component is closed.

SMF data set activity records are written when the data set is closed, regardless of whether the JOB/STEP/TASK/USER completed successfully. For more information about the data sets that generate SMF record types 14, 15, or 64, see the documentation for SMF.

To define the data sets for which you want events to be generated, you can perform either of the following:

- Use the EQQRXTRG program to centralize and automate the population of the data set to which the EQQJCLLIB DD name refers. For detailed information about running event-driven workload automation, see *Managing the Workload*.
- Build a selection table, as described in Invoking the EQQLSENT macro on page 361. The selection table is located in ECSA. It is automatically loaded from the data set referred to by the EQQJCLIB DD name when the event writer is started in a tracker if a table has not previously been loaded since IPL. To reload the table at any time, issue the z/OS modify command:

```
F procname,NEWDSLST
```

Note: No support is available for the data set triggering function before the event writer is started immediately after a z/OS IPL. When the event writer has started after IPL, data set triggering functions are available if the event writer is subsequently stopped. To stop data set triggering at any time issue the `NEWDSLST` modify command to load a table that contains only the end-of-table indicator.

To implement support for the data set triggering function, perform these actions:

- Update SYS1.PARMLIB member SMFPRMnn as described in Updating SMF parameters on page 94.
- Install SMF exit IEFU83 using the EQQU831 sample. See Macro invocation syntax for EQQEXIT on page 358 on how to specify the SRREAD parameter.
- Define the data set selection criteria as described by the event-driven resource handling section in *Managing the Workload*.

  The procedure described in Invoking the EQQLSENT macro on page 361 is supported for compatibility with earlier versions only.

Step 7. Setting up the RACF® environment

If your installation protects data and resources from unauthorized use, you must define HCL Workload Automation for Z to your security system. This section assumes that the Resource Access Control Facility (RACF®) is installed and active on your z/OS system. It describes the activities you must perform to define and enable the security environment for HCL Workload Automation for Z.

For detailed plans and instructions about how to establish a security strategy for your HCL Workload Automation for Z resources, see *Customization and Tuning*.

Controlling the user ID of the address space

If you run HCL Workload Automation for Z as a started task, you must associate the cataloged procedure name with a suitably authorized RACF® user. The user ID must be defined in the STARTED resource class.

If you use any of the following definitions in your initialization statements, you must also define an OMVS segment for the controller user ID:

- TCPIP parameter in the ROUTOPTS statement.
- MONOPTS statement.
- HTTP or HTTPS parameter in the ROUTOPTS statement.

For any user IDs running the controller, server, output collector, or TCP connected tracker started task, ensure that you defined a HOME directory in the OMVS segment. For details, see HOME directory requirements for started tasks on page 197.

Controlling the user ID of submitted jobs

HCL Workload Automation for Z can submit three kinds of jobs:

- Normal production jobs on page 102, which are submitted when their prerequisites in the current plan are fulfilled.
- Stand-alone cleanup jobs on page 103, which are submitted to run cleanup actions separately from the original job.
- Dialog jobs on page 103, which you can submit directly from a panel in the HCL Workload Automation for Z dialog.

Normal production jobs

HCL Workload Automation for Z submits production jobs to the internal reader, or starts started tasks, when all prerequisites are met. HCL Workload Automation for Z first attempts to find jobs in the JS file. If no job is found, it keeps searching in the following order:

- Partitioned data set indicated by the JOBLIBRARY operation user field, if specified. If no job is found, an error message is issued.
- Job-library-read exit (EQQUX002), if loaded. If no job is found, it searches the EQQJBLIB partitioned data set concatenation.
- EQQJBLIB partitioned data set concatenation, if the JOBLIBRARY user field and EQQUX002 are not specified.

You can determine the authority given to a job or started task by the following ways:

- Submitting work with the authority of the HCL Workload Automation for Z address space. The job or started task is given the same authority as the Z controller or tracker whose submit subtask actually submits the work. For example, work that is transmitted from the Z controller and then submitted by the tracker is given the authority of the tracker.
- Using the user field name SUBJOBUSER, specified at operation level, to cause any kind of jobs to be submitted with a specified user ID. If you use this method, it is preferred over using the job-submit exit EQQUX001.

  The user field name SUBJOBUSER must always be uppercase. The user field value cannot be longer than 8 characters and cannot contain blanks: if you specify a value longer than 8 characters it is truncated; if you specify a value containing a blank, the characters after the blank are not considered.
- Using the job-submit exit, EQQUX001. This exit is called when HCL Workload Automation for Z is about to submit work.
  - You can use the RUSER parameter of the EQQUX001 exit to cause the job or started task to be submitted with a specified user ID. The RUSER name is supported even if the job or started task is first sent to a tracker before being started.
  - In certain circumstances you might need to include a password in the JCL to propagate the authority of a particular user. You can use the job-submit exit (EQQUX001) to modify the JCL and include a password. The JCL is saved in the JCL repository (JS*n*) data set *before* the exit is called, thus avoiding the need to store JCL with specific passwords. This method prevents the password from being visible externally. For more information about the job-submit exit, see *Customization and Tuning*.

Stand-alone cleanup jobs

Their purpose is to run data set cleanup actions and can be submitted when:

- An automatic internal process takes place (for example, when cleanup type immediate is used and an operation ends in error)
- A Start Cleanup command is issued by an HCL Workload Automation for Z dialog or the Dynamic Workload Console.

Activate exit EQQUX001 to make sure that the submitter of the stand-alone cleanup job is the same as the submitter of the original job, otherwise the stand-alone cleanup job will run with the same authority as the controller or the tracker that submits it. The current EQQUX001 sample contains a procedure to set the RUSER value according to the value of the USER= keyword in the jobcard of the original job.

Dialog jobs

When you submit HCL Workload Automation for Z batch jobs from your TSO address space, they go through normal TSO functions. This means that you can submit any job allowed by TSO/E. HCL Workload Automation for Z makes no authority checks when the job is submitted.

For the HCL Workload Automation for Z batch job to run successfully, it must be authorized to reference the data sets it uses. The submitting TSO user might also need authorization to use a specific function. For example, a user could have update authority to the AD file but not have the authority to use the AD mass update function.

Controlling dynamic allocation of job libraries

You control the dynamic allocation of job libraries by setting the user field JOBLIBRARY at operation level. If you set JOBLIBRARY, its value is preferred over the job-library-read exit (EQQUX002) and EQQJBLIB partitioned data set concatenation.

Note: Carefully examine how often you choose to allocate job libraries dynamically, because this might heavily impact the scheduling process.

For more detailed information, see the section about defining jobs in *Managing the Workload*.

Protecting data sets

For basic security of HCL Workload Automation for Z data, you should restrict access to all the product data sets.

Two categories of users need different levels of access to the product data sets:

- Software support people must be able to debug problems and reorganize VSAM files. You might give them alter access to all the product data sets.
- Administrators and operators must be able to use the product dialogs. They need read access to ISPF-related data sets (such as the panel and message libraries), but they do not access the databases (such as the workstation database) directly: these files are accessed by the HCL Workload Automation for Z subsystem, not by any code in the TSO user's address space. Authority to access the data for a dialog user is given using the authorization functions provided by the product.

The HCL Workload Automation for Z started task needs:

- Alter access to VSAM data sets
- Read access to input data sets, such as the message library (EQQMLIB) and parameter library (EQQPARM)
- Update access to all other HCL Workload Automation for Z data sets
- Update access to catalogs and alter access to data sets for all work that HCL Workload Automation for Z tracks, if you use the Restart and Cleanup function.

Controlling access to resources

Before HCL Workload Automation for Z performs any request initiated by a user, a security verification check is passed to the system authorization facility (SAF) to ensure that the user is authorized to access all resources needed to run the request. A user can request HCL Workload Automation for Z services from:

- An ISPF dialog session
- TSO commands
- The program interface (PIF)
- The application programming interface (API)
- Dynamic Workload Console

Any security software that interfaces with SAF also works with HCL Workload Automation for Z. For this section, the security product is assumed to be RACF®.

The z/OS router service calls RACF® to perform authority checks. It provides an installation exit that you can use instead of, or in addition to, RACF® to perform resource control functions.

Use the HCL Workload Automation for Z reserved resource class IBMOPC.

The default class for HCL Workload Automation for Z is OPCCLASS. If you use a different class name, you must specify it in the AUTHDEF statement. Generally, this means specifying CLASS(IBMOPC) in the AUTHDEF statement. If you are running more than one HCL Workload Automation for Z system, for example a test system and production system, you might want to define more than one RACF® class. By using different CLASS parameters in each AUTHDEF statement, you can specify a different authorization scheme for each system.

To control access to HCL Workload Automation for Z functions, give at least one TSO user-class authority to the resource class. This TSO user can then allow other HCL Workload Automation for Z users to access resources as needed.

HCL Workload Automation for Z also uses the APPL resource class. Define the subsystem name as a resource in the APPL class.The easiest way to do this is to have the RACF® administrator give class authority to the APPL resource class to one TSO user. This TSO user defines the subsystem name (for example, OPCC) to the APPL resource class by entering:

```
/*Define subsystem resource*/
RDEFINE APPL OPCC  UACC(NONE)
```

See *RACF® Command Reference* and *RACF® Administrator's Guide* if you are unfamiliar with this process.

When the subsystem name is defined to RACF®, you can give other TSO users access to HCL Workload Automation for Z. For example, to allow the TSO user OPCUGRP to access OPCC with an update access authority by default, enter:

Example

```
/*Permit access to HCL Workload Automation for Z*/
PERMIT OPCC ID(OPCUGRP) ACCESS(UPDATE) CLASS(APPL)
```

For remote dialog users and remotely run PIF applications, the server will do the authority checking; it will check both the APPL class subsystem name resource and the scheduler fixed resources. The user for which the server does authority checking is:

- For dialog users, the TSO user ID.
- For PIF applications, the user ID defined in the security environment of the PIF job.

Permitting access to the controller through the API

If you use the API, you can control access to the Z controller through the security functions of both APPC and HCL Workload Automation for Z. Ensure that you consider both these environments when you update RACF®. For more information about controlling access to HCL Workload Automation for Z through the API, see *Customization and Tuning*.

Controlling access to HCL Workload Automation for Z resources when using the Dynamic Workload Console

The WebSphere Application Server Liberty Base performs a security check when a user tries to use Dynamic Workload Console, checking the user ID and password. The WebSphere® Application Server associates each user ID and password to an administrator.

The scheduler resources are currently protected by RACF®.

The Dynamic Workload Console user should only have to enter a single user ID and password combination, and not provide two levels of security checking (at the WebSphere® Application Server level and then again at the HCL Workload Automation for Z level).

The security model is based on having the WebSphere® Application Server security handle the initial user verification, while at the same time obtaining 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 mapping the administrator to a RACF® user ID. When a WebSphere® Application Server user tries to initiate an action on z/OS®, the administrator ID is used as a key to obtain the corresponding RACF® user ID.

The server uses the RACF® user ID to build the RACF® environment to access HCL Workload Automation for Z services, so the administrator must relate, or map, to a corresponding RACF® user ID.

For information about how to get the RACF® user ID, see *HCL Workload Automation for Z: Customization and Tuning*.

Permitting access to the controller through the Dynamic Workload Console

If you use the Dynamic Workload Console, you can control access to the Z controller through the security functions of both WebSphere Application Server Liberty Base and HCL Workload Automation for Z. Ensure that you consider both these environments when you update RACF. For more information about controlling access to HCL Workload Automation for Z through the Dynamic Workload Console, see *Customization and Tuning*.

Authorizing HCL Workload Automation for Z as a job submitter

Consider the following resource classes when implementing security for HCL Workload Automation for Z. The examples assume that the RACF® user for the HCL Workload Automation for Z address space is OPCAPPL, which is the name specified in the started-procedure table.

JESJOBS

If your installation has activated the JESJOBS class, you must permit HCL Workload Automation for Z to submit all jobs that are defined in the current plan. One way of doing this is to permit HCL Workload Automation for Z to submit all jobs. You can do this by:

1. Defining the submit resource:

```
RDEFINE JESJOBS SUBMIT.*.*.* UACC(NONE) OWNER(OPCAPPL)
```

2. Authorizing HCL Workload Automation for Z:

   ```
   PERMIT SUBMIT.*.*.* CLASS(JESJOBS) ID(OPCAPPL) ACC(READ)
   ```

SURROGAT

A *surrogate job submission* occurs when all the following conditions are met:

1. USER=*xxxx* is specified on the job card of the submitted job.
2. The *xxxx* is not the same as the submitting (RACF®) user.
3. No password is specified on the job card.

You might use the job-submit exit (EQQUX001) to return a submitting user in the RUSER field. This is required if you want stand-alone cleanup jobs to be submitted with the same authority as the original job, otherwise you can replace it with surrogate job submission.

To permit HCL Workload Automation for Z to submit this job, perform the following steps:

1. Activate the surrogate class:

   ```
   SETROPTS CLASSACT(SURROGAT)
   ```

2. Define the submit resource:

   ```
   RDEFINE SURROGAT APLUSER.SUBMIT UACC(NONE) OWNER(APLUSER)
   ```

3. Authorize HCL Workload Automation for Z:

   ```
   PERMIT APLUSER.SUBMIT CLASS(SURROGAT) ID(OPCAPPL) ACC(READ)
   ```

If the `PRIVILEGED` or `TRUSTED` attribute is set in the Started Procedure Table (SPT) entry, the HCL Workload Automation for Z is authorized to submit jobs under any user regardless of what is defined in the resource rules.

For further information, see the *RACF® Administrator's Guide*.

Authorizing HCL Workload Automation for Z to issue JES commands

Consider the following resource classes when implementing security for HCL Workload Automation for Z. The examples assume that the RACF user for the HCL Workload Automation for Z address space is OPCAPPL, which is the name specified in the started-procedure table.

OPERCMDS

If the OPERCMDS class is active and you have specified HOLDJOB(YES) or HOLDJOB(USER) for an event writer, the HCL Workload Automation for Z address space where the event writer is started must be authorized to issue the JES release command. One method is to permit HCL Workload Automation for Z to issue all JES commands.

To permit HCL Workload Automation for Z to issue JES commands on a JES2 system, perform the following steps:

1. Define the resource:

   ```
   RDEFINE OPERCMDS JES2.* UACC(NONE)
   ```

2. Authorize HCL Workload Automation for Z:

   ```
   PERMIT JES2.* CLASS(OPERCMDS) ID(OPCAPPL) ACC(UPDATE)
   ```

On a JES3 system, replace JES2.* with JES3.* in the example. Alternatively, you could specify the JES%.* resource name for either a JES2 or JES3 system.

If you use HCL Workload Automation for Z to schedule started tasks, the address space must be authorized to issue the z/OS start command. One way of doing this is to permit HCL Workload Automation for Z to issue all z/OS commands. To do this, perform the following steps:

1. Define the resource:

   ```
   RDEFINE OPERCMDS ZOS.* UACC(NONE)
   ```

2. Authorize HCL Workload Automation for Z:

   ```
   PERMIT ZOS.* CLASS(OPERCMDS) ID(OPCAPPL) ACC(UPDATE)
   ```

Authority to use the z/OS start command is also required if you use Hiperbatch™ support for HCL Workload Automation for Z operations.

JESSPOOL

If the JESSPOOL class is active and you use the HCL Workload Automation for Z JCC function, you must authorize HCL Workload Automation for Z to access SYSOUT data sets for all jobs in the current plan. Also, the output collector and data store require the ALTER access for the JESSPOOL class. One way of doing this is to permit HCL Workload Automation for Z to access all SYSOUT data sets. To permit HCL Workload Automation for Z, output collector, and data store to access all SYSOUT data sets.

To access all SYSOUT data sets, perform the following steps on each system where the JCC, output collector, and data store are started:

1. Define the resource:

   ```
   RDEFINE JESSPOOL *.* UACC(NONE)
   ```

2. Authorize HCL Workload Automation for Z:

   ```
   PERMIT *.* CLASS(JESSPOOL) ID(OPCAPPL) ACC(ALTER)
   ```

If the `PRIVILEGED` or `TRUSTED` attribute is set in the Started Procedure Table (SPT) entry for HCL Workload Automation for Z, the address space is authorized to issue any commands and to process spool data sets regardless of what is defined in the resource rules.

For further information, see the *RACF® Security Administrator's Quick Reference.*

Authorizing HCL Workload Automation for Z end-to-end and Dynamic Workload Console server tasks for security resource EZB.BINDDVIPARANGE

If you use the Dynamic Workload Console, you must give UPDATE authorization for the EZB.BINDDVIPARANGE to the user ID of the Dynamic Workload Console server when using DVIPA host names. If you specify the TCPOPTS statement for the server, the HOSTNAME parameter overrides the JSCHOSTNAME parameter of the SERVOPTS statement, if any.

Step 8. Securing communications

HCL Workload Automation for Z supports authentication and cryptography by activating the Secure Sockets Layer (SSL) transport protocol for transmitting and accepting secure information.

You can configure HCL Workload Automation for Z to enable SSL communication in a TCP/IP network or, you can implement SSL security for HTTP connections as required.

Security for TCP/IP connections

The scheduler authentication mechanism uses the SSL services of z/OS®. For further details, see *z/OS® Cryptographic Services System Secure Sockets Layer Programming*.

To enable SSL authentication for your network, perform the following actions:

1. Create the SSL work directory by using the EQQPCS10 sample JCL. You can use the same directory as the one used for SSL in end-to-end. In the following examples, the directory is `/u/tws/ssl`
2. From `/u/tws/ssl/` as current directory, open a shell prompt, start the **gskkyman** utility of z/OS® Cryptographic Services System SSL, and do the following:
    a. Create the keystore database and consider protecting it from unauthorized access, because it has to contain private key and trusted certificates. For example, consider the database `/u/tws/ssl/tws1.kdb`.
    b. Generate a password file and store it in the SSL directory defined in the previous step, for example `/u/tws/ssl/tws1.sth`.
    c. At this point you can:
        - Create a certificate request and send it to the Certificate Authority.
        - Store the signed certificate in the database.
        - Import the certificate of the Certification Authority which signed your certificate.

    In this way you have a database containing both your certificate and Certification Authority's one.

    The scheduler uses a default name to identify your certificate; therefore you are not required to set up a multiple database handling. If you need different certificates in order to partition your network from a security point of view, you need different databases. The advantage of this solution is that you can store each database in a different directory, with its own access list.

    Alternatively, you can create your own custom self-signed certificates. For details, see the procedure described in the section about configuring TLS with custom self-signed certificates in *Customization and Tuning*.

3. Configure HCL Workload Automation for Z, by specifying the TCPOPTS statement for each component of your network. Consider each component according a client-server model. Typically, a client-server group is composed by the trackers and data stores communicating with the corresponding controller, or by a remote interface communicating with the corresponding server.

   When the controller or the server started task communicates with a partner component, the communication is always started by the partner component; therefore the partner acts as a client. Differently from the end-to-end communication, the communicating partners use the same port numbers for both non-SSL and SSL communications. Specify the same TCPOPTS parameters for all the components in a client-server group.

   For a detailed description of the TCPOPTS statement, see *Customization and Tuning*. The following example shows a TCPOPTS definition to activate an SSL support that uses a SAF key ring.

   ```
   TCPOPTS
     . . .
     SSLEVEL(FORCE)                      1
     SSLKEYSTORE(MYKEYRING)              2
     SSLKEYSTORETYPE(SAF)                3
     SSLAUTHSTRING('OPCMASTER')          4
     SSLAUTHMODE(STRING)                 5
   ```

   In this example:

   1

   The FORCE keyword enables the SSL communication.

   2

   MYKEYRING is the SAF key ring used to connect the security certificates.

   3

   SAF is the type of key ring used.

   4

   OPCMASTER is the string defined as Common Name (CN) in the certificate.

   5

   The STRING keyword enables the check on the CN string.

When designing your configuration from a security point of view, consider that:

- To enforce your security, you can use the SSLAUTHMODE(STRING), that requires to:
  - Create an SSL certificate for each controller started task. This certificate will be used by the controller and its remote partners. Define the certificate using as Common Name a unique string corresponding to the controller.
  - Create an SSL certificate for each server started task. This certificate will be used by the server and its remote partners. Define the certificate using as Common Name a unique string corresponding to the server.

The SSLAUTHSTRING must match the information contained in the certificate sent by the partner. To verify it, you can use the **gskkyman** utility that allows displaying the keys database and SSL certificate content. The certificate CN is returned by **gskkyman** as the first line of the "Subject".

- If you prefer to use SSLAUTHMODE(CAONLY), then you can use a single SSL certificate for all your network.

Security for HTTP connections

You can provide security for an HTTP connection between the following components:

- The Z controller and the HCL Workload Automation Agent (z-centric agent).
- The Z controller and another Z controller (z/OS remote engine).
- The Z controller and the dynamic domain manager.
- The Z controller and the HCL Workload Automation master domain manager (distributed remote engine).

SSL-secure connections are implemented using specific settings in the statement, and the HTTPS keyword in the statement. For more information about these statements, see *Customization and Tuning*.

If you use the secure connection with the SSL protocol, you must import the security certificates into your security system.

Note: If you imported the default security certificates during the installation of the previous version of the product, you must remove them and run the EQQRCERT job to import the new certificates. If you already imported the new default security certificates during the installation of the HCL Workload Automation agent for z/OS, then you must not perform this procedure again. Complete the procedure for creating a secure connection by configuring the SSLKEYRING keyword with the value used for installation of the HCL Workload Automation agent for z/OS.

At installation time, the default security certificates are automatically stored into the SEQQDATA library:

EQQCERCL

The security certificate for the client.

EQQCERSR

The security certificate for the sever.

You can decide to use these default certificates or create your own. In both cases, you must import them into your security system. If you are using RACF®, you are provided with the sample job EQQRCERT to import the certificates. To run this job, ensure that you use the same user ID that RACF® associates with the controller started task.

If you create your own certificates for an HTTP connection with the master domain manager or with the dynamic domain manager, you must run the customizing steps described in the section about customizing SSL connection to the master domain manager and dynamic domain manager in *HCL Workload Automation: Administration Guide*. A procedure about creating your own custom self-signed certificates is described in the section about configuring TLS with custom self-signed certificates in *Customization and Tuning*.

If you are using SSL to communicate with a master domain manager, backup master domain manager, or dynamic domain manager, then the prefix of the common name of the controller certificate must be defined in the **Broker.AuthorizedCNs** option in the `BrokerWorkstation.properties` file located in the `*TWA_home*/TDWB/config` directory of the distributed engine.

The EQQRCERT job performs the following actions:

- Copies the EQQCERCL certificate to a temporary sequential data set
- Copies the EQQCERSR certificate to a temporary sequential data set
- Imports EQQCERCL to RACF®
- Imports EQQCERSR to RACF®
- Deletes the temporary sequential data sets
- Creates the SAF key ring that is used to connect the imported certificates
- Updates the RACF® database with the new certificates and key ring

Step 9. Allocating data sets

Note: A standby Z controller uses the same data sets as the Z controller.

At this stage of the installation of your HCL Workload Automation for Z system, you allocate the data sets that your JCL procedures refer to. You can create the data sets by using the jobs created by the EQQJOBS installation aid.

If you are using the EQQJOBS installation aid, you will already have generated several members in the output library that you specified.

Consider carefully where HCL Workload Automation for Z data sets are allocated in your production environment. Some data sets can be highly active. Avoid placing these data sets on DASD volumes with high activity because this can result in poor performance due to contention. Also consider the ability to recover data sets if a DASD volume becomes unusable. If you place all your data sets on the same volume, you must recover many data sets before you can continue your HCL Workload Automation for Z service. *HCL Workload Automation for Z: Customization and Tuning* describes recovery of HCL Workload Automation for Z data sets in detail.

The space to allocate for your data sets depends upon the workload at your installation. It is difficult to give precise figures for the amount of space you will need. The space allocated by the sample JCL should give you enough space to at least get started. These amounts will be enough for the HCL Workload Automation for Z service for many installations. For details about allocating space for VSAM data sets, see Table 23: Calculations of VSAM data set size on page 116 .

The following sections describe the HCL Workload Automation for Z data sets and include examples of the JCL needed to create them.

Allocating the VSAM data sets

Perform this task if you are installing a Z controller.

Table 22: HCL Workload Automation for Z VSAM data sets on page 113 shows the VSAM data sets and their characteristics. The JCL procedure for the Z controller uses all of these data sets except for EQQLDDS and EQQLTBKP, which are used only in the planning batch jobs. Allocate all these VSAM data sets for a Z controller.

Table 22. HCL Workload Automation for Z VSAM data sets

Sample	DD name	Record type	Attributes	Share option	Keys	Record size	Data set
EQQPCS09	N/A	KSDS	REUSE NSPND	3	19 0	200 32000	Archive of current plan
EQQPCS01	EQQADDS	KSDS	UNIQUE SPANNED	3	25 0	1000 131072*	Application description
EQQPCS01	EQQCP1DS	KSDS	REUSE NSPND	3	19 0	200 32000	Current plan 1
EQQPCS01	EQQCP2DS	KSDS	REUSE NSPND	3	19 0	200 32000	Current plan 2
EQQPCS01	EQQCXDS	KSDS	REUSE NSPND	3	64 0	500 32000	Current plan extension
EQQPCS01	EQQXD1DS	KSDS	REUSE NSPND	3	68 0	500 32000	Extended data 1
EQQPCS01	EQQXD2DS	KSDS	REUSE NSPND	3	68 0	500 32000	Extended data 2
EQQPCS01	EQQNXDDS	KSDS	REUSE NSPND	3	68 0	500 32000	New extended data
EQQPCS01	EQQJS1DS	KSDS	REUSE SPANNED	3	28 0	804 180004	JCL repository 1
EQQPCS01	EQQJS2DS	KSDS	REUSE SPANNED	3	28 0	804 180004	JCL repository 2
EQQPCS01	EQQLDDS	KSDS	REUSE SPANNED	2	28 0	440 131072	Long-term-plan work
EQQPCS01	EQQLTBKP	KSDS	REUSE SPANNED	3	28 0	200 131072	Long-term-plan backup
EQQPCS01	EQQLTDS	KSDS	REUSE SPANNED	3	28 0	200 131072	Long-term plan
EQQPCS01	EQQNCPDS	KSDS	REUSE NSPND	3	19 0	200 32000	New current plan

Table 22. HCL Workload Automation for Z VSAM data sets (continued)

Sample	DD name	Record type	Attributes	Share option	Keys	Record size	Data set
EQQPCS01	EQQNCXDS	KSDS	REUSE NSPND	3	64 0	500 32000	New current plan extension
EQQPCS01	EQQNSTDS	KSDS	UNIQUE SPANNED	3	68 0	500 32000	New step awareness
EQQPCS01	EQQOIDS	KSDS	UNIQUE NSPND	3	28 0	800 32000	Operator instruction
EQQPCS07	EQQPKIxx	KSDS	UNIQUE INDEXED	1,3	34 0	77 77	Primary Index
EQQPCS01	EQQRDDS	KSDS	UNIQUE NSPND	3	64 0	400 32000	Special resource descriptions
EQQPCS01	EQQSCPDS	KSDS	REUSE NSPND	3	19 0	200 32000	Current plan backup copy for IBM® Tivoli® Monitoring integration
EQQPCS07	EQQSDFxx	LINEAR	N/A	2,3	N/A	N/A	Data files
EQQPCS01	EQQSIDS	KSDS	UNIQUE NSPND	3	64 0	110 220	Side information file: ETT and configuration information
EQQPCS07	EQQSKIxx	KSDS	UNIQUE INDEXED	1,3	38 0	76 32000	Secondary Index
EQQPCS01	EQQSTDS	KSDS	UNIQUE SPANNED	3	68 0	500 32000	Step awareness
EQQPCS01	EQQWSDS	KSDS	UNIQUE NSPND	3	10 0	100 32000	Workstation, calendar, and period descriptions.
EQQPCS01	EQQSCPDS	KSDS	REUSE NSPND	3	19 0	200 32000	Current plan backup copy for IBM® Tivoli® Monitoring integration

Note:

Table 22. HCL Workload Automation for Z VSAM data sets (continued)

Sample	DD name	Record type	Attributes	Share option	Keys	Record size	Data set
1. * The maximum record size for EQQPCS01 is the default maximum value. This can be increased as in the example that follows. 2. In specific situations where the size of the CP files (CP1, CP2, NCP, SCP) are large and the batch daily planning jobs cause a considerable number of updates to the NCP, it is possible for the NCP to become very large. This might require the allocation of additional extents (not additional volumes, because ADDVOL support is not available for the NCP file). Consider freespace allocation for the current plan, including NCP (EQQCP1DS, EQQCP2DS, EQQCXDS, EQQNCPDS and EQQSCPDS), application descriptions (EQQADDS), resource descriptions (EQQRDDS), and operator instructions (EQQOIDS) data sets. 3. The extended data sets XD1DS, XD2DS, NXDDS, OXDDS have a logical correspondence and use like the current plan data sets CP1DS. CP2DS, NCPDS, ONCPDS. As the old CP (OCP) can be either the CP1 or the CP2 according to which one is inactive and not current, with the same logic OXD can be either XD1 or XD2 according to which one is not currently active. 4. If you are upgrading from a previous HCL Workload Automation for Z release, it is recommended to use the new samples shipped and set with EQQJOBS. The allocation samples like EQQPCS01 use variables. If you are customizing previous allocation JCLs, make sure you correctly position the changes and use the correct set of defined variables. 5. The IDCAMS ALTER ADDVOLUMES command is not supported for HCL Workload Automation for Z data sets because it requires that the data set be closed and reopened before the VSAM is updated with the new volumes added.							

You can allocate the VSAM data sets by submitting the sample listed in Table 22: HCL Workload Automation for Z VSAM data sets on page 113. Alternatively, you can allocate one or more of the VSAM data sets by running a job like this:

```
Allocating a VSAM data set
//ALOCVSAM JOB STATEMENT PARAMETERS
//*----------------------------------*
//* ALLOCATE AN OPC VSAM DATA SET *
//*----------------------------------*
//ALLOC   EXEC PGM=IDCAMS,REGION=512K
//SYSPRINT DD  SYSOUT=Q
//EQQVOL1  DD  DISP=OLD,VOL=SER=volser,UNIT=3390
//SYSIN    DD  *
  DEFINE +
    CLUSTER ( +
      NAME('OPC.INST.AD') UNIQUE +
      SPANNED +
      SHR(3) VOL(volser) CYLINDERS(2 2) +
            ) +
    DATA    ( +
      NAME('OPC.INST.ADDATA') +
      KEYS(25 0) RECORDSIZE(1000 132072) +
            ) +
    INDEX   ( +
         NAME('OPC.INST.ADINDEX') +
                  )
/*
```

This example allocates the application description database.

You can allocate VSAM data sets on different device types.

Allocate enough space for your data sets, depending upon the amount of work HCL Workload Automation for Z processes at your installation. For details about allocating space for VSAM data sets Table 23: Calculations of VSAM data set size on page 116.

Table 23. Calculations of VSAM data set size

Data set	**Size in bytes is total of:**	
	Number of	**Multiplied by**
Application description (EQQADDS)	Application and group definitions	240
	Run cycles	160
	Positive run days	3
	Negative run days	3
	Rules definition segment (based on the combination specified)	Average is 96
	Application dependencies	100
	Operations	140
	Internal dependencies	16
	External dependencies	84
	Time interval to resolve ext. dep. (based on matching criteria)	40
	Special resources	64
	Operation Extended Info	200
	Operation System Automation Info	336
	Conditional dependency defined on the operation	32
	Conditional dependency details	56
	Time interval to resolve cond. dep. (based on match. criteria)	36
	Operation User fields	74
	Operation Remote Engine Info	200
	Operation variable duration/deadline (based on run cycle)	44
	Operation not started alert/action settings	20
	Additional AD info (`adrinfo`)	360
	Tags	<160
	Variable tables	98
	Variables	476
	Variable dependencies	88

Table 23. Calculations of VSAM data set size (continued)

Data set	Size in bytes is total of:	
	Number of	**Multiplied by**
Current plan (EQQCPnDS)	Header record (one only)	188
	Workstations	212
	Workstation open intervals	48
	Workstation access method data	72
	Occurrences	302
	Operations	356
	Dependencies	14
	Special resource references	64
	Operation Extended Information	200
	Jobs	116
	Executed steps	20
	Print operations	20
	Unique application names	64
	Operations currently in error	264
	Reruns of an operation	264
	Potential predecessor occurrences	32
	Potential successor occurrences	24
	Operations for which job log information has been collected	111
	Stand-alone clean up	70
	Restart and clean up operinfo retrieved	44
	Number of occurrences	43
Extended data (EQQXDnDS and EQQNXDDS)	Header record (one only)	244
	Bind requests	565
JCL repository (EQQJSnDS)	Number of jobs and started tasks	80
	Total lines of JCL	80
	Operations for which job log information has been collected	107
	Total lines of job log information	143
Note: As a base, calculate a figure for all your jobs and started tasks controlled by HCL Workload Automation for Z. Add to this figure the expected space required for jobs and started tasks in the current plan.		
Long-term plan (EQQLTDS)	Header record (one only)	92
	Occurrences	160
	External dependencies	35
	Operations changed in the LTP dialog	58

Table 23. Calculations of VSAM data set size (continued)

Data set	Size in bytes is total of:	
	Number of	**Multiplied by**
Operator instruction (EQQOIDS)	Instructions Instruction lines	78 72
Special resource database (EQQRDDS)	Resource definitions Defined intervals Entries in the WS connect table	216 48 8
Side information file (EQQSIDS)	ETT requests	128
Workstation/calendar (EQQWSDS)	Calendars Calendar dates Periods Period origin dates Workstation closed dates Workstations Workstation access method data Interval dates Intervals	96 52 94 6 80 124 72 52 32

Note:

1. Use the current plan data set calculation (EQQCPnDS) for the new current plan data sets (EQQNCPDS and EQQSCPDS).
2. Use the long-term-plan data set calculation (EQQLTDS) for the long-term-plan work data set (EQQLDDS) and the long-term-plan backup (EQQLTBKP).
3. Use the special resource database calculation (EQQRDDS) for the current plan extension data set (EQQCXDS) and the new current plan extension (EQQNCXDS).

Consider the following information when allocating VSAM data sets.

Archive of current plan (EQQACPDS)

The ACP file is a copy of the old current plan that is created by the database archive job (EQQDBARC). The file name is passed to the job with the parameter VSAMBKP.

Application description data set (EQQADDS)

The application description data set contains application descriptions and JCL variable tables. This data set is allocated as a spanned data set by EQQPCS01. It has a default maximum record size of 131 072. This allocation limits the variable definitions in a variable table to 275   (131 072/476 = 275), provided there are no variable dependencies. If you also use variable dependencies, the number of variables in a JCL variable table is less than 275.

If you use a greater number of variable definitions in a variable table, allocate the application description data set with a greater record size. To calculate how great the record size should be, use this method:

```
LRECL = 86 + (maximum number of variables in one table * 476) +
(number of variable dependencies * 88)
```

where 86 is the length of the header record, 476 is the length of each variable record and 88 is the length of each variable dependency record.

This VSAM data set must be allocated with share option set to 3 SHR(3). Do not use share option 2 or 1.

Note: HCL Workload Automation for Z supports VSAM extended addressability for the AD data set, therefore its size can exceed 4 GB.

Current plan data sets (EQQCP*n*DS)

The current plan VSAM files are opened and closed many times by HCL Workload Automation for Z during normal processing. If HCL Workload Automation for Z is unable to open one of the files, for example if the file is already opened by another job or TSO user, the normal mode manager (NMM), is terminated. The NMM issues message EQQN027E which reports the reason for the unexpected termination. You can issue a `MODIFY` command to restart the NMM subtask.

It is recommended that you do not access the current plan files from outside the HCL Workload Automation for Z address space. Backups of current plan information should be taken from the new current plan (EQQNCPDS). Shut down the Z controller address space if full-pack backups are taken of the volumes where the data sets reside.

Note: HCL Workload Automation for Z supports VSAM extended addressability for CP*n* and NCP data sets, therefore their size can exceed 4 GB.

JCL repository data sets (EQQJS*n*DS)

Take special care when allocating the JCL repository data sets. The following information describes the allocation and use of these data sets.

HCL Workload Automation for Z maintains its own copy of JCL in the JCL repository data set for every job that it submits in the current plan. It uses a primary and alternate data set for the JCL repository, EQQJS1DS and EQQJS2DS. It reorganizes the JCL repository data set that is in use by copying it to the alternate data set and then switching over to use 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 often the automatic copy process should occur.

Use the EQQPCS01 sample job created by the EQQJOBS installation aid to allocate the JS data sets. This job allocates the JS data sets with the SPANNED attribute and maximum record size 180000. This limits the maximum number of JCL statements to 2249 for any one job. If you run jobs with a greater number of JCL statements, increase the record size.

Calculate the required record size, in bytes, by multiplying the number of JCL statements in your largest job by 80, and add an extra 80 bytes for the header record. If you define your JS file without SPANNED, the greatest maximum record size that you can specify is 32760 bytes. This lets you store a job with up to 408 JCL records. If you define the JS data sets with SPANNED, the maximum record size you can specify is slightly less than a control area (CA). If you use the EQQUX002 exit, the largest job that can be returned by this exit is 7599 JCL records. Consider this, when you define the maximum record size of the JS data sets.

Note: HCL Workload Automation for Z supports VSAM extended addressability for JS data sets, therefore their size can exceed 4 GB.

Operator Instruction data set (EQQOIDS)

The operator instruction (OI) database contains operator instructions, each of which corresponds to an operation in the AD database and provides specific instructions about how this operation has to be handled.

This VSAM data set must be allocated with share option set to 3 SHR(3). Do not use share option 2 or 1.

Current plan backup copy data set (EQQSCPDS)

During the creation of the current plan, the SCP data set is used as a CP backup copy for the integration with IBM® Tivoli® Monitoring.

This VSAM data set must be allocated with share option set to 3 SHR(3). Do not use share option 2 or 1.

Note: HCL Workload Automation for Z supports VSAM extended addressability for SCP data sets, therefore their size can exceed 4 GB.

Data sets for step awareness (EQQSTDS and EQQNSTDS)

The EQQSTDS and EQQNSTDS data sets are used to enable the step awareness function. They are updated only by the controllers and appropriate cleanup actions are performed at the end of the CP turnover process. For details about the current plan turnover, see the *Diagnosis Guide and Reference*.

Data sets for extended data (EQQXDnDS)

The extended data VSAM files are opened and closed by HCL Workload Automation for Z together with the current plan VSAM files. For this reason, the same considerations for the current plan data sets apply to the data sets for the extended data.

Allocating Restart and Cleanup VSAM data sets

Use the EQQPCS07 member generated by the EQQJOBS installation aid. It is contained in the output library specified on the CREATE SAMPLE JOB JCL panel (EQQJOBS8). Submit the EQQPCS07 job to define and initialize the Restart and Clean Up of VSAM files.

Restart and cleanup data sets (EQQPKI*xx*, EQQSKI*xx*, and EQQSDF*xx*)

Every HCL Workload Automation for Z address space that uses the Restart and Cleanup function requires the allocation of a local VSAM repository for the structured information related to each job run.

These data sets have the same structure as the Data Store VSAM files and can be allocated by running the EQQPCS07 sample. Consider that every HCL Workload Automation for Z requires the allocation of a unique local VSAM repository.

The set of data sets allocated by EQQPCS07 is used by the controller started tasks and it is different from the similar set allocated for the data store started task.

Allocating non-VSAM data sets

This section describes the physical sequential (PS) and partitioned (PDS) data sets. Table 24: HCL Workload Automation for Z non-VSAM data sets on page 121 shows the non-VSAM data sets and their characteristics. Before you allocate the non-VSAM data sets, review the following sections, which contain important information about each of these data sets.

For all the sequential data sets listed below, the current version of HCL Workload Automation for Z supports DSNTYPE LARGE, which allows allocation of sequential data sets larger than 65535 tracks.

Table 24. HCL Workload Automation for Z non-VSAM data sets

Sample	DD Name	RECFM	LRECL	BLKSIZE	DSORG	Data set
EQQPCS02	AUDITPRT	FBA	133	13300	PS	Input to EQQAUDIT
EQQPCS01	–	U	–	6300	PS	CLIST library (optional)
EQQPCS01	EQQCKPT	U	–	8200	PS	Checkpoint
EQQPCS01	EQQFLEX	U	–	2400	PS	License information
EQQPCS01	EQQBKPT	U	–	8200	PS	Backup checkpoint
	EQQDL*nn*	U	–	6300	PS	Dual job-tracking-log
EQQPCS01	EQQDMSG	VBA	84	3120	PS	HCL Workload Automation for Z diagnostic message and trace
EQQPCS01	EQQEMAIL	FB	80	3120	PDS	HCL Workload Automation for Z email
EQQPCS01	EQQINCID	FB	80	3120	PDS	HCL Workload Automation for Z incident notification
EQQPCS01	EQQSMTP	FB	80	3120	PDS	SMTP data set (internal reader)
EQQPCS02	EQQDUMP	FB	80	3120	PS	HCL Workload Automation for Z diagnostic
EQQPCS11	EQQDUMP	FB	80	3120	PS	Diagnostic for Output collector

Table 24. HCL Workload Automation for Z non-VSAM data sets (continued)

Sample	DD Name	RECFM	LRECL	BLKSIZE	DSORG	Data set
EQQPCS02	EQQEVDS/ EQQEVD*nn*/ EQQHTTP0	F	100	100	PSU	Event
EQQPCS01	EQQEVLIB	FB	80	3120	PDS	Event-driven workload automation (EDWA) configuration file repository
EQQPCS02	EQQINCWK	FB	80	3120	PS	JCC incident work
EQQPCS01	EQQJBLIB	FB	80	3120	PDS	Job library
EQQPCS01	EQQJCLIB	FB	80	3120	PDS	JCC message table
EQQPCS01	EQQJTABL	F	240	240	PS	Critical job table log file
EQQPCS01	EQQJTARC	U	–	6300	PS	Job-tracking archive
EQQPCS01	EQQJT*nn*	U	–	6300	PS	Job-tracking-log
EQQPCS14	EQQDBARC	U	–	6300	PS	Extended-auditing archive
EQQPCS14	EQQDB*nn*	U	–	6300	PS	Extended-auditing log
EQQPCS01	EQQLOGRC	F	128	128	PS	Joblog and Restart Information pending requests Log data set
EQQPCS02	EQQLOOP	VBA	125	1632	PS	Loop analysis message log
EQQPCS02	EQQMLOG	VBA	125	1632	PS	Message log
EQQPCS11	EQQMLOG	VBA	125	1632	PS	Message log for Output collector started task
EQQPCS01	EQQMONDS	F	160	160	PSU	Monitoring task data set used to store events for IBM® Tivoli® Monitoring
EQQPCS09	EQQOCPBK	–	–	–	–	Data set to allocate the GDG root. The GDG entry is allocated during DP batch run and contains a backup of the old current plan.
EQQPCS11	EQQOUCEV	F	160	160	PSU	Stores events used in the communication between the controller and Output collector

Table 24. HCL Workload Automation for Z non-VSAM data sets (continued)

Sample	DD Name	RECFM	LRECL	BLKSIZE	DSORG	Data set
						for retrieving job logs from the z-centric environment.
EQQPCS11	EQQOUCKP	FB	80	3120	PDSE	Request checkpoint data set used by Output collector as it reads and processes events in the EQQOUCEV data set.
EQQPCS01	EQQPARM	FB	80	3120	PDS	Initialization-statement library
EQQPCS01	EQQPRLIB	FB	80	3120	PDS	Automatic-recovery-procedure library
EQQPCS01	EQQSTC	FB	80	3120	PDS	Started-task submit
EQQPCS02	EQQTROUT	VB	32756	32760	PS	Input to EQQAUDIT
–	EQQYPARM				PDS/PS	PIF
EQQPCS01 EQQPCS02	SYSMDUMP	F	4160	4160	PS	System dump data set
EQQPCS11	SYSMDUMP	F	4160	4160	PS	System dump data set for Output collector
–	–	FB	80	3120	PS	Job-completion-checker incident log

You can allocate these non-VSAM data sets by using the samples listed in Table 24: HCL Workload Automation for Z non-VSAM data sets on page 121, which are generated by the EQQJOBS installation aid.

Note: The data sets cannot be defined as compressed SMS data sets. If you have not customized the members, as described in Step 9. Allocating data sets on page 112, you can allocate a partitioned data set by running a job like the following example. In this example, a started-task-submit data set (EQQSTC) is allocated.

```
Allocating an HCL Workload Automation for Z partitioned data set
//ALLOCPDS JOB STATEMENT PARAMETERS
//*-----------------------------------------*
//* ALLOCATE A PARTITIONED DATA SET *
//*-----------------------------------------*
//ALLOC   EXEC PGM=IEFBR14
//SYSUT1   DD  DSN=OPCESA.INST.EQQSTC,
//             DISP=(,CATLG),
//             VOL=SER=volser,
//             SPACE=(TRK,(5,0,1)),
//              UNIT=3390,
//              DCB=(RECFM=FB,LRECL=80,BLKSIZE=3120)
```

To allocate an HCL Workload Automation for Z sequential data set, you can run a job like the following example. In this example, an event data set (EQQEVDS) is allocated. The IEBGENER utility ensures that the allocated data set has an end-of-file marker in it.

Note: If you allocate HCL Workload Automation for Z data sets using your own jobs, ensure that they have an end-of-file marker in them.

Example

```
Allocating an HCL Workload Automation for Z sequential data set
//ALLOCPS  JOB  STATEMENT PARAMETERS
//*---------------------------------------*
//* ALLOCATE A SEQUENTIAL DATA SET *
//*---------------------------------------*
//ALLOC   EXEC PGM=IEBGENER
//SYSPRINT DD  DUMMY
//SYSUT1   DD  DUMMY,DCB=(RECFM=F,BLKSIZE=100,LRECL=100)
//SYSUT2   DD  DSN=OPCESA.INST.EVENTS,
//             DISP=(NEW,CATLG),
//             UNIT=3390,
//             VOL=SER=volser,
//             SPACE=(CYL,3,,CONTIG),
//             DCB=(RECFM=F,BLKSIZE=100,LRECL=100,DSORG=PS)
//SYSIN    DD  DUMMY
```

The following sections describe the HCL Workload Automation for Z non-VSAM data sets. They provides you important information to be considered when allocating your data sets.

Internal reader data set (EQQBRDS)

When an HCL Workload Automation for Z subsystem is used to submit work, specify the internal reader data set, EQQBRDS, in your started-task procedures. The DD statement must contain the external-writer-data set name, INTRDR, and the class of the internal reader. The class you specify is used as a default message class for jobs that do not have a MSGCLASS parameter specified on their job cards.

Example

```
Example internal reader DD statement
//EQQBRDS  DD  SYSOUT=(A,INTRDR)
```

Primary checkpoint data sets (EQQCKPT)

HCL Workload Automation for Z uses the checkpoint data set to save the current status of the HCL Workload Automation for Z system. If the Z controller is stopped and restarted, HCL Workload Automation for Z uses the checkpoint data set to return the system to the same state as when it was stopped, ready to continue processing.

Note: HCL Workload Automation for Z uses the backup checkpoint data set (EQQBKPT) only when a remote hot standby controller is being used.

HCL Workload Automation for Z automatically formats the checkpoint data set the first time it is used. In its initial state, the checkpoint data set specifies that a new current plan exists. The new current plan is defined by DD name EQQNCPDS. HCL Workload Automation for Z attempts to copy the new plan and make it the current plan. If the copy is successful, HCL Workload Automation for Z is fully operational. If the copy is not successful, HCL Workload Automation for Z has become active without a current plan.

Note:

1. A strong relationship exists between the HCL Workload Automation for Z checkpoint data set and the current plan data set. There is also a strong relationship between the event positioning record (EPR) in the EQQCKPT data set and a specific destination and, therefore, also to a specific event data set. If this relationship is broken, the results of the synchronization processing at controller restart can be unpredictable. This is because events could be lost or reprocessed. Ensure that you do not accidentally delete or overwrite the checkpoint data set
2. To initialize the checkpoint data set, the OPCHOST keyword of the OPCOPTS initialization statement must be set to its default value, that is, OPCHOST(YES), the first time the scheduler is started. With OPCHOST(YES), the NMM initializes the checkpoint data set with FMID and LEVEL corresponding to SSX.

   The OPCHOST value can then be changed. For example, you can change the value to OPCHOST(PLEX) when the subsystem is used as the controlling system in XCF.

The space allocation for the data set must be at least 15 tracks. This allocation can accommodate 1000 workstation destinations.

Backup checkpoint data set (EQQBKPT)

HCL Workload Automation for Z uses the backup checkpoint data set only when a remote hot standby controller is being used. It is not possible to make a recovery.

Unless you have mistakenly mentioned that you are using a remote hot standby controller when running EQQJOBS, this data set must not be allocated and the related DD card must not be in the JCL.

Diagnostic data sets (EQQDMSG, EQQDUMP, and SYSMDUMP)

Allocate diagnostic data sets for HCL Workload Automation for Z address spaces, dialog users, batch jobs, and server.

Diagnostic message and trace data set (EQQDMSG)

You should allocate EQQDMSG for each dialog user. You can allocate EQQDMSG either as a SYSOUT data set or as a DASD data set. If EQQDMSG is not defined, output is written to EQQDUMP.

Diagnostic data set (EQQDUMP)

The tracker, Z controller, and server write debugging information to diagnostic data sets when validity checking discovers internal error conditions. When diagnostic information is logged, a 3999 user abend normally accompanies it. For service purposes, always include an EQQDUMP DD statement for every HCL Workload Automation for Z address space, dialog user, batch job, and server.

Diagnostic data sets are usually allocated as DASD data sets, but they can also be allocated to SYSOUT.

Dump data set (SYSMDUMP)

EQQPCS02 contains two allocations for the SYSMDUMP data set. For an HCL Workload Automation for Z address space, the data set is allocated with the low-level qualifier SYSDUMP. Allocate a unique SYSMDUMP data set for every HCL Workload Automation for Z address space. For the scheduler server jobs, SYSMDUMP is allocated with the low-level qualifier SYSDUMPS. EQQPCS01 contains the allocation for the SYSMDUMP data set for HCL Workload Automation for Z batch jobs; this data set is allocated with the low-level qualifier SYSDUMPB. The HCL Workload Automation for Z batch jobs can use the same data set. It is allocated with a disposition of MOD in the JCL tailored by EQQJOBS.

Furthermore, SYSMDUMP data sets should be defined with a UACC of UPDATE, that is, WRITE-ENABLED to all user IDs under which a job scheduled by HCL Workload Automation for Z might possibly be submitted. This is because the SUBMIT SUBTASK of the controller or of the tracker which is submitting a given job might abend while running under the user exit EQQUX001 supplied user ID (RUSER user ID) rather than under the user ID associated with the started task. If this occurs, DUMPTASK fails with an ABEND913 if the user ID in control does not have WRITE access to the SYSMDUMP data set.

The UACC of UPDATE access should be defined to all PIF, dialog, and Dynamic Workload Console servers. If a user is not authorized to update the SYSMDUMP data set, and a server failure occurs while running a request for that user, DUMPTASK fails with an ABEND 912. No diagnostic data will be captured.

Note: If you allocate the SYSMDUMP data set on your own, consider that the SYSMDUMP allocation can also be changed to use LRECL=4160,RECFM=FBS,BLKSIZE=n*4160 (where "n*4160" is a system-determined multiple block size) according to the new possibilities offered by IPCS and z/OS V1.6 or later. The EQQJOBS allocation has been left unchanged, for compatibility with previous allocated data sets.

Email data set (EQQEMAIL)

This data set includes the members that specify the rules for the emails that the Z controller sends when an alert occurs. The EQQEMAIL DD statement is required in the HCL Workload Automation for Z JCL procedure.

Note:

1. Unlikely other statements, in the EQQEMAIL DD statement data can be in columns 1 through 80.
2. Variable names cannot be split in two lines.

The EQQEMAIL data set contains a member named RULES with the rules that apply to the emails to be sent, and the members with the emails' text. You can set the default values in the following statements:

MAILOPTS

You define the defaults for the RULES member

ALERTS

In the MAIL parameter you define the default members for the emails' text.

For more details about these statements, see *Customization and Tuning*

For details about the emailing feature, see the section about sending emails when an alert condition occurs in *Managing the Workload*.

Incident data set (EQQINCID)

This data set includes the members that specify the rules for the incidents that the Z controller notifies through ServiceNow or a chat tool, when an alert occurs. The EQQINCID DD statement is required in the HCL Workload Automation for Z JCL procedure.

Note:

1. Unlikely other statements, in the EQQINCID DD statement data can be in columns 1 through 80.
2. Variable names cannot be split in two lines.

The EQQINCID data set contains the members with the incidents' text, and according to the incident notifying tool that you are using:

ServiceNow

A member named RULESTCK (with the rules that apply to the incidents to be opened)

A chat tool such as IBM Z ChatOps, Slack, Microsoft Teams, or Mattermost

A member named RULES (with the rules that apply to the incidents to be notified)

For details about how to have the Z controller:

- Open an incident through ServiceNow, see the section about opening incidents through a chat tool in *Managing the Workload*.
- Notify incidents through a chat tool, see the section about notifying incidents and sharing information through a chat tool in *Managing the Workload*.

License data set (EQQFLEX)

HCL Workload Automation for Z uses the license data set to store all the information related to the license for scheduling jobs on z-centric and dynamic agents.

Event data sets (EQQEVDS, EQQEVD*nn*, and EQQHTTP0)

Every HCL Workload Automation for Z address space requires a unique event data set. The data set is device-dependent and must have only a primary space allocation. Do *not* allocate any secondary space. The data set is formatted the first time it is used. Each time you use the data set, HCL Workload Automation for Z keeps a record of where to start. When the last track of the data set is written, HCL Workload Automation for Z starts writing on the first track again.

Note: The first time HCL Workload Automation for Z is started with a newly allocated event data set, an SD37 error occurs when HCL Workload Automation for Z formats the event data set. Do not treat this as an error.

The data set contains records that describe events created by HCL Workload Automation for Z job-tracking functions. An event-writer task writes to this data set; an event-reader task reads from it. The job-submit task also uses the event data set to checkpoint its activities, using the first record in the data set (the header record). The submit task in a controller address space takes these checkpoints when the computer workstation is the same system (the workstation destination is blank), so the address space needs the EQQEVDS event data set allocated even if there is no event writer task. When an event writer task is started in the controller address space, it shares the data set with the submit task.

The header record contains checkpoint information for up to 13 workstations per destination. If you plan to have more than 13 workstations defined to use a single destination, you can allocate the event data set with a large logical record length to accommodate the required number. To calculate the record length required, use this formula:

```
LRECL = (No-of-WS-with-this-destination * 6) + 22
```

Because the event data set provides a record of each event, events will not be lost if an event-processing component of HCL Workload Automation for Z must be restarted. The submit checkpointing process ensures that submit requests are synchronized with the Z controller, thereby preventing lost requests caused by communication failures.

Define enough space for a single extent data set so that it does not wrap around and write over itself before an event is processed. Two cylinders are enough at most installations. The space allocation must be at least 2 tracks when the record length is 100. There must be sufficient space in the event data set to accommodate 100 records. Consider this requirement if you will define the event data set with a record length greater than 100. For example if you define an LRECL of 15 000, the minimum space allocation is 34 tracks, which equates to 102 records and an event data set that would wrap around very quickly in most installations.

To aid performance, place the event data set on a device that has low activity. If you run programs that use the RESERVE macro, try to allocate the event data set on a device that is not reserved or where only short reserves are taken. The reserve period must be less than 5 minutes.

If you use the job log retrieval function, consider allocating the event data set with a greater LRECL value than that in Table 24: HCL Workload Automation for Z non-VSAM data sets on page 121. This improves performance because input/output

(I/O) operations will be reduced because fewer continuation (type NN) events will be created. You can specify 0, or a value from 100 to 32 000 bytes for LRECL. Any other value will cause the event writer to end, and message EQQW053E will be written to the message log. If you do not specify a value for LRECL or specify 0, the data set will be forced to have an LRECL of 100 when it is opened by HCL Workload Automation for Z. However, the data set must be **unblocked**: the block size must be equal to the logical record length. If you intend to activate job log retrieval function, use one of the these formulas to estimate the LRECL that you should specify:

Example

```
Calculating the optimum LRECL
LRECL=((NN/EV) * 20) + 100   OR   LRECL=(4 * N) + 100
```

In the first formula, NN is the number of continuation events, and EV is the number of all other events. Event types are found in position 21 of the event records. In the second formula, N is the average number of NN events per job. If your calculation yields a value of less than 110, there will be little or no improvement in performance. In this case, you should specify an LRECL value of 100.

You will probably need to test your system first to get an idea of the number and event types that are created. You can then reallocate the event data set when you have gathered information about the events created at your installation. But, before you reallocate an event data set, ensure that the current plan is completely up-to-date. You must also stop the event writer, and any event reader, that uses the data set.

Note: Do not move HCL Workload Automation for Z event data sets once they are allocated. They contain device-dependent information and cannot be copied from one device type to another, or moved on the same volume.
An event data set that is moved will be reinitialized. This causes all events in the data set to be lost. If you have DFHSM or a similar product installed, you should specify that HCL Workload Automation for Z event data sets are not migrated or moved.

Event-driven workload automation configuration file data set (EQQEVLIB)

This data set contains the configuration files required by the event-driven workload automation (EDWA) process. The configuration files, which are created by the EQQRXTRG program, are used by the trackers to monitor the event conditions. The event-driven workload automation configuration file data set is accessed by the controller, which, when configuration files are created or modified, deploy them to the trackers by storing the files into the data set identified by the EQQJCLIB DD card. This is the same data set to which the trackers' JCLs refer.

By using the event-driven workload automation configuration file data set, you can automate and centralize the deployment of configuration files to the trackers without having to use the EQQLSENT macro for each tracker.

Job library data set (EQQJBLIB)

The job library data set contains the JCL for the jobs and started tasks that HCL Workload Automation for Z will submit. It is required by a Z controller. If you already have a job library that you will use for HCL Workload Automation for Z purposes, specify this data set on the EQQJBLIB statement. If not, allocate one before you start the Z controller.

Note: Allocate the job library data set with a only primary space allocation. If a secondary allocation is defined and the library goes into an extent when HCL Workload Automation for Z is active, you must stop and restart the controller. Also, do not compress members in this PDS. For example, do not use the ISPF `PACK ON` command, because HCL Workload Automation for Z does not use ISPF services to read it.

The limitation of allocating the job library data set with only a primary space allocation is nota applicable for PDSE data sets.

Note: Each member in the EQQJBLIB must contain one job stream (only one job card), and the job name on the job card must match the job name in the HCL Workload Automation for Z scheduled operation.

Job-completion-checker data sets

You can optionally use the job completion checker (JCC) to scan SYSOUT for jobs and started tasks. Depending on the JCC functions you want to use, allocate at least one of the three data sets associated with the JCC:

JCC-message-table library (EQQJCLIB)

If the success or failure of a job or started task cannot be determined by system completion codes, the JCC function can be used to scan the SYSOUT created and set an appropriate error code. You determine how the SYSOUT data is scanned by creating JCC message tables. A general message table (EQQGJCCT) must be defined. Job-specific message tables can be created to search for specific data strings in particular jobs. These tables are stored in the PDS with a member name that matches the job name.

Every HCL Workload Automation for Z subsystem where you start the JCC task must have access to a message table library. If you want, you can use the same message table library for all HCL Workload Automation for Z systems.

If you use the data set-triggering function, the data set-selection table (EQQEVLST or EQQDSLST) must be stored in EQQJCLIB.

Note: Allocate the JCC message table data set with only primary space allocation. The limitation is not applicable for PDSE data sets.

JCC-incident-log data set

You can optionally use the JCC to write records to an incident log data set. This data set is defined by the INCDSN parameter of the JCCOPTS statement.

When scanning SYSOUT data sets, the JCC recognizes events that you define as unusual. If the EQQUX006 exit is loaded by HCL Workload Automation for Z, the JCC records these events in the incident log data set. The incident log data set can be shared by several JCC tasks running on the same system or on different systems. The data set can also be updated manually or even reallocated while the JCC is active. If the JCC is unable to write to the incident log, the incident work data set is used instead.

JCC-incident work data set (EQQINCWK)

Occasionally, the JCC cannot allocate the incident log data set. This can happen if another subsystem or an HCL Workload Automation for Z user has already accessed the data set. In this case, the JCC writes to the incident work file, EQQINCWK, instead. If it is not empty, the work file is copied and emptied each time the incident log data set is allocated.

Job-tracking data sets (EQQJTARC, EQQJT*nn*, EQQDL*nn*)

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

- Job-tracking logs (EQQJT*nn*)
- Dual job-tracking logs (EQQDL*nn*)
- Job-tracking archive (EQQJTARC)

You must allocate EQQJTARC and at least two job-tracking logs (EQQJT01 and EQQJT02) for a Z controller. The actual number of JT logs that you should allocate is determined by the value that you specify on the JTLOGS keyword of the JTOPTS initialization statement. If you decide to allocate three job-tracking logs, specify the DD names EQQJT01, EQQJT02, and EQQJT03. If you specified EQQJT01, EQQJT02, and EQQJT04, an error occurs and HCL Workload Automation for Z terminates. HCL Workload Automation for Z uses the job-tracking logs in turn. When a current plan backup is performed, the active log is appended to EQQJTARC data set.

The size of the CP files, JT and JTARC, can become large, but with appropriate tuning of the size and of the DP frequency, they will not allocate additional extents. If necessary, use the allocation of additional extents (not additional volumes, because only extent allocation is supported in the shipped JT allocation samples). The JTLOG keyword default defines five job-tracking logs. It is recommended that you specify at least three job-tracking logs. Job-tracking logs are switched at every current plan backup. If the interval between backups is very low and JTLOGS(2) is specified, the previously used job-tracking log might not have been archived before HCL Workload Automation for Z must switch again. If it cannot switch successfully, the normal-mode-manager (NMM) subtask is automatically shut down, preventing further updates to the current plan.

You can optionally allocate dual JT logs. These logs are identified by the EQQDL*nn* DD names in the Z controller started-task JCL. Allocate the same number of dual JT logs as JT logs. The numeric suffixes, *nn*, must be the same as for the JT logs, because HCL Workload Automation for Z uses the logs with the same number: EQQJT01 and EQQDL01, EQQJT02 and EQQDL02, and so on. HCL Workload Automation for Z writes job-tracking information to both logs, so that if the active JT log is lost it can be restored from the dual log, and HCL Workload Automation for Z can be restarted without losing any events. To achieve the maximum benefit from dual JT logs, you should allocate them:

- With the same attributes as the JT logs
- With at least the same amount of space as the JT logs
- On alternate I/O paths and physical volumes than their corresponding JT logs

HCL Workload Automation for Z tries to use dual JT logs if you specify DUAL(YES) on the JTOPTS initialization statement of a Z controller.

The job-tracking-archive data set accumulates all job-tracking data between successive creations of a new current plan (NCP). Therefore, allocate EQQJTARC with enough space for all job-tracking records that are created between daily planning jobs; that is, extend or replan of the current plan. In other words, be sure that you allocate for EQQJTARC an equal or greater amount of space than the total of the space you allocate for the JT files, or you will get a system error. When the daily planning batch job is run, the active job-tracking log is appended to EQQJTARC, and the JT log is switched. The archive log, EQQJTARC, is then copied to the track log data set referenced by the EQQTROUT DD name during the daily planning process. When HCL Workload Automation for Z takes over the NCP, the archive data set is emptied.

For a detailed description of the HCL Workload Automation for Z recovery procedures that use the job-tracking data sets, see *Customization and Tuning*.

Extended-auditing data sets (EQQDBARC, EQQDBnn)

If you have set AUDIT AMOUNT(EXTENDED), extended-auditing data sets are a log of records that show the values that were set before and after the database was changed. They comprise:

- Extended-auditing logs (EQQDB*nn*)
- Extended-auditing archive (EQQDBARC)

You can control the level of information to be logged by editing the SYSIN card in the EQQAUDIB sample, as follows:

1. Set the input file to use:

   DBX

   For EQQDB*nn*

   DBR

   For EQQDROUT
2. Specify the level of information to log in the report (this applies only to the delete or add action):

   K

   Only the key is reported.

   S

   Summary information is reported (default).

   F

   Complete information is reported.
3. Set the database to audit. If you do not specify any value, all the databases are audited.

   AD

   Application Description

   CAL

   Calendar

JV

Job Variable Table

OI

Operator Instructions

PER

Period

RD

Special Resource

RUN

Run cycle group

WS

Workstation

4. Specify the database key to audit. If you did not set a database, this vale is ignored.

ad_name

For the AD database.

calendar_name

For the CAL database.

jv_table_name

For the Job Variable Table database.

oi_ad_name oi_op_num

For the Operator Instructions database.

period_name

For the Period database

special_resource_name

Special Resource

run_cycle_group_name

Run cycle group

ws_name

For the WS database

For example, a SYSIN card to audit the EQQDB*nn* database by logging the complete information about the workstation named `CPU1` in the WS database, looks like the following:

```
//SYSIN    DD *
DBXFWS  CPU1
/*
```

You must allocate EQQDBARC and at least two extended-auditing logs (EQQDB01 and EQQDB02) for a Z controller. The actual number of DB logs that you should allocate is determined by the value that you specify on the JTLOGS keyword of the JTOPTS initialization statement. If you decide to allocate three extended-auditing logs, specify the DD names EQQDB01, EQQDB02, and EQQDB03. If you specified EQQDB01, EQQDB02, and EQQDB04, an error occurs and HCL Workload Automation for Z terminates. HCL Workload Automation for Z uses the extended-auditing logs in turn. When a current plan backup is performed, the active log is appended to EQQDBARC data set.

The size of the DB and DBARC data sets can become large, but with appropriate tuning of the size and of the DP frequency, they will not allocate additional extents. If necessary, use the allocation of additional extents (not additional volumes, because only extent allocation is supported in the shipped DB allocation samples).

The extended-auditing-archive data set accumulates all extended-auditing data between successive creations of a new current plan (NCP). Therefore, allocate EQQDBARC with enough space for all extended-auditing records that are created between daily planning jobs; that is, extend or replan of the current plan. In other words, ensure that you allocate for EQQDBARC an equal or greater amount of space than the total of the space you allocate for the DB files, or you will get a system error. When the daily planning batch job is run, the active extended-auditing log is appended to EQQDBARC, and the DB log is switched. The archive log, EQQDBARC, is then copied to the extended-auditing log data set referenced by the EQQDBOUT DD name during the daily planning process. When HCL Workload Automation for Z takes over the NCP, the EQQDBARC data set is emptied.

HCL Workload Automation for Z recovery procedures that use the extended-auditing data sets are described in *Customization and Tuning*.

Message log data set (EQQMLOG)

The message log data set can be written to SYSOUT or a data set. The data control block (DCB) for this data set is defined by HCL Workload Automation for Z as follows:

Example

```
EQQMLOG DCB attributes
DCB=(RECFM=VBA,LRECL=125,BLKSIZE=1632)
```

If the message log data set becomes full during initialization, or when a subtask is restarted, HCL Workload Automation for Z will abend with error code SD37. In either case, you must stop HCL Workload Automation for Z and reallocate the message log data set with more space. In all other circumstances, if the data set fills up, HCL Workload Automation for Z redirects messages to the system log instead.

Note: The scheduler ABENDs with error code sb37 or sd37 if the message log data set becomes full under any of the following circumstances:

- During initialization
- When a subtask is restarted
- While processing any modify command which requires parsing of initialization parameters or specifies the newnoerr, noerrmem(member), or lstnoerr options

In the last case, the ABEND also occurs if the EQQMLOG is already full when any such command is issued. In all these cases you must reallocate more space to the message log data set. In all the other cases, if the data set fills up, the scheduler redirects messages to the system log instead.

EQQPCS02 contains two allocations for the EQQMLOG data set. For an HCL Workload Automation for Z address space, the data set is allocated with the low-level qualifier MLOG. For the scheduler server jobs, the data set is allocated with the low-level qualifier MLOGS.

Note: If you allocate the message log data set on DASD, define a different data set for HCL Workload Automation for Z batch program. The data set must also be different from the one used by each HCL Workload Automation for Z address space (controller, standby controller, tracker, and server). The data set cannot be shared.

See also Using two message log (MLOG) data sets on page 49.

Loop analysis log data set (EQQLOOP)

The loop analysis log data set can be written to SYSOUT or a data set. The data control block (DCB) for this data set is defined by HCL Workload Automation for Z as follows:

Example

```
EQQLOOP DCB attributes
DCB=(RECFM=VBA,LRECL=125,BLKSIZE=1632)
```

This data set is defined the same way as for EQQMLOG, but it is specific for loop analysis and is populated only if a loop condition occurs. It is required by daily planning batch programs (extend, replan, and trial).

SMTP data set (EQQSMTP)

This data set is needed only if you have set `ALERTS MAIL` to have the Z controller send an email when a specific event occurs.

Specify the EQQSMTP DD statement in the HCL Workload Automation for Z JCL procedure, to indicate the internal reader data set used for sending the email via z/OS. The DD statement must contain the external-writer-data set name, INTRDR, and the class of the internal reader. The class you specify is used as a default message class for jobs that send the email.

For details about the ALERTS statement, see *Customization and Tuning*

Example

```
Example internal reader DD statement
//EQQSMTP   DD   SYSOUT=(B,INTRDR)
```

Controller and Output collector communication data sets

The EQQOUCEV and EQQOUCKP data sets are used for the communication that takes place between the controller and Output collector in the retrieval process of the job logs produced in the z-centric environment. The communication process is based upon events. Every time a job in the z-centric environment completes or terminates, the controller queues an event for the output collector with the information necessary to identify the job and the agent that run it.

The communication process is as follows:

1. The Event Manager task of the controller writes an event in EQQOUCEV every time a job completes or terminates and updates the next-to-write counter.
2. The Output collector started task reads an event from this data set, checkpoints it in EQQOUCKP, dispatches it to the proper thread, and marks the event as processed moving to the next-to-read index in the data set header.

EQQOUCEV is a sequential data set organized in records. Includes a header pointing to the next-to-read and next-to-write records.

EQQOUCKP is a partitioned data set. It is used to checkpoint the incoming requests to prevent their loss in case of unplanned closures. The EQQOUCEV queue manager writes the request in EQQOUCKP before placing it in the destination queue in memory while the thread that collects the log from the agent deletes the request after it is satisfied. The member name must be unique but not necessarily meaningful (for example J0000001 or J0000002).

Parameter library (EQQPARM)

Each HCL Workload Automation for Z subsystem reads members of a parameter library when it is started. Parameter library members (residing in library extent), that have been created, cannot be accessed after they have been opened. To avoid this problem, the data set that defines the EQQPARM library should be allocated without any secondary extents. The limitation is not applicable for PDSE data sets. The library contains initialization statements that define runtime options for the subsystem. Allocate at least one parameter library for your HCL Workload Automation for Z systems. You can keep the parameters for all your subsystems in one library, as long as it resides on a DASD volume that is accessible by all systems.

PIF parameter data set (EQQYPARM)

Allocate the PIF parameter data set if you intend to use a programming interface to HCL Workload Automation for Z. The data set can be sequential or partitioned. In the PIF parameter file you specify how requests from the programming interface should be processed by HCL Workload Automation for Z. By defining an INIT initialization statement in the PIF parameter data set, you override the global settings of the INTFOPTS statement.

For a detailed description of the initialization statements, see *Customization and Tuning*.

Automatic-recovery-procedure library (EQQPRLIB)

Allocate a data set for the automatic-recovery-procedure library if you intend to use the HCL Workload Automation for Z automatic-recovery function. The library is used by the `ADDPROC` JCL rebuild parameter of the JCL recovery statement. This parameter lets you include JCL procedures in a failing job or started task before it is restarted.

Started-task-submit data set (EQQSTC)

The started-task-submit data set is used by HCL Workload Automation for Z to temporarily store JCL when a started task is to be started. Use these attributes for this data set:

Example

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

Include an EQQSTC in the JES PROCLIB concatenation on each system where HCL Workload Automation for Z schedules started-task operations. The data set is used as a temporary staging area for the started-task JCL procedure. When the start command has been issued for the task and control for the task has passed to JES, HCL Workload Automation for Z deletes the JCL by resetting the PDS. This means that you never need to compress the data set. For more information, see Implementing support for started-task operations on page 141.

Note: HCL Workload Automation for Z does not support partitioned data set extended (PDSE) libraries for a started-task-submit data set.

Allocating Data Store data sets

At this stage of your installation, use the EQQPCS04 member generated by the EQQJOBS installation aid. It is contained in the output library specified on the CREATE DATA STORE SAMPLES panel (EQQJOBS5). Submit the EQQPCS04 job to define and initialize the Data Store VSAM files.

The Data Store VSAM files can be of three types:

Unstructured

The type associated to EQQUDFxx; used to save joblogs. These files are allocated only when the Joblog Retrieval option in panel EQQJOBS7 is set to Y. The UDF data sets (DDNAME EQQUDFNN) are used only by the data store started task.

Structured

The type associated to EQQSDFxx; used to save structured joblog information. These files are required. The EQQSDFXX data sets allocated for the data store have the same structure as the EQQSDFXX data sets used by the controller but they are a different set.

KSDS

The type used for EQQPKIxx and EQQSKIxx. The EQQPKIXX and EQQSKIXX data sets allocated for the data store have the same structure as the structure of the EQQPKIXX and EQQSKIXX data sets used by the controller but they are a different set.

They are listed and described in Table 25: Data Store VSAM data sets on page 138:

Table 25. Data Store VSAM data sets

Sample	DD Name	Rec. Type	Attributes	Share Option	Keys	Record Size	data set
EQQPCS04	EQQPKIxx	KSDS	UNIQUE INDEXED	1, 3	34 0	77 77	Primary Index
EQQPCS04	EQQSDFxx	LINEAR	N/A	2 , 3	N/A	N/A	Data files
EQQPCS04	EQQSKIxx	KSDS	UNIQUE INDEXED	1, 3	38 0	76 32000	Secondary Index
EQQPCS04	EQQUDFxx	LINEAR	N/A	2 , 3	N/A	N/A	Data files

For information about how to estimate the size of the Data Store VSAM files, see *HCL Workload Automation for Z: Customization and Tuning*.

Allocating data sets for the Dynamic Workload Console reporting feature

Use the EQQPCS09 member generated by the EQQJOBS installation aid and contained in the output library specified on the CREATE SAMPLE JOB JCL panel (EQQJOBS3) to define and allocate:

- The GDG base entry used for old current plan backup which is created during the daily plan batch process, when you specify the BATCHOPTS statement with the JRUNHISTORY parameter set to YES. The GDG data set is identified in the daily planning EXTEND or REPLAN batch job by the EQQOCPBK ddname.
- The VSAM data set where the archiving process copies each generation data set. Allocate the VSAM data set with the same characteristics as the current plan VSAM data set, because it is used to store the old current plan.

For detailed information about the archiving process, see *Managing the Workload*.

Allocating the files and directories

The following features use files on UNIX™ System Services (USS):

- End-to-end scheduling with z-centric capabilities, if SSLKEYRINGTYPE is set to USS in the HTTPOPTS statement.
- Features running Java™ utilities:
  - Historical run data archiving for Dynamic Workload Console reporting
  - Event-driven workload automation for data set triggering
  - License computation for submitting jobs on z-centric and dynamic agents.

By default, the EQQJOBS installation aid sets the following paths for the following directories:

JAVA utilities enablement work directory (EQQJOBS9)

`/var/TWS/inst`

SSL for TCP/IP connection work directory (EQQJOBSC)

`/var/TWS/inst/ssl`

By keeping the default directories, if the end-to-end work directory is deleted, the Java™ and SSL work directories are also deleted. To avoid this problem, set different paths for the different work directories. For example:

JAVA utilities enablement work directory (EQQJOBS9)

`/var/TWS/JAVAUTL`

SSL for TCP/IP connection work directory (EQQJOBSC)

`/var/TWS/SSL`

To create the correct directories and files, run the following sample jobs for each controller that supports the specific feature:

- The EQQPCS08 sample, for the:
  Historical run data archiving.
  Event-driven workload automation.
  Jobs submission on z-centric and dynamic agents.

To run the previous samples, you must have one of the following permissions:

- UNIX™ System Services (USS) user ID (UID) equal to 0
- BPX.SUPERUSER FACILITY class profile in RACF®
- UID specified in the JCL in eqqUID and belonging to the group (GID) specified in the JCL in eqqGID

The user must also have the `/bin/sh` login shell defined in his OMVS section of the RACF® profile. Make sure that the login shell is set as a system default or use the following TSO command to define it:

```
ALTUSER username OMVS(PROGRAM('/bin/sh'))
```

To check the current settings:

1. Run the following TSO command:
   ```
   LISTUSER username OMVS
   ```
2. Look in the `PROGRAM` line of the OMVS section.

After running EQQPCS08, you find the following file in the work directory:

java/env.profile

Defines the environmental variable required by the Java™ utilities.

You can customize this file and enable the use of the DB2 installed on the host for running HCL Workload Automation for Z reports from the Dynamic Workload Console by following the procedure described in Setting up to use the DB2 on z/OS for reports on Dynamic Workload Console on page 139 to

Setting up to use the DB2 on z/OS for reports on Dynamic Workload Console

Customize the `env.profile` file, unloaded in the work directory by EQQPCS08, to use a DB2 on z/OS to run reports from the Dynamic Workload Console without having to install another DB2 on the distributed side.

To use the DB2 on z/OS to run HCL Workload Automation for Z reports on the Dynamic Workload Console:

1. Open the env.profile on page 139 file and uncomment and customize the following rows:

   ```
   #export DB2_JDBC_PATH=/usr/lpp/db2910_jdbc/classes
   #export CLASSPATH="$DB2_JDBC_PATH"/db2jcc.jar:"$DB2_JDBC_PATH"/db2jcc_license_cisuz.jar:
   "$CLASSPATH":
   ```

   where the path specified in the `DB2_JDBC_PATH` is the path where the `db2jcc.jar` and `db2jcc_license_cisuz.jar` files are stored on the z/OS system.

   These rows are preceded by the comment:

   ```
   #If DB2 is on z/OS, customize and uncomment the following line
   ```

2. To continue the setup, see the section about how to create the database in the z/OS environment in *Managing the Workload*.

Setting up the Workload Automation Programming Language environment

To create the correct environment to use the Workload Automation Programming Language, ensure that you run the EQQWPLCO sample job.

Step 10. Creating JCL procedures for address spaces

Perform this task for a tracker, Data Store, Z controller or standby Z controller, output collector.

You must define a JCL procedure or batch job for each HCL Workload Automation for Z address space.

For details, see Defining subsystems on page 90.

The EQQJOBS dialog generates several members in the output library that you specified. The following table lists the members that provide samples for the scheduler's address spaces:

Table 26. Started task JCL samples for HCL Workload Automation for Z address spaces

Address Space for:	Member
Controller and tracker	EQQCON (sample started task) EQQCONP (sample started task parameters)
Controller	EQQCONO (sample started task) EQQCONOP (sample started task parameters)
Tracker	EQQTRA (sample started task) EQQTRAP (sample started task parameters)
Server	EQQSER (sample started task) EQQSERP (sample started task parameters)
Data Store	EQQDST (sample started task) EQQDSTP (sample started task parameters)

Table 26. Started task JCL samples for HCL Workload Automation for Z address spaces (continued)

Address Space for:	Member
Output collector	EQQOUC (sample started task) EQQOUCP (sample started task parameters)

These members contain started task JCL that is tailored with the values you entered in the dialog. Tailor these members further, according to the data sets you require. Alternatively, you can copy a member from the SEQQSAMP library to one of your own libraries, and tailor it manually.

If you create a new library for your HCL Workload Automation for Z started-task procedures, remember to specify the library in the JES PROCLIB concatenation. Then you must restart JES to include the new library.

If you prefer, you can run HCL Workload Automation for Z as a batch job rather than as a started task. Here, the JCL can reside in any library and will require a job card, besides the JCL requirements in Table 27: HCL Workload Automation for Z required data sets on page 142.

Implementing support for started-task operations

The JCL procedures for started-task operations started by HCL Workload Automation for Z must be stored in a PDS concatenated on the EQQJBLIB DD name. You can include existing data sets, such as SYS1.PROCLIB, if you prefer. Preparation, tailoring, and variable substitution are handled the same way as for batch job operations. When a started-task operation is started by HCL Workload Automation for Z, the JCL procedure is written to the started-task-submit data set (EQQSTC) on the system where the operation is to be run. HCL Workload Automation for Z issues a START command for this procedure and then removes the JCL procedure from the EQQSTC data set.

JES2 users should specify the started-task-submit data set on the PROC*nn* DD statement of the JES2 JCL procedure on each z/OS system. The suffix *nn* is the value specified for the PROCLIB parameter of the STCCLASS statement in JES2PARM. To ensure that the correct version of the JCL procedure is started, place the EQQSTC data set first in the concatenation.

JES3 users should specify the started-task-submit data set on the IATPLB*nn* DD statement of the JES3 global system. The suffix *nn* is the value specified in the JES3 standards parameter STCPROC. To ensure that the correct JCL procedure will be started, place the EQQSTC data set first in the concatenation. For each submit task that is running on a JES3 local system in the JES3 complex, also include that data set in the JES3 global concatenation.

If you do not use the Restart and Cleanup function, you must follow the previous instructions to work with started-task operations. Otherwise, because the Restart and Cleanup function adds a job card to the procedures for scheduled STC workstation operations at the same time that it adds the //TIVDST*xx* 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.

You must add the EQQSTC data set to the IEFPDSI DD statement in MSTJCL*xx* instead of to the JES2 PROC*nn* or the JES3 global IATPLBnn DD statement as mentioned above.

In addition, all data sets listed in IEFPDSI must be included in the system master catalog.

Note:

1. To include EQQSTC, you must restart JES.
2. Do not use the BLDL parameter of the JES3 PROC statement to specify the procedure name of a started task that is to be scheduled by HCL Workload Automation for Z.

The EQQSTC data set can be shared by HCL Workload Automation for Z subsystems that run on the same z/OS® image. If you use *global resource serialization* (GRS), the EQQSTC data set can be shared by all z/OS® systems defined in the GRS ring if you propagate requests for the resource. To propagate the resource requests to all systems in the ring, define the resource SYSZDRK.data data set name in the SYSTEM inclusion RNL of the GRSRNLnn member of SYS1.PARMLIB. For more information about defining the GRS resource name list, see *z/OS® Initialization and Tuning Reference*.

Required data sets

Table 27: HCL Workload Automation for Z required data sets on page 142 shows the data sets required by an HCL Workload Automation for Z started task. Include the data sets in your JCL procedures as indicated in this table.

Table 27. HCL Workload Automation for Z required data sets

DD Name	Required by				Defines
	Controller	Tracker	Server	Data Store	
EQQADDS	✓				Application descriptions and JCL variable tables
EQQBRDS	✓	✓			A JES internal-reader
EQQCKPT	✓				Checkpoint data set
EQQCP1DS	✓				Primary current plan
EQQCP2DS	✓				Alternate current plan
EQQCXDS	✓				Current plan extension
EQQEVDS	✓	✓			Event data set for the submit checkpointing function and for the event-writer task
EQQEVLIB	✓				Configuration file repository for event-triggered resource handling
EQQFLEX	✓				License data set
EQQJBLIB	✓				JCL PDS libraries

Table 27. HCL Workload Automation for Z required data sets (continued)

DD Name	Required by				Defines
	Controller	Tracker	Server	Data Store	
EQQLOGRC	✓				Joblog and Restart Information pending requests log data set
EQQJS1DS	✓				Primary JCL repository
EQQJS2DS	✓				Alternate JCL repository
EQQJTABL	✓				Job table log file. The scheduler considers this data set as required only if you defined at least one critical job. Allocate it with the same size as EQQJTARC.
EQQJTARC	✓				Job-tracking archive
EQQJT*nn*	✓				Job-tracking logs
EQQLTDS	✓				Long-term plan
EQQMLIB	✓	✓	✓	✓	Message library
EQQMLOG	✓	✓	✓	✓	Output message log
EQQNCPDS	✓				New current plan
EQQNCXDS	✓				New current plan extension
EQQNXDDS	✓				NCP extension
EQQOIDS	✓				Operator instructions
EQQPARM	✓	✓	✓	✓	Parameter library
EQQRDDS	✓				Special resource descriptions
EQQSCPDS	✓				Current plan backup copy data set. Needed for integration with IBM® Tivoli® Monitoring.
EQQSIDS	✓				Side information; ETT criteria and configuration data
EQQWSDS	✓				Workstation, calendar and period descriptions
EQQXD1DS	✓				CP1 extension
EQQXD2DS	✓				CP2 extension

Note:

1. The data sets that are required for a Z controller are also required for a standby Z controller.
2. The number of job-tracking-log data sets to include depends on the value that you specify in the JTLOGS keyword of the JTOPTS initialization statement. Specify at least 3 job-tracking logs. The default value is 5.
3. You must specify EQQEVDS for a Z controller even if an event writer is not started in the Z controller address space. The EQQEVDS data set is used for submit checkpointing. It can be the same data set that is used by an event-writer function. Use a unique EQQEVDS for each address space.
4. In order to set the TCP/IP task up correctly, you need to change the scheduler start procedure to include the C runtime libraries (CEE.SCEERUN in the STEPLIB DD statement).

   If you have multiple TCP/IP stacks, or if the name you used for the procedure that started the TCPIP address space was not the default (TCPIP), then you must change the start procedure to include the SYSTCPD DD card to point to a data set containing the TCPIPJOBNAME parameter.

   The standard method to determine the connecting TCP/IP image is:
   - Connect the TCP/IP specified by TCPIPJOBNAME in the active TCPIP.DATA
   - Locate TCPIP.DATA using the SYSTCPD DD card.

Optional data sets

Table 28: HCL Workload Automation for Z optional data sets on page 144 shows the data sets that you can optionally include in your JCL procedures. Specify these data sets only if you want to use the function with which they are associated.

Table 28. HCL Workload Automation for Z optional data sets

DD Name	Can be used by					Defines
	Controller	Tracker	Server	Data Store	Output collector (empty)	
AUDITPRT	✓					Input to EQQAUDIT
EQQDL*nn*	✓					Dual job-tracking logs
EQQDUMP	✓	✓	✓			Diagnostic dump output
EQQEVD*nn*		✓				Event data set for an event-reader task
EQQINCWK		✓				JCC incident work file
EQQJCLIB		✓				JCC library for message tables and for data set triggering selection table
EQQMONDS	✓					Data set used by monitoring task to store events for IBM® Tivoli® Monitoring.
EQQOUCEV	✓					Data set used for the communication that takes place between the controller and

Table 28. HCL Workload Automation for Z optional data sets (continued)

DD Name	Can be used by					Defines
	Controller	Tracker	Server	Data Store	Output collector (empty)	
						Output collector in the retrieval process of the job logs produced in the z-centric environment.
EQQOUCKP	✓					Data set used for the communication that takes place between the controller and Output collector in the retrieval process of the job logs produced in the z-centric environment.
EQQPKI*xx*	✓					Primary index
EQQPRLIB	✓	✓				Automatic-recovery procedures
EQQSDF*nn*	✓					Structured data files
EQQSKI*xx*	✓					Secondary index
EQQSTC	✓	✓				Started-task-submit data set
EQQTROUT	✓					Input to EQQAUDIT
EQQUDF*nn*				✓		Unstructured data files
STEPLIB	✓	✓	✓			Load-module library
SYSMDUMP	✓	✓	✓			Dump data set

Note:

1. The optional data sets that you specify for a Z controller must also be specified for a standby Z controller.
2. If you use dual job-tracking, the number of dual job-tracking logs (EQQDL*nn*) must be the same as the number of job-tracking logs (EQQJT*nn*).
3. Include EQQDUMP and SYSMDUMP for diagnostic purposes.
4. The EQQEVD*nn* DD name identifies the event data set for an event-reader task. The *nn* value is the sequence number specified in the ERSEQNO keyword of the event reader that will process this data set. It is always a 2-digit number. That is, if the sequence number is less than 10, a leading 0 must be added.
5. Specify the EQQSTC data set if you use HCL Workload Automation for Z to schedule started-task operations.
6. Use the standard JCL naming conventions for each user-defined DD name; that is, 1-8 alphanumeric or national characters, of which the first character must be alphabetic or national.

7. The STDENV DD name can point to a sequential DS or a PDS member (for example, a member of the PARMLIB) in which the user can define environment variables to initialize Language Environment®. STDENV must have a F or FB format with a record length equal or greater than 80. In this data set/member you can put your environment variables specifying VARNAME=value. On each row you can specify only 1 variable, characters after column 71 are ignored. If you need more than 71 characters, you can add any character in column 72 and continue on the next row (the character in column 72 is ignored).
8. THE EQQTROUT DD card must point to a data set or be dummy, but it cannot be removed from the daily plan JCL. In particular if a CP extend or replan job is submitted with the EQQTROUT DD deleted or commented out, the DRTOP/DNTOP JOBSTEP can end with RC08, even if a new plan is created and taken over, because EQQTROUT could not be opened. Also the data set pointed by the EQQTROUT DD must be allocated by using the DCB values provided in the allocation JCL sample: RECFM=VB,LRECL=32756,BLKSIZE=32760 otherwise the contents of the data set will be unreadable.
9. Data sets EQQOUCEV and EQQOUCKP are required if you activate the Output collector started task.

Step 11. Defining the initialization statements

The initialization statements that you need to define depend on the functions of HCL Workload Automation for Z that you want to use. For details about how to define initialization statements, see *Customization and Tuning*.

When HCL Workload Automation for Z starts running, it reads the parameter library to determine initialization options and parameters. The parameter library is specified by the PARM parameter of the EXEC statement for the tracker and controller started-task procedures, and by the EQQPARM DD statement for all the other started tasks and EQQBATCH jobs. If the PARM parameter is not specified, the default member name STDPARMS is used; if the name does not exist, message EQQZ010E is issued.

Step 12. Setting up the ISPF environment

Perform this task if you are installing the scheduler dialogs.

Because HCL Workload Automation for Z dialogs run under ISPF, you must set up an ISPF environment. If you are not familiar with ISPF dialogs, see *ISPF Guide and Reference* and *ISPF Examples*.

To set up your ISPF environment, perform the steps described in the following topics:

1. Setting up the CLIST library on page 146
2. Setting up the ISPF tables on page 147
3. Allocating dialog data sets to your TSO session on page 150
4. Invoking the HCL Workload Automation for Z dialog on page 151

Setting up the CLIST library

When you ran the SMP/E apply job, the scheduler CLIST library was copied to a data set allocated to DD name SEQQCLIB. Allocate this data set to the SYSPROC DD name of the TSO logon procedure JCL. This library includes the EQQXSUBC

CLIST, which is used by the HCL Workload Automation for Z dialog when a user requests an HCL Workload Automation for Z background batch job to be submitted.

For the online EQQAUDIT to work, either copy EQQAUDNS into a library that is part of the TSO SYSPROC concatenation or add the batch-job skeleton library, which is created by EQQJOBS, into the SYSPROC concatenation.

Setting up the ISPF tables

Allocate to the ISPF table library (ISPTLIB) the following tables located in the SEQQTBL0 library:

EQQACMDS

ISPF command table

EQQAEDIT

Default ISPF edit profile

EQQELDEF

Default ended-in-error-list layouts

EQQEVERT

Ended-in-error-list variable-entity read table

EQQLUDEF

Default dialog connect table

EQQRLDEF

Default ready-list layouts

EQQXVART

Dialog field definitions

If you use the ISPF command table EQQACMDS, invoke HCL Workload Automation for Z as a separate ISPF application with the name EQQA. Invoking the HCL Workload Automation for Z dialog on page 151 describes this in more detail. If you want to use a different ISPF application name, for example EQQB, create a command table with the name EQQBCMDS.

The customization of the ISPF Dialog is affected and depends on the ISPF application names. This makes necessary that you create copies of the EQQACMDS and EQQAEDIT members of SEQQBTL0 for each ISPF application and locate these copies in ISPTLIB. For example, for the ISPF application names EQQX and EQQY you need to create the ISPTLIB members EQQXCMDS, EQQYCMDS, EQQXEDIT, and EQQYEDIT.

If necessary, you can modify or create an ISPF command table, using ISPF/PDF option 3.9. Note that ISPF/PDF option 3.9 writes the created or modified table to the data set allocated to the ISPTABL.

Setting up the default dialog-controller connection table

Table EQQLUDEF contains values used when establishing the connection between the scheduler dialog user and the controller. These are default values set initially for your installation by the system programmer. Individual users can then modify the values to suit their requirements. Modify the table, adding the following information:

- The names of the controllers in your installation
- When a controller is accessed remotely, the combination of the controller name and the LU name of a server set up to communicate with it
- The set of dialog–controller connections that are to be available to all dialog users

When a user opens the scheduler dialog 0.1, the scheduler first tries to read the connection table EQQALTCP in the ISPF profile library ISPPROF. The connection table name begins with the NEWAPPL ID specified when invoking the scheduler dialog. For example, if the ISPF application name is EQQB, the connection table name is EQQBLTCP. If you used a different ISPF application name *xxxx*, the connection table name is *xxxxLTCP* (if the application name is shorter than four characters, it is filled with *x* up to length 4). If it cannot find the table, it reads the default connection table EQQLUDEF from the ISPTLIB allocation.

When a user modifies the connection table (through the scheduler dialog option 0.1), the changes are written to the EQQALTCP (or *xxxxLTCP*) table of ISPPROF.

To change the distributed EQQLUDEF table:

1. Choose the scheduler dialog option 0.1.
2. Set up the dialog-controller connections for the installation.
3. Copy the connection table EQQALTCP (or *xxxxLTCP*) from your ISPF profile library to the scheduler table library allocated to ISPTLIB, renaming the copy to the default connection table name EQQLUDEF.
4. Optionally, to change the default subsystem name value shown on the EQQOPCAP panel, you need to change the default setting of the `&xopcnm` variable (currently set to OPCC) in the EQQXINIP dummy panel. You must do this before the 0.1 dialog option is selected for the first time, so that your customized value is used when the EQQLUDEF profile is read and the EQQxLTCP profile is created upon the first access to dialog option 0.1.

You can access and work with different controllers from the same TSO session, using `ISPF SPLIT` to start different HCL Workload Automation for Z instances with different ISPF application names. In this case you might want to add more than one option to invoke HCL Workload Automation for Z from the ISPF master application menu, as in the following example:

```
BODY
 .
 .
 .
     1 ....... - ............
     2 ....... - ............
     . ....... - ............
     OA OPC - Operations Planning and Control A <===
     OB OPC - Operations Planning and Control B <===
     . ....... - ............
  PROC
 .
```

```
.
.
    1, ....
    2, ....
    ., ....
    OA, 'PANEL(EQQOPCAP) NEWAPPL(EQQA)
    OB, 'PANEL(EQQOPCAP) NEWAPPL(EQQB)
.
.
.
 END
```

Note: Because the value of the ISPF variable &XOPCNM. (displayed in the EQQOPCAP dialog as "You are communicating with *xxxx*") and the default controller selected in the 0.1 dialog (EQQXLUSL) are stored, respectively, in members xxxxPROF and xxxxLOUT, make sure that any changes you make to these ISPF profile members are made consistently. For example, if you modify or delete xxxxPROF, you must also modify or delete xxxLOUT.

Setting up list tables and graphical attribute tables

The ISPF tables for list layouts, EQQRLDEF and EQQELDEF, are the default tables displayed for all HCL Workload Automation for Z dialog users in your installation. They can be modified to suit an individual user's requirements or you can create new defaults for all users in your installation. Modified tables are stored in the user's ISPF profile library under another member name. *HCL Workload Automation for Z: Customization and Tuning* describes how to modify the default tables for your installation.

GDDM® default values are used for graphical attributes. The defaults can be modified to suit the requirements of an individual user or you can create default values for all users. Modified defaults are stored in the EQQAXGRC member of the ISPF profile data set.

When setting up these tables for dialog users, keep the following points in mind:

- When a user requests a graphical display using the GRAPH command, HCL Workload Automation for Z first searches through the ISPPROF library for the EQQAXGRC ISPF table. If it cannot find the table there, the product searches the ISPTLIB library for the table.
- When a user modifies the graphical display attributes (using the ATTR command from within an HCL Workload Automation for Z dialog), the EQQAXGRC ISPF table is written to the ISPPROF library.
- When a user displays an ended-in-error list, HCL Workload Automation for Z first searches for the layout in the EQQELOUT table on ISPPROF. If it cannot find the layout there, the product uses the layout from the EQQELDEF table on ISPTLIB.
- When a user modifies an ended-in-error list layout, the changes are written to the EQQELOUT table.
- When a user displays a ready list, HCL Workload Automation for Z first searches for the layout in the EQQRLOUT table of ISPPROF. If it cannot find the layout there, the product uses the layout from the EQQRLDEF table on ISPTLIB.
- When a user modifies a ready list layout, the changes are written to the EQQRLOUT table.

Allocating dialog data sets to your TSO session

Table 29: ISPF and HCL Workload Automation for Z dialog data sets on page 150 describes the ISPF and HCL Workload Automation for Z data sets that you must allocate to the TSO session to run the HCL Workload Automation for Z dialog.

Table 29. ISPF and HCL Workload Automation for Z dialog data sets

DD Name	HCL Workload Automation for Z use	Created by
SYSPROC	CLIST library	SMP/E run (SEQQCLIB)
ISPPROF	User-session defaults, read/write tables	Your existing ISPPROF data set
ISPPLIB	Panel library	SMP/E run (SEQQP*xxx*, SEQQG*xxx*)
ISPMLIB	Message library	SMP/E run (SEQQM*xxx*)
ISPSLIB	Skeleton JCL library	EQQJOBS option 2
ISPTLIB	Read tables (default)	SMP/E run (SEQQTBL0)
EQQMLIB	Message library	SMP/E run (SEQQM*xxx*)
EQQM LOG	Message log	TSO logon procedure
EQQTMPL	Advanced ISPF panel templates	SMP/E run (SEQQLxxx)

Note:

1. The *xxx* suffix represents the national language version supplied with your distribution media.
2. If you did not install the HCL Workload Automation for Z load modules in a library defined in the LNKLST*nn* member of SYS1.PARMLIB, also allocate the load-module library to either the STEPLIB or ISPLLIB DD statements. Except for the EQQMINO2 module, the HCL Workload Automation for Z dialog modules need not run APF-authorized. So if EQQMINO2 is not in the LNKLST*nn* concatenation, you must copy it to another library so that it can be loaded APF-authorized.The product dialog loads EQQMINO2 through IKJEFTSR, therefore you cannot use LIBDEF to add the library containing EQQMINO2 to your STEPLIB or ISPLLIB concatenations.
3. Consider allocating EQQDMSG and EQQDUMP to the TSO session for diagnostic purposes.
4. Ensure that the library containing HCL Workload Automation for Z batch job skeletons, generated by EQQJOBS, is allocated to the ISPSLIB DD statement.
5. You need the EQQMLIB library to run the HCL Workload Automation for Z TSO commands or to use a TCP/IP connected dialog server..
6. The EQQMLOG data set must be allocated to the TSO session of ISPF dialog users to ensure catching all the AUDIT related error messages when the interactive invocation of the audit is used.

7. For the online EQQAUDIT to work, either copy EQQAUDNS into a library that is part of the TSO SYSPROC concatenation or add the batch-job skeleton library, which is created by EQQJOBS, into the SYSPROC concatenation.
8. EQQTMPL identifies the libraries where the Advanced ISPF panel templates are loaded. The templates are the predefined layouts available for the advanced ISPF panels.

   More views are provided for the same panel; for example, for the EQQMOPRV panel (list of operations in the plan), the templates provided are:

View	Description
EQQMOPRT	Compact
EQQMOPLT	Full
EQQMOPJT	Job Detail

Invoking the HCL Workload Automation for Z dialog

You can invoke the HCL Workload Automation for Z dialog in the following ways:

- Using the EQQOPCAC sample CLIST on page 151
- Modifying an existing ISPF selection menu on page 151
- Selecting the main menu directly from TSO on page 152
- Using the ISPF select service on page 152

To switch to the advanced style, see Switching to the advanced style for ISPF panels on page 152.

Using the EQQOPCAC sample CLIST

You can invoke the HCL Workload Automation for Z dialog by using the sample CLIST EQQOPCAC. When you run the sample CLIST in TSO READY mode, EQQOPCAC allocates the dialog data sets and invokes ISPF with the initial master panel EQQ@MSTR. The EQQ@MSTR panel, which is in the HCL Workload Automation for Z panel library, lets you select the applications ISPF/PDF or HCL Workload Automation for Z.

Modifying an existing ISPF selection menu

You can invoke the HCL Workload Automation for Z dialog by including HCL Workload Automation for Z as an option on your existing ISPF master application menu, or on any other selection menu. The following example shows how to do this. The statements that you insert are marked on the right with an arrow (`<====`).

```
ISPF-selection-menu modification for HCL Workload Automation for Z
)BODY
⋮
```

```
    1  ....... - ............
    2  ....... - ............
    .  ....... - ............
    O  OPC - Operations Planning and Control <====
    .  ....... - ............
)PROC
⋮REQCLEANUP - Created by ActiveSystems 12/14/99 Entity not
defined.
 = TRANS(TRUNC(REQCLEANUP - Created by ActiveSystems 12/14/99 Entity not
defined.,'.')
    1 , ....
    2 , ....
    . , ....
    O , 'PANEL(EQQOPCAP) NEWAPPL(EQQA)' <====
    . , ....
⋮
)END
```

Before you can invoke the HCL Workload Automation for Z dialog, allocate the data sets. You can allocate these data sets through the TSO logon procedure, or by running a CLIST after TSO logon.

Although you can use any name that follows the guidelines already established at your installation, the sample ISPF command table, EQQACMDS, is valid only if you use the ISPF application name EQQA. If you change the application name on the ISPSTART command, remember to create the corresponding ISPF command table in the table library.

Selecting the main menu directly from TSO

You can invoke the HCL Workload Automation for Z dialog by selecting the main menu directly from TSO. You do this from TSO by entering this TSO command:

```
/*Invoking the HCL Workload Automation for Z dialog directly from TSO*/
ISPSTART PANEL(EQQOPCAP) NEWAPPL(EQQA)
```

Using this method to invoke the dialog means that the main menu, panel EQQOPCAP, is the first ISPF panel displayed. If you enter the ISPF command SPLIT, EQQOPCAP is displayed on the alternate screen. With this method, you cannot use ISPF/PDF and HCL Workload Automation for Z dialogs at the same time. This method is therefore suitable for users who require *only* HCL Workload Automation for Z.

Using the ISPF select service

You can invoke the HCL Workload Automation for Z dialog by using the SELECT command from a CLIST or from a program. See your ISPF publications to review these procedures.

Switching to the advanced style for ISPF panels

To use the advanced style for ISPF panels, you need to specify Y in the 0.8 option, SETTING PANEL STYLE. The advanced ISPF panels enable you to get a quick, at-a-glance scrollable view of the AD and CP operations, with color-coded fields that represent application and operation status, as well as the addition of an Action menu from where you can select administrative tasks to perform. They are provided for the AD application to enable you to list and browse a single AD and also for the CP operation to list and browse a single operation in the plan. All of the commands available for an operation in the current plan are concentrated in the new operation list panel (EQQSOPRV, EQQMOPRV).

Step 13. Using XCF for communication

Include this task when installing a tracker, Z controller, standby Z controller, or Data Store that will use XCF for communication.

To use the cross-system coupling facility (XCF) for communication between HCL Workload Automation for Z systems, you must:

- Ensure that XCF startup options are suitable for your HCL Workload Automation for Z configuration
- Include the necessary initialization-statement options for each HCL Workload Automation for Z started task.

XCF groups

An HCL Workload Automation for Z XCF system consists of one Z controller and one or more trackers defined as members in the XCF group. You can include one or more standby Z controllers in the group. If you want to connect the Data Store to the controller via XCF, you need to define a specific XCF group for them, different to the one defined to connect the controller to the z/OS® tracker.You can also specify more than one HCL Workload Automation for Z group in a sysplex. For example, you might want to have a test and production HCL Workload Automation for Z group in your sysplex.

HCL Workload Automation for Z supports these sysplex configurations:

MULTISYSTEM

XCF services are available to HCL Workload Automation for Z started tasks residing on different z/OS systems.

MONOPLEX

XCF services are available only to HCL Workload Automation for Z started tasks residing on a single z/OS system.

Note: Because HCL Workload Automation for Z uses XCF signaling services, group services, and status monitoring services with permanent status recording, a *couple* data set is required. HCL Workload Automation for Z does not support a *local sysplex*.

With XCF communication links, the controller can submit workload and control information to trackers that use XCF signaling services. The trackers use XCF services to transmit events to the Z controller. HCL Workload Automation for Z systems are either ACTIVE, FAILED, or NOT-DEFINED for the HCL Workload Automation for Z XCF complex.

Each active member tracks the state of all other members in the group. If an HCL Workload Automation for Z group member becomes active, stops, or terminates abnormally, the other active members are notified. This list describes the actions taken by each started task in the group:

Z controller

When the Z controller detects that a tracker member changes to failed state, it stops sending work to the tracker. When it detects that a tracker has become active, it sends work to the tracker system and instructs the tracker to start transmitting event information.

Standby

When a standby Z controller that is enabled for takeover detects that the Z controller has changed to failed state, it attempts to become the new Z controller. If there is more than one standby Z controller in the group, the first one to detect failure of the Z controller attempts to take over the Z controller functions.

tracker

When a tracker member detects that the Z controller or standby Z controller has failed, it stops sending event information. The tracker member continues to collect events and writes them to the event data set. When the Z controller or standby Z controller becomes active again it informs the tracker that it is ready to receive events.

XCF runtime options

You specify XCF runtime options in the COUPLE*nn* member of SYS1.PARMLIB and change them using SETXCF operator commands. For a description about how to change the options in the COUPLE*nn* member, see Updating XCF initialization options on page 98.

Initialization statements used for XCF

HCL Workload Automation for Z started tasks use the following initialization statements for XCF connections between controller and tracker:

XCFOPTS

Identifies the XCF group and member name for the started task. Include XCFOPTS for each started task that should join an XCF group.

ROUTOPTS

Identifies all XCF destinations to the Z controller or standby Z controller. Specify ROUTOPTS for each Z controller and standby Z controller.

TRROPTS

Identifies the Z controller for a tracker. TRROPTS is required for each tracker on a controlled system. On a controlling system, TRROPTS is not required if the tracker and the Z controller are started in the same address space.

HCL Workload Automation for Z started tasks use these initialization statements for XCF for controller/Data Store connections:

CTLMEM

Defines the XCF member name identifying the controller in the XCF connection between controller and Data Store.

DSTGROUP

It defines the XCF group name identifying the Data Store in the XCF connection with the controller.

DSTMEM

XCF member name, identifying the Data Store in the XCF connection between controller and Data Store.

DSTOPTS

Defines the runtime options for the Data Store.

FLOPTS

Defines the options for Fetch Job Log (FL) task.

XCFDEST

It is used by the FL (Fetch Job Log) task to decide from which Data Store the Job Log will be retrieved.

If you did not include these runtime options when you defined the initialization statements, do this now. Step 11. Defining the initialization statements on page 146 and *HCL Workload Automation for Z: Customization and Tuning* describe the initialization statements.

Step 14. Activating the network communication function

Include this task when installing a tracker, Z controller, or standby Z controller that will use NCF for communication.

If you want to use a VTAM® link to connect a tracker to the Z controller, activate NCF. The Z controller can then send work to the tracker and receive event information back, using the VTAM® link. To achieve this connection, activate NCF in both the Z controller and the tracker. To do this:

- Add NCF to the VTAM® network definitions.
- Add NCF session parameters.
- Activate network resources.

If you want to connect a controller and Data Store using SNA you need different VTAM® definitions. NCF is involved only in the tracker connection; the equivalent task in the Data Store connection is the FN task.

Adding NCF to the VTAM® network definitions

You must define NCF as a VTAM® application on both the controlling system and each controlled system. Before defining NCF, select names for the NCF applications that are unique within the VTAM® network.

To define NCF as an application to VTAM®:

1. Add the NCF applications to the application node definitions, using APPL statements.
2. Add the application names that NCF is known by, in any partner systems, to the cross-domain resource definitions. Use cross-domain resource (CDRSC) statements to do this.

You must do this for all systems that are linked by NCF.

The application node and the cross-domain resource definitions are stored in the SYS1.VTAMLST data set, or in members of a data set that is in the same concatenation as SYS1.VTAMLST. For a detailed description of defining application program major nodes and cross-domain resources, see *VTAM® Resource Definition Reference*.

The following example illustrates the definitions needed for a cross-domain setup between a Z controller and a tracker.

Note:

1. HCL Workload Automation for Z requires that the application name and the ACBNAME are the same.
2. IS1ZOS1 and IS1ZOS2 are only sample names.

At the *Z controller*:

1. Define the NCF Z controller application. Add a VTAM® APPL statement like this to the application node definitions:

```
Z controller VTAM applications
VBUILD TYPE=APPL
 OPCCONTR APPL   VPACING=10,                                        C
                 ACBNAME=OPCCONTR
```

2. Define the NCF tracker application. Add a definition like this to the cross-domain resource definitions:

```
Z controller VTAM cross-domain resources
VBUILD TYPE=CDRSC
 OPCTRK1  CDRSC  CDRM=IS1ZOS2
```

At the *tracker*:

1. Define the NCF tracker application. Add a VTAM® APPL statement like this to the application node definitions:

```
tracker VTAM applications
VBUILD TYPE=APPL
 OPCTRK1  APPL   ACBNAME=OPCTRK1,                                   C
                 MODETAB=EQQLMTAB,                                  C
                 DLOGMOD=NCFSPARM
```

2. Define the NCF Z controller application. Add a CDRSC statement like this to the cross-domain resource definitions:

```
tracker VTAM cross-domain resources
VBUILD TYPE=CDRSC
 OPCCONTR CDRSC  CDRM=IS1ZOS1
```

IS1ZOS1 and IS1ZOS2 are the cross-domain resource managers for the Z controller and the tracker, respectively.

At the *Datastore*:

1. Define the NCF Datastore application. Add a VTAM® APPL statement like this to the application node definitions:

```
Datastore VTAM applications
  VBUILD TYPE=APPL
   OPCDST1 APPL ACBNAME=OPCDST1,                             C
                MODETAB=EQQLMTAB,                            C
                DLOGMOD=NCFSPARM
```

2. Define the NCF controller application. Add a CDRSC statement like this to the cross-domain resource definitions:

   ```
   Datastore VTAM cross-domain resources
     VBUILD TYPE=CDRSC
      OPCCONTR CDRSC CDRM=IS1ZOS1
   ```

Adding NCF session parameters

You can define the session parameters for NCF either by adding the sample EQQLMTAB logon-mode table or by using your own table. If you use the sample table, assemble and link-edit the EQQLMTAB table into the SYS1.VTAMLIB library concatenation for all trackers where an NCF transmitter application is defined.

Note that the APPL statement that defines an NCF application at a tracker must contain the logon-mode-table information in the MODETAB and DLOGMOD parameters.

The EQQLMTAB member of the SEQQSKL0 library contains this logon table definition plus the JCL necessary to assemble and link-edit the table:

```
EQQLMTAB
//LOGON JOB STATEMENT PARAMETERS
//ASM  EXEC PGM=ASMA90,PARM='OBJ,NODECK'
//SYSLIB  DD DSN=SYS1.MACLIB,DISP=SHR
//        DD DSN=SYS1.SISTMAC1,DISP=SHR
//SYSUT1  DD UNIT=SYSDA,SPACE=(1700,(400,50))
//SYSLIN  DD  DSN=&LOADSET,UNIT=SYSDA,SPACE=(80,(250,50)),
//  DISP=(,PASS)
//SYSPRINT DD SYSOUT=*
//SYSIN  DD *
EQQLMTAB MODETAB
         MODEENT LOGMODE=NCFSPARM,                            C
               FMPROF=X'04',                                  C
               TSPROF=X'04',                                  C
               PRIPROT=X'F3',                                 C
               SECPROT=X'F3',                                 C
               COMPROT=X'0000',                               C
               PSERVIC=X'000000000000000000000000',           C
               RUSIZES=X'8787'
         MODEEND
         END
//LINK  EXEC  PGM=IEWL,PARM='XREF,LIST,LET,CALL'
//SYSPRINT DD SYSOUT=*
//SYSLMOD  DD  DSN=SYS1.VTAMLIB(EQQLMTAB),DISP=SHR
//SYSUT1  DD UNIT=SYSDA,SPACE=(1700,(400,50))
//SYSLIN DD  DSN=&LOADSET,DISP=(OLD,DELETE)
```

If you choose to provide session parameters in another table or entry, modify the APPL definitions for the transmitter applications accordingly. Note that NCF uses an LU-type 0 protocol with a recommended minimum RU-size of 500 bytes. Do *not* specify an RU-size smaller than 32 bytes. NCF does not modify the session parameter specified in the LOGMODE table entry in any way.

For a complete description of logon mode tables and the macros that define them, see *VTAM® Customization*.

COS table

No class of service (COS) table entry is specified for EQQLMTAB in the sample. Specify a COS entry that is valid in your VTAM® environment unless you intend to use the default provided by VTAM®.

The routing you specify in the COS entry should be fast and reliable so that unnecessary delays are not introduced in the HCL Workload Automation for Z remote job-tracking function.

Activating network resources

The VTAM® network must be active when the NCF application is started so that network resources are available for the NCF sessions. All participating NCF-application minor nodes must be activated before the NCF application is started by the tracker.

You activate VTAM® resources by entering the `VARY NET` command or by specifying automatic activation in the VTAM® network-definition procedure used during VTAM® startup. You can activate NCF-application minor nodes and CDRSC minor nodes directly, using the VARY ACT command. You can also activate them indirectly by activating their major nodes. For further information, see *VTAM® Operation*.

Diagnostic data set

If you have not already allocated the EQQDUMP diagnostic data set for the tracker or Z controller, do this now. NCF writes debugging information to this diagnostic data set when internal error conditions are detected. When diagnostic information is logged, the information is normally accompanied by a user abend.

Note: Update the HCL Workload Automation for Z started-task procedure with DD name EQQDUMP if this DD name is not already defined.

Step 15. Using TCP/IP for communication

Include this task when installing a scheduler component that will use TCP/IP for communication.

To use the Transmission Control Protocol/Internet Protocol (TCP/IP) for communication among HCL Workload Automation for Z systems:

- Ensure that TCP/IP protocol is available and the relative started task is started on your z/OS® configuration.
- Include the necessary initialization statement options for each product component.

Initialization statements used for TCP/IP

HCL Workload Automation for Z started tasks use the following initialization statements for connecting the scheduler started tasks through TCP/IP:

ROUTOPTS

To identify all the TCP/IP remote destinations for the controller or standby controller. A ROUTOPTS statement is required for each controller and standby controller.

TRROPTS

To identify the controller for a tracker. A TRROPTS statement is required for each tracker on a controlled system.

FLOPTS

To identify all the TCP/IP data store remote destinations for the controller.

DSTOPTS

To identify the controller for a Data Store.

TCPOPTS

An optional statement to specify the TCP/IP options for the local component. To identify the remote partner, use one of the previous statements.

Step 16. Activating support for the API

Include this task when installing a Z controller, or standby Z controller that you want to communicate with through the HCL Workload Automation for Z API.

HCL Workload Automation for Z uses LU to LU communication to pass data between an ATP and a subsystem through the API. To use API requests GET, PUT, and DELETE, the LU that the ATP sends requests to (the target LU) must be owned by the Z controller. For CREATE requests, if the target LU is not owned by an HCL Workload Automation for Z address space where an event-writer task is started, the ATP must send requests so that the events are broadcast on the target z/OS system. *Tivoli® Workload Automation: Developer's Guide: Driving Tivoli® HCL Workload Automation for Z* and *HCL Workload Automation for Z: Customization and Tuning* describe when a request is broadcast.

To activate support for the API, perform these actions in the order shown:

1. Define VTAM® resources.
2. Update APPC options.
3. Activate HCL Workload Automation for Z support for APPC.

If you are installing a standby Z controller, perform corresponding actions on the standby system.

You might need to refer to one or more of these publications:

- *VTAM® Resource Definition Reference*
- *APPC Management*
- *z/OS Initialization and Tuning Reference*
- *Tivoli® Workload Automation: Developer's Guide: Driving Tivoli® HCL Workload Automation for Z*, which documents the API

The actions described here are based on z/OS systems. If you use a later z/OS release, check for enhancements that might make some actions unnecessary.

Defining VTAM resources

Start by defining the associated VTAM® resources:

- Defining a local LU on page 160
- Defining logon modes on page 160
- Defining cross-domain resources on page 161

Then follow the instructions provided in Updating APPC options on page 162 and Activating support for APPC on page 162.

Defining a local LU

Define a local LU in a member in the SYS1.VTAMLST concatenation on the system where you are installing HCL Workload Automation for Z. This example shows how a VTAM® APPL statement might be defined:

```
Local LU definition
VBUILD TYPE=APPL
 IS4MEOP4 APPL ACBNAME=IS4MEOP4,                                       C
               APPC=YES,                                               C
               AUTOSES=5,                                              C
               DMINWNL=3,                                              C
               DMINWNR=6,                                              C
               DSESLIM=9,                                              C
               MODETAB=APPCMODE,                                       C
               SECACPT=CONV,                                           C
               SRBEXIT=YES,                                            C
               VERIFY=OPTIONAL,                                        C
               VPACING=2
```

The LU is called IS4MEOP4 and uses the logon-mode table APPCMODE.

Before you can establish a session with v, a partner LU must be defined. If a partner TP is run at a different node, ensure that an LU is defined at that node.

The controller subsystem currently has tasks that use APPC. The subsystem is defined as one LU node to APPC and VTAM®.

Defining logon modes

The logon-mode table, which you specify in the LU APPL definition statement, must be in the SYS1.VTAMLIB concatenation. To enable LU 6.2 communication for z/OS, you need the VTAM® logon-mode SNASVCMG. For applications, APPC also requires at least one logon-mode entry other than SNASVCMG. You can create a new logon-mode table or add logon modes to an existing table. The name of the logon-mode table that is used by the LU and the partner LU need not be the same, but both LUs must use the same logon-mode names. That is, the logon modes used by these LUs must appear in each table, and they must have the same names. This example of an uncompiled logon-mode table contains three logon modes:

```
Example logon-mode table
APPCMODE MODETAB
          EJECT
 *----------------------------------------------------------------------*
```

```
* Logmode table entry for resources capable of acting as LU 6.2       *
* devices required for LU management.                                 *
*---------------------------------------------------------------------*
 SNASVCMG MODEENT                                                     C
             LOGMODE=SNASVCMG,                                        C
             FMPROF=X'13',                                            C
             TSPROF=X'07',                                            C
             PRIPROT=X'B0',                                           C
             SECPROT=X'B0',                                           C
             COMPROT=X'D0B1',                                         C
             RUSIZES=X'8585',                                         C
             ENCR=B'0000',                                            C
             PSERVIC=X'060200000000000000000300'
*---------------------------------------------------------------------*
* Logmode table entry for resources capable of acting as LU 6.2       *
* devices for PC target.                                              *
*---------------------------------------------------------------------*
 LU62SYS1 MODEENT                                                     C
             LOGMODE=LU62SYS1,                                        C
             RUSIZES=X'8989',                                         C
             SRCVPAC=X'00',                                           C
             SSNDPAC=X'01'
*---------------------------------------------------------------------*
* Logmode table entry for resources capable of acting as LU 6.2       *
* devices for host target.                                            *
*---------------------------------------------------------------------*
 APPCHOST MODEENT                                                     C
             LOGMODE=APPCHOST,                                        C
             RUSIZES=X'8F8F',                                         C
             SRCVPAC=X'00',                                           C
             SSNDPAC=X'01'
         MODEEND
         END
```

Defining cross-domain resources

If the HCL Workload Automation for Z TP and the partner TP are not running in the same VTAM domain, ensure that their respective LUs can communicate by defining cross-domain resources. In this example, LU name IS1ZOS1 is used for the system where the Z controller is activated, and IS1MVS2 for the system that the partner TP is running on.

On the HCL Workload Automation for Z controlling system:

```
Partner LU cross-domain resources
VBUILD TYPE=CDRSC
  LUZOS2 CDRSC  CDRM=IS1ZOS2
```

On the partner system:

```
&opc LU cross-domain resources
VBUILD TYPE=CDRSC
 LUOPC CDRSC  CDRM=IS1ZOS1
```

Updating APPC options

To associate the HCL Workload Automation for Z scheduler (the subsystem) with the local LU that you defined earlier, modify the APPC options by updating the APPCPM*nn* member of SYS1.PARMLIB. The following example shows an APPCPM*nn* member:

```
APPCPMnn example
LUADD                    /* Add local LU to APPC config.   */
   ACBNAME(IS4MEOP4)      /* Name of LU                     */
   SCHED(EOP4)            /* Scheduler name/OPC subsys name */
   TPDATA(SYS1.APPCTP)    /* Profile data set for this LU    */
   TPLEVEL(SYSTEM)        /* TP level for which LU searches */
```

Ensure that the scheduler name is the same as the HCL Workload Automation for Z subsystem name. In this example, the subsystem name is EOP4. A side information file is not used by HCL Workload Automation for Z. However, the LU must be associated with a TP profile data set; you need not specify a profile for HCL Workload Automation for Z in the data set because HCL Workload Automation for Z does not use TP profiles.

To allocate a TP profile data set, you can run a job such as:

```
//ALTPDSET JOB STATEMENT PARAMETERS
//TPSAMPLE EXEC PGM=IDCAMS
//VOLOUT   DD   DISP=OLD,UNIT=3380,VOL=SER=volser
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
        DEFINE CLUSTER (NAME(SYS1.APPCTP) -
            VOLUMES(volser) -
            INDEXED REUSE -
            SHAREOPTIONS(3 3) -
            RECORDSIZE(3824 7024) -
            KEYS(112 0) -
            RECORDS(300 150)) -
          DATA -
            (NAME(SYS1.APPCTP.DATA)) -
          INDEX -
            (NAME(SYS1.APPCTP.INDEX))
```

TP profile data sets are VSAM KSDS data sets.

Activating support for APPC

When you have defined the necessary VTAM® resources and updated APPC options, you can activate HCL Workload Automation for Z support for APPC. Do this by specifying APPCTASK(YES) on the OPCOPTS statement. Perform this action when you have completed all other actions and before you start to use the HCL Workload Automation for Z API.

Step 17. Activating support for the product dialog and programming interface using the server

Include this task when activating an HCL Workload Automation for Z server. To use the Dynamic Workload Console, see Step 19. Activating support for Dynamic Workload Console on page 167.

The HCL Workload Automation for Z dialogs and programming interface can be used on a z/OS® system other than the system where the controller is running. A server is required, running on the same z/OS® system as the controller.

The dialogs and the programming interface on the remote z/OS® system communicate with the server using APPC or TCP/IP. The EQQXLUSL help panels flow describes how to activate the communication with the server using the dialog.

The APPC communication requires also the VTAM® and APPC definitions that are described in the following sections.

For further information, see the following publications:

- *VTAM® Resource Definition Reference*
- *APPC Management*
- *MVS™ Initialization and Tuning Reference*

See also member EQQVTAMS in the EQQJOBS output library or SEQQSAMP library.

To activate the APPC communication, perform the following steps:

1. The server tasks run on the same system as the controller.
2. On the system where the servers and the controller run, you must define the following LUs:
    - One LU for the server 'without' the BASE keyword, and specifying the server started task as SCHEDULER.
3. On the system where you want to enable TSO users to communicate with HCL Workload Automation for Z via the server interface, you must define one LU with the keywords BASE and SCHED; that is:
    - An APPC MASTER LU 'with' the BASE keyword, and specifying SCHED(ASCH). This LU is not for HCL Workload Automation for Z; it is an APPC requirement that there be a BASE LU with SCHED(ASCH) on every system where APPC is used.
4. When the servers and the APPC address space are all started, you will see messages in the SYSLOG and server EQQMLOGs, stating that communication has been established between the server and APPC. These messages are displayed during the first start and after an IPL.
5. The dialog user then selects option 0.1 and specifies the name of the controller subsystem with which he wants to communicate, and the LUNAME of the server via which that communication is to be routed. For more information on specifying these values, press the PF1 (help) on panel EQQXLUSL.

The HCL Workload Automation for Z dialog code in the TSO logon address space then sends an APPC request which is picked up by the APPC BASE LU on that system and routed to the (server) LU specified in the request. The server then passes the dialog data to the controller across the z/OS® subsystem interface, serving as a local proxy for the dialog user. The controller cannot tell whether it is talking to a local ISPF dialog user, or to a remote user via a server.

Defining VTAM resources for the product dialog and program interface using the server

To use the HCL Workload Automation for Z programming interface or dialog from a remote system, you need to activate the APPC support on the remote system.

Assure that there is a LU defined as default LU for the APPC communication (BASE LU) in the APPCPMnn parmlib member. If none is defined, add it as follows:

The HCL Workload Automation for Z dialog and programming interface use the default APPC support defined on the system on which the functions are used. To activate this support:

1. Define a default VTAM® APPL supporting APPC:

```
    VBUILD TYPE=APPL
APPCOUT APPL APPC=YES
             ACBNAME=APPCOUT
             …
```

2. Update the APPCPMnn member of SYS1.PARMLIB for the default VTAM® APPL defined above:

```
 LUADD                                /* Add local LU to APPC config.  */
ACBNAME(APPCOUT)    /* Name of LU                                      */
SCHED(ASCH)                               /* No scheduler associated   */
BASE                /* default LU for the system    */
TPDATA(SYS1.APPCTP) /* Profile data set for this LU */
TPLEVEL(SYSTEM)     /* TP level for which LU search */
```

3. Add any cross-domain resource definition needed to resolve VTAM® addressing.

Defining VTAM® resources for the server

Start by defining the associated VTAM resources for the server:

- Defining a local LU for the server on page 164
- Defining logon modes for the server on page 165

Then follow the instructions provided in Updating APPC options for the server on page 165 and Defining VTAM resources in a parallel sysplex on page 166.

Defining a local LU for the server

Define a local LU in a member in the SYS1.VTAMLST concatenation on the system where you are installing HCL Workload Automation for Z. This example shows how a VTAM® APPL statement might be defined:

```
Local LU for the server definition
VBUILD TYPE=APPL
 IS4MEOP5 APPL APPC=YES,                                         C
              AUTOSES=5,                                         C
              DMINWNL=3,                                         C
              DMINWNR=6,                                         C
              DSESLIM=20,                                        C
              MODETAB=APPCMODE,                                  C
              SECACPT=ALREADYV,                                  C
              SRBEXIT=YES,                                       C
              VERIFY=OPTIONAL,                                   C
              VPACING=2
```

The LU is called IS4MEOP5 and uses the logon-mode table APPCMODE.

The maximum number of TSO dialog users and PIF programs that can simultaneously access an HCL Workload Automation for Z controller via a single server depends on the DSESLIM parameter of the VTAM® LU for that server. Once the specified number of sessions has been established, all subsequent users and PIF programs that try to use that server will hang until one of the existing sessions ends.

The number of servers required by an installation depends on how extensive PIF applications are used. While it can be sufficient with one server for the dialogs, a number of servers can be required for the PIF applications. PIF applications that are frequently used and with long execution time might need separate servers.

Defining logon modes for the server

The logon-mode table, which you specify in the LU APPL definition statement, must be in the SYS1.VTAMLIB concatenation.

The server support requires logon-mode table entries as specified in the following uncompiled example:

Example

```
APPCDIA logon-mode table for server
*---------------------------------------------------------------------*
* Logmode table entry for the dialogs and the programming interface   *
*---------------------------------------------------------------------*
APPCDIA MODEENT                                                      C
              LOGMODE=APPCDIA,                                       C
              RUSIZE=X'8888',                                        C
              SRCVPAC=X'00',                                         C
              SSNDPAC=X'01',                                         C
          MODEENT                                                    C
  APPCFIF MODEENT                                                    C
              LOGMODE=APPCFIF,                                       C
              RUSIZE=X'8888',                                        C
              SRCVPAC=X'00',                                         C
              SSNDPAC=X'01',                                         C
          MODEENT                                                    C
```

The RUSIZE gives a user size for sending buffer of 2048 bytes and a receiving buffer of 4096 bytes.

Updating APPC options for the server

To associate the server (and controller) with the scheduler that you defined earlier, modify the APPC options by updating the LUADD statement in the APPCPM*nn* member of SYS1.PARMLIB. The following example shows an APPCPM*nn* member:

Example

```
APPCPMnn example
LUADD                    /* Add local LU to APPC config.   */
   ACBNAME(IS4MEOP5)      /* Name of LU                     */
   SCHED(EOP5)            /* Scheduler name/OPC subsys name */
   TPDATA(SYS1.APPCTP)    /* Profile data set for this LU    */
   TPLEVEL(SYSTEM)        /* TP level for which LU searches */
```

Ensure that the scheduler name in LUADD is the same of the scheduler server (in this example is EOP5).

Each server identifies itself to APPC as an APPC scheduler with the same name as the started task name. If you specified the SCHEDULER parameter in the SERVOPTS statement, this name is used instead of the started task name.

Defining VTAM® resources in a parallel sysplex

In an installation with a Parallel Sysplex® where the scheduler can start on any of a number of z/OS® images, each z/OS® image within the parallel sysplex should have the same local LU name for a given server. The same LU name must not exist in any other network interconnected to the parallel sysplex network; identical LU names within network, unique LU names across networks.

For details on the parallel sysplex installation, see Step 13. Using XCF for communication on page 153.

The LU name (the APPL statement name) should be given with a wildcard character, in case the scheduler works in a parallel sysplex and is not set up to run on a specific z/OS® image. The APPL statement will then become a Model Application Program Definition, for the identically named LUs on the z/OS® images where the scheduler might start. The wildcard character should be chosen such that one model definition is set up for the controller and one model definition for each of the servers. The optional ACBNAME parameter must be omitted, the name of the APPL statement is then used as the ACBNAME.

For example, say the scheduler can start on z/OS® images ZOS1 and ZOS2 in a parallel sysplex. The LU name for controller OPCB is IS4MOPCB, and there are three servers to handle the communication to OPCB, OPCBCOM1, OPCBCOM2 and OPCBCOM3, with LU names IS4MSV1B, IS4MSV2B and IS4MSV3B. VTAM® Version 4 Release 3 is available. The following model definitions could then be used (a '?' in the APPL statement name represents a single unspecified character):

```
IS4MOP?B  APPL APPC=YES,...
IS4MS?1B  APPL APPC=YES,...
IS4MS?2B  APPL APPC=YES,...
IS4MS?3B  APPL APPC=YES,...
```

Note that the wildcard character must be chosen such that the no other VTAM® LU name than the intended LU name matches the model definition.

Starting the server

You can start the server by using the z/OS® START command, or you can have the controller start and stop the server automatically. In the latter case, include the servers (srv1, srv2, ...) in the OPCOPTS statement in the HCL Workload Automation for Z parameter library.

A SERVOPTS statement is required in the parameters file. All SERVOPTS paramters can be left out and defaulted.

Step 18. Activating support for end-to-end scheduling with z-centric capabilities

To schedule jobs on HCL Workload Automation distributed z-centric agents, activate the end-to-end scheduling with z-centric capabilities. Perform the following steps:

1. Define the z-centric agent destinations in the ROUTOPTS initialization statement.
2. Customize the connection parameters in the HTTPOPTS initialization statement.

Note: Use this statement to activate or disable the SSL connection protocol . If you want to disable the SSL connection, you can either:

- Specify neither SSLKEYRING nor SSLPORT parameters.
- Specify SSLPORT(0).

3. Configure the Z controller to enable the submission of jobs on z-centric and dynamic agents, as described in Enabling jobs submission on z-centric and dynamic agents.
4. Optionally, activate the output collector on page 69 started task to retrieve job logs and copy them to a JES SYSOUT for processing by an external output management product. Customize the HTTPOPTS (OUTPUTCOLLECTOR and JLOGHDRTEMPL parameters), OPCOPTS (OUTCOL parameters), and OUCOPTS initialization statements.

   For the user ID running the output collector started task, ensure that you defined a HOME directory in the OMVS segment. For details, see HOME directory requirements for started tasks on page 197.

For details about the configuration steps, see *Scheduling End-to-end with z-centric Capabilities*.

Step 19. Activating support for Dynamic Workload Console

Perform this step if you want to use the Dynamic Workload Console to design and run your workload.

Before using the Dynamic Workload Console, you need to install the console, which also includes the installation of the Z connector. The Z connector forms the bridge between the console and HCL Workload Automation for Z.

After having installed the Dynamic Workload Console, you can define a z/OS engine to connect to one HCL Workload Automation for Z. To add more z/OS engine definitions after the installation process, see Defining a z/OS engine in the Z connector on page 261.

The console communicates with the product through the Z connector and the scheduler server by using the TCP/IP protocol. The console needs the server to run as a started task in a separate address space. The server communicates with HCL Workload Automation for Z and passes the data and return codes back to the Z connector.

Perform the following task:

- Install and configure the Dynamic Workload Console as described in the sections included in Installing Dynamic Workload Console and Z connector on page 213.

Dynamic Workload Console considerations

Dynamic Workload Console V9.5 or later uses WebSphere Application Server Liberty Base to handle the initial user verification. In all cases, however, it is necessary to obtain a valid corresponding RACF® user ID to be able to work with the security environment in z/OS®.

Note: You cannot control the port from which the Dynamic Workload Console server started task replies to a request from the Z connector. The response ports are randomly selected.

To optimize the thread handling between Z connector and the scheduler server, you can group console users by RACF® user ID. To define this grouping, associate a list of console users to the same RACF® user ID, by editing the `TWSZOSConnConfig.properties` file located in:

```
DWC_DATA_DIR\usr\servers\dwcServer\resources\properties
```

Where *`DWC_DATA_DIR`* is, by default, `%Program Files%\wa\DWC\` on Windows, and `/opt/wa/DWC` on UNIX.

In `TWSZOSConnConfig.properties` set the last two properties as follows:

```
com.ibm.tws.zconn.usr.mapping.enable=true
com.ibm.tws.zconn.usr.mapping.file=mapping_file_path\mapping_file
```

where *`mapping_file`* is the name of the file that contains the mapping between console user and RACF® user ID, as in the following example:

```
engine=zos1919 user=twsuser1,twsuser2 zosuser=zos1919user1
               user=twsuser3,twsuser4 zosuser=zos1919user2
```

Activating server support for the Dynamic Workload Console

To set up the required server environment, customize the INIT and SERVOPTS initialization statements. For example:

```
SERVOPTS SUBSYS (OPCX)
         USERMAP (USERS)
         PROTOCOL (TCP)
         PORTNUMBER (425)
         CODEPAGE (IBM-037)
INIT     CALENDAR (DEFAULT)
```

For more information about the INIT and SERVOPTS statements, see *HCL Workload Automation for Z: Customization and Tuning*.

You can start the server by using the z/OS® START command, or you can have the controller start and stop the server automatically. In the latter case, include the servers (srv1, srv2, ...) on the OPCOPTS statement in the HCL Workload Automation for Z parameter library. The server with TCP/IP support requires access to the C language runtime library (either as STEPLIB or as LINKLIST). If you have multiple TCP/IP stacks, or a TCP/IP started task with a name different from `TCPIP`, then a SYSTCPD DD card is required pointing to a TCP/IP data set containing the TCPIPJOBNAME parameter.

You are always required to define OMVS segments for server started tasks and ensure that a HOME directory is defined in the OMVS segment. For details, see HOME directory requirements for started tasks on page 197.

Step 20. Activating support for the Java™ utilities

This step describes the required actions to use one of the following features (for detailed information about the features, see *Managing the Workload*):

- Dynamic Workload Console reporting
- Event-driven workload automation for data set triggering, with centralized deploy process
- License computation for submitting jobs on z-centric and dynamic agents

As installation actions, perform the following steps:

1. Install IBM® Semeru Runtime Certified Edition (formerly known as IBM® 64-bit SDK for z/OS®, Java Technology Edition). For information about how to install it, see the related IBM documentation.
2. Copy the JZOS Java™ Launcher load module (JVMLDM*nn*) from the JAVA_HOME directory to the SYS1.SIEALNKE system data set. For details about customizing the JZOS Java™ Launcher, see see the related IBM documentation.
3. Make sure that you applied the appropriate FMID (for details about FMIDs, see the Program Directory).
4. Run EQQJOBS with the option to enable JAVA utilities, to create the EQQPCS08 sample JCL.
5. Customize EQQPCS08 and submit it.
6. Define the TRGOPT initialization statement in a member of the EQQPARM library.
7. Define the event rule in XML format. You can use a partitioned data set member to be used as input for the following step. The SEQQSAMP library contains EQQXML01 member as sample of event rule definition.
8. Select option 1.7.3 from the main menu, edit and submit the job to produce the configuration files.

Chapter 5. Verifying your installation

Perform this task for a tracker Z controller or standby Z controller.

When you have installed a tracker, Z controller, standby Z controller, or server, start it and perform initial verification procedures. To completely verify HCL Workload Automation for Z, start all the address spaces in your configuration, and create database entries, a long-term plan, and a current plan. This is required to verify job submission and connections between systems, and requires some knowledge of the product. Therefore, verification is divided into two parts:

- Initial verification of individual HCL Workload Automation for Z address spaces.
- Verification of your configuration.

You can therefore perform some verification tasks that do not require the knowledge of detailed aspects of HCL Workload Automation for Z. When you are more familiar with the product components and functions, you can perform further testing.

The following topics are described:

If you are installing a tracker and Z controller in the same address space, review the initial verification procedures for both a tracker and a Z controller.

Verifying installation of a tracker

When you have completed the installation tasks for a tracker, perform its initial verification. Because connections and submission of work cannot be verified in isolation, you can perform further verification of the tracker after you have installed the controlling system, established connections between HCL Workload Automation for Z systems, and created a current plan. For a detailed description of these verification tasks, see Verifying configuration on page 182.

To initially verify the tracker, perform the following tasks:

1. Follow the appropriate procedures for the HCL Workload Automation for Z subsystem that you are installing.
2. Ensure that you have completed all the necessary installation tasks.
3. Start the tracker and check the message log (EQQMLOG).
4. Verify that tracking events are created in the event data set (EQQEVDS).
5. Perform problem determination for tracking events if events are missing from the event data set.

For TCP/IP connections only, ensure that a valid current plan exists before verifying the tracker.

Ensuring that all installation tasks are complete

Ensure that you have performed all the installation tasks that are needed for your HCL Workload Automation for Z service. That is, you should have:

- Followed the appropriate procedures for the HCL Workload Automation for Z subsystem that you are installing
- Installed the required JES and SMF exits, and verified that they are active
- Created a JCL procedure for the tracker
- Allocated required data sets
- Given the security access for the subsystem to access the data sets
- Specified the initialization statements in the parameter library (EQQPARM)
- Included the tracker in the same XCF group as the Z controller, if the tracker uses an XCF connection
- Defined a VTAM® LU name for the tracker and activated the VTAM® resources, if the tracker uses an NCF connection.

Checking the message log (EQQMLOG)

Start the tracker.

When the tracker is started, check the message log:

- Check that the return code for all initialization options is 0 (message EQQZ016I).
- Ensure that all required subtasks are active.
  - The data-router and submit tasks are always started. You should see these messages:

    ```
    EQQZ005I SUBTASK DATA ROUTER TASK IS BEING STARTED
    EQQF001I DATA ROUTER TASK INITIALIZATION IS COMPLETE

    EQQZ005I SUBTASK JOB SUBMIT TASK  IS BEING STARTED
    EQQSU01I THE SUBMIT TASK HAS STARTED
    ```
  - Also, verify that the tracker has started an event writer. You should see these messages:

    ```
    EQQZ005I SUBTASK EVENT WRITER IS BEING STARTED
    EQQW065I EVENT WRITER STARTED
    ```
- Examine error messages.

Note: The first time the event writer is started, it formats the event data set. Ignore the SD37 abend code that is issued during the formatting process.

If you see error messages in the message log for an NCF connection, this is because you cannot fully verify an NCF connection until the Z controller is active and a current plan exists. Active tracker-connection messages for XCF connections are written to the Z controller message log when the Z controller is started. If you have specified any of these functions, follow the procedures in Verifying configuration on page 182 when you have completed initial verification procedures.

- Check that your log is complete.

  If your log seems to be incomplete, information might be in a buffer. If you are unsure whether the log is complete, issue a dummy modify command like this: `F ssname,xx`. Message EQQZ049E is written to the log when the command is processed. This message will be the last entry in the log.

Verifying tracking events

The next verification phase is to check that the tracker is collecting tracking-event information and writing it to the event data set (EQQEVDS).

HCL Workload Automation for Z job tracking works correctly only if it receives information about status changes for all jobs or started tasks to be tracked. Job tracking gets this information from SMF and JES exits. These exits gather the necessary information, and an exit record is added to the HCL Workload Automation for Z event-writer queue via ECSA buffers.

The event writer

The event writer removes the event from its queue and creates an event record that is written to an event data set. The event writer also forwards the event if it has been started with an event-reader function. If a separate event reader is used, the event is read from the event data set. In either case, the reader task uses the connection with the Z controller to transfer the event to a queue at the Z controller. The event-manager subtask then processes the event and the current plan is updated.

The event data set

The event data set is needed to even out any difference in the rate that events are being generated and processed, and to prevent events from being lost if the HCL Workload Automation for Z address space or a subtask must be restarted. The first byte in an exit record is A if the event is created on a JES2 system, or B if the event is created on a JES3 system. This byte is found in position 21 of a standard event record, or position 47 of a continuation (type N) event. Bytes 2 and 3 in the exit record define the event type. These event types are generated by HCL Workload Automation for Z for jobs and started tasks:

1

Reader event. A job has entered the JES system.

2

Job-start event. A job has started to execute.

3S

Step-end event. A job step has finished executing.

3J

Job-end event. A job has finished executing.

3P

Job-termination event. A job has been added to the JES output queues.

4

Print event. An output group has been printed.

5

Purge event. All output for a job has been purged from the JES system.

If any of these event types are not being created in the event data set (EQQEVDS), a problem must be corrected before HCL Workload Automation for Z is started in production mode.

Note:

1. The creation of step-end events (3S) depends on the value you specify in the STEPEVENTS keyword of the EWTROPTS statement. The default is to create a step-end event only for abending steps in a job or started task.
2. The creation of print events depends on the value you specify in the PRINTEVENTS keyword of the EWTROPTS statement. By default, print events are created.

Perform these actions to verify that events are being created on your system:

1. Run a job:
   a. Submit a job like the following, ensuring that the output is written to a non-held output class:

      Test job

      ```
      //VERIFY1  JOB   STATEMENT PARAMETERS




      //VERIFY   EXEC  PGM=IEBGENER
      //*
      //SYSPRINT DD DUMMY
      //SYSUT2   DD SYSOUT=A
      //SYSIN    DD DUMMY
      //SYSUT1   DD *
          SAMPLE TEST OUTPUT STATEMENT 1
      //*
      ```
   b. Verify that the job has executed, printed, and purged.
   c. Browse the EQQEVDS data set using the ISPF/PDF browse facility. You should find these events on the event data set:
      - Type 1 event
      - Type 2 event
      - Type 3J event
      - Type 3P event
      - Type 4 event
      - Type 5 event.

      The events are prefixed with A for JES2 or B for JES3. You might also find type 3S as events, depending on the value specified on the STEPEVENTS keyword of the EWTROPTS initialization statement.
2. Repeat step 1 for a started task.

Performing problem determination for tracking events

Problem determination depends on which event is missing and whether the events are created on a JES2 or JES3 system. In Table 30: Problem determination for missing tracking events on page 174, the first column refers to the event type that is missing, and the second column tells you what action to perform. Events created on a JES2 system are prefixed with A, and events created on a JES3 system with B. The first entry in the table applies when all event types are missing (when the event data set does not contain any tracking events).

Table 30. Problem determination for missing tracking events

Type	Problem determination actions
All	1. Verify in the EQQMLOG data set that the event writer has started successfully. 2. Verify that the definition of the EQQEVDS ddname in the HCL Workload Automation for Z started-task procedure is correct (that is, events are written to the correct data set). 3. Verify that the required exits have been installed. 4. Verify that the IEFSSN*nn* member of SYS1.PARMLIB has been updated correctly, and that an IPL of the z/OS system has been performed since the update.
A1	If both A3P and A5 events are also missing: 1. Verify that the HCL Workload Automation for Z version of the JES2 exits 7 and 51 routines have been correctly installed. Use the JES commands `$T EXIT(7)` and `$T EXIT(51)` or `$DMODULE(OPCAXIT7)` and `$DMODULE(TWSXIT51)`. 2. Verify that the JES2 initialization data set contains a LOAD statement and an EXIT7 statement for the HCL Workload Automation for Z version of JES2 exit 7 (OPCAXIT7). Verify also that the JES2 initialization data set contains a LOAD statement and an EXIT51 statement for the version of JES2 exit 51 (TWSXIT51) 3. Verify that the exit has been added to a load module library reachable by JES2 and that JES2 has been restarted since this was done.
B1	1. Verify that the HCL Workload Automation for Z version of the JES3 exit IATUX29 routine has been correctly installed. 2. Verify that the exit has been added to a load-module library that JES3 can access. 3. Verify that JES3 has been restarted.
A2/B2	1. Verify that the job for which no type 2 event was created has started to execute. A type 2 event will not be created for a job that is flushed from the system because of JCL errors. 2. Verify that the IEFUJI exit has been correctly installed: a. Verify that the SMF parameter member SMFPRM*nn* in the SYS1.PARMLIB data set specifies that the IEFUJI exit should be called. b. Verify that the IEFUJI exit has not been disabled by an operator command.

Table 30. Problem determination for missing tracking events (continued)

Type	Problem determination actions
	c. Verify that the correct version of IEFUJI is active. If SYS1.PARMLIB defines LPALIB as a concatenation of several libraries, z/OS uses the first IEFUJI module found. d. Verify that the library containing this module was updated by the HCL Workload Automation for Z version of IEFUJI and that z/OS has been IPLed since the change was made.
A3S/B3S	If type 3J events are also missing: 1. Verify that the IEFACTRT exit has been correctly installed. 2. Verify that the SMF parameter member SMFPRM*nn* in the SYS1.PARMLIB data set specifies that the IEFACTRT exit should be called. 3. Verify that the IEFACTRT exit has not been disabled by an operator command. 4. Verify that the correct version of IEFACTRT is active. If SYS1.PARMLIB defines LPALIB as a concatenation of several libraries, z/OS uses the first IEFACTRT module found. 5. Verify that this library was updated by the HCL Workload Automation for Z version of IEFACTRT and that z/OS has been IPLed since the change was made. If type 3J events are not missing, verify, in the EQQMLOG data set, that the event writer has been requested to generate step-end events. Step-end events are created only if the EWTROPTS statement specifies STEPEVENTS(ALL) or STEPEVENTS(NZERO) or if the job step abended.
A3J/B3J	If type 3S events are also missing, follow the procedures described for type 3S events.
A3P	If A1 events are also missing, follow the procedures described for A1 events.
B3P	1. Verify that the HCL Workload Automation for Z version of the JES3 exit IATUX19 routine has been correctly installed. 2. Verify that the exit has been added to a load-module library that JES3 can access. 3. Verify that JES3 has been restarted.
A4/B4	1. If you have specified PRINTEVENTS(NO) on the EWTROPTS initialization statement, no type 4 events are created. 2. Verify that JES has printed the job for which no type 4 event was created. Type 4 events will not be created for a job that creates only held SYSOUT data sets. 3. Verify that the IEFU83 exit has been correctly installed: a. Verify that the SMF parameter member SMFPRM*nn* in the SYS1.PARMLIB data set specifies that the IEFU83 exit should be called. b. Verify that the IEFU83 exit has not been disabled by an operator command. c. Verify that the correct version of IEFU83 is active. If SYS1.PARMLIB defines LPALIB as a concatenation of several libraries, z/OS uses the first IEFU83 module found.

Table 30. Problem determination for missing tracking events (continued)

Type	Problem determination actions
	d. Verify that the library containing this module was updated by the HCL Workload Automation for Z version of IEFU83 and that z/OS has been IPLed since the change was made. e. For JES2 users (A4 event), ensure that you have not specified TYPE6=NO on the JOBCLASS and STCCLASS statements of the JES2 initialization parameters.
A5	1. Verify that JES2 has purged the job for which no A5 event was created. 2. Ensure that you have not specified TYPE26=NO on the JOBCLASS and STCCLASS statements of the JES2 initialization parameters. 3. If A1 events are also missing, follow the procedures described for A1 events.
B5	1. Verify that JES3 has purged the job for which no B5 event was created. 2. If B4 events are also missing, follow the procedures described for B4 events.

Verifying installation of a controller and dialogs

When you have completed the installation tasks for a Z controller, perform its initial verification. Because connections and submission of work cannot be verified in isolation, you can perform further verification of the Z controller after you have installed the controlling system, established connections between HCL Workload Automation for Z systems, and created a current plan. For a detailed description of these verification tasks, see Verifying configuration on page 182.

To initially verify the Z controller, perform the following steps:

1. Ensure that you have completed the installation tasks.
2. Start the Z controller, and check the message log (EQQMLOG).
3. Check that you can access HCL Workload Automation for Z data via the dialogs, and that authority checking is functioning as required.

If you encounter an error during verification, see Performing problem determination on page 178.

Ensuring that all installation tasks are complete

Make sure that you have:

- Created a started-task procedure for the Z controller
- Allocated data sets
- Given security authority to the started task to access its data sets
- Specified the initialization statements in the parameter library (EQQPARM)
- Included the Z controller in an XCF group, if it uses an XCF connection
- Defined a VTAM® application ID for the Z controller and activated the VTAM® resources, if it uses an NCF connection

- Updated SYS1.PARMLIB and defined VTAM® resources, if users communicate with the Z controller through the HCL Workload Automation for Z API or if the HCL Workload Automation for Z server is used.
- Set up the ISPF environment for HCL Workload Automation for Z dialog users.

Checking the message log (EQQMLOG)

Start the Z controller (for detailed information about starting and stopping HCL Workload Automation for Z, see *Managing the Workload*).

When the Z controller is started, check the message log:

- Ensure that the return code for all initialization options is 0 (message EQQZ016I).
- Check that all required subtasks are active.

  Look for the following messages when the Z controller is started:

  Active general-service messages

  ```
  EQQZ005I SUBTASK GENERAL SERVICE IS BEING STARTED
  EQQZ085I SUBTASK GS EXECUTOR 01 IS BEING STARTED
  EQQG001I SUBTASK GS EXECUTOR 01 HAS STARTED
  EQQG001I SUBTASK GENERAL SERVICE HAS STARTED
  ```

  Note: The preceding messages, EQQZ085I and EQQG001I, are repeated for each general service executor that is started. The number of executors started depends on the value you specified on the GSTASK keyword of the OPCOPTS initialization statement. The default is to start all five executors.

  Active data-router-task messages

  ```
  EQQZ005I OPC SUBTASK DATA ROUTER TASK IS BEING STARTED
  EQQF001I DATA ROUTER TASK INITIALIZATION IS COMPLETE
  ```

  When you start a Z controller and no current plan exists, you will still see a number of EQQZ005I messages each indicating that a subtask is being started. But these subtasks will not start until a current plan is created. You will also see this message:

  Current plan message

  ```
  EQQN105W NO VALID CURRENT PLAN EXISTS. CURRENT PLAN VSAM I/O IS NOT
  POSSIBLE
  ```

  If you have specified an event-reader function or NCF connections, these tasks will end if no current plan exists. You can verify the remaining tasks when you have created a current plan and connections can be established. Verifying configuration on page 182 describes these tasks.

- Check that the log is complete.

Note: If your log seems incomplete, information might be in a buffer. If you are unsure whether the log is complete, issue a dummy modify command like `F ssname,xx`. Message EQQZ049E is written to the log when the command is processed. This message will be the last entry in the log.

Checking the server message log

After the controller is started, it starts the servers automatically if you specified the SERVERS parameter in the OPCOPTS statement. Otherwise, you must start them using the z/OS® START command. When the server is started, check the message log:

- Ensure that the return code for all the initialization options is 0 (message EQQZ016I)
- Look for the following messages when the server is started:

Active server messages

```
EQQZ0051 SUBTASK SERVER IS BEING STARTED
EQQPH001 SERVER TASK HAS STARTED
```

Checking dialog functions

Before invoking the HCL Workload Automation for Z dialog, ensure that you have set up the ISPF environment as described in The ISPF environment on page 192. Then invoke the HCL Workload Automation for Z dialog, and select an option from 0 to 10 on the main menu. If a new panel appears, you have established communication with the HCL Workload Automation for Z subsystem. You can further test the dialogs by performing functions, such as creating an application description. For more information on specific dialog functions, see *Managing the Workload*.

If you have used RACF® to protect Z controller resources from unauthorized access, verify that the protection mechanism works as expected.

Performing problem determination

If you encounter problems during the verification of HCL Workload Automation for Z, correct the errors and verify that the problem has been fixed. For more information on problem analysis, see the *Diagnosis Guide and Reference*.

Dialog problems

Various errors can occur when you are running HCL Workload Automation for Z dialogs. These errors cause the terminal alarm to sound and a short message to appear in the upper-right corner of your terminal screen. The message text for errors that cause the terminal alarm to sound usually contains the ALARM=YES flag. If this happens when you are trying to verify that HCL Workload Automation for Z dialogs are correctly installed, press the Help key (usually PF1) in ISPF. ISPF then displays a more complete error message in the long message area on your terminal. The following examples show two dialog error messages. The message number in each example is followed by the long message text and an explanation of the error. The examples highlight two errors. They are related to the dialog interface module, EQQMINOx.

EQQX115

```
EQQX115E TSO Service Facility RC: 20, RSNC: 40
```

The EQQMINOx load module is not installed in a library that can be reached by TSO. EQQMINOx must be present either in the STEPLIB library of the current TSO session or in a library in the current LINKLIB LNKLST*nn* concatenation. If EQQMINOx has been installed in a LINKLIB library, either an LLA refresh process or an IPL is required to make the module accessible by z/OS users.

EQQX120

```
The EQQMINOx program can only be called by an APF-authorized task
```

The EQQMINOx load module must be APF authorized. It must reside in a data set that is defined in SYS1.PARMLIB as being an APF-authorized library. Also, EQQMINOx must be defined to TSO as being an APF-authorized program. Ensure that you have followed the instructions described in Modifying TSO parameters on page 98.

Authority problems

It is easiest to verify that the Z controller has been correctly installed without activating authority checking. Then, when authority checking is activated, some TSO users should no longer be able to do what they could do before. An HCL Workload Automation for Z dialog message should be issued, specifying that they are not authorized to perform a particular dialog function, or that they are not authorized to use any HCL Workload Automation for Z dialog.

If the Z controller authority functions have not been correctly installed, this will usually enable TSO users to use dialog functions that they are not authorized to use. The symptom of this problem is the absence of an expected error message. If this happens, follow this procedure which assumes the security monitor being used is RACF®.

1. Verify the APPL class. If the user should not be able to use any HCL Workload Automation for Z dialog, verify that the APPL class is active and that the Z controller is defined as a resource in the APPL class. Also, verify that the user is not present in the access list to any of these resources and that universal access NONE has been specified. Use the `SETR LIST` command to display active classes, and use the `RLIST` command to display access lists in the APPL resource class.
2. Verify that the name of the HCL Workload Automation for Z RACF® resource class has been defined to the HCL Workload Automation for Z started task in the AUTHDEF statement. You can check this by browsing the Z controller message log (EQQMLOG).
3. Verify the definition of the HCL Workload Automation for Z resource class. Check the source of the RACF® class descriptor table, and compare this with the definition supplied by the ICHRRCDE sample in the SEQQSAMP library (see Sample library (SEQQSAMP) on page 343).
4. Verify fixed resources. If the user should not be able to use a specific dialog, such as the Calendar dialog, verify that the HCL Workload Automation for Z RACF® resource class is active and that CL is defined as a resource in this class. Also, verify that the user is not present in the access list to the CL resource and that universal access NONE has been specified.
5. Verify subresources. If, for example, the user should be able to update only a subset of all applications in the Application Description dialog, but is instead able to update all applications, verify that the SUBRESOURCES keyword

has been correctly specified for the Z controller in the AUTHDEF statement. Also verify that the Z controller has been restarted since the AUTHDEF statement was changed, and that HCL Workload Automation for Z RACF® profiles have been refreshed since the HCL Workload Automation for Z subresource profiles were updated.

Verifying installation of a standby controller

When you have completed the installation tasks for a standby Z controller, perform initial verification. Because connections cannot be verified in isolation, you can perform further verification of the standby Z controller after you have installed the controlling system, established connections between HCL Workload Automation for Z systems, and created a current plan. For a detailed description of these verification tasks, see Verifying configuration on page 182.

To initially verify the standby Z controller, perform these tasks:

1. Ensure that you have completed all the necessary installation tasks.
2. Start the standby Z controller, and check the message log (EQQMLOG).

Ensuring that all installation tasks are complete

Make sure that you have performed the following actions:

- Created a JCL procedure for the standby Z controller
- Given the security authority for the address space to access the same data sets as the Z controller
- Specified the initialization statements in the parameter library (EQQPARM)
- Included the standby Z controller in the same XCF group as the Z controller
- Defined a VTAM® application ID for the standby Z controller, and activated the VTAM® resources, if the standby Z controller uses NCF connections
- Assigned an IP address to the tracker, if the tracker uses a TCP/IP connection.
- Updated SYS1.PARMLIB and defined VTAM® resources, if the HCL Workload Automation for Z API or the HCL Workload Automation for Z server is used.

Checking the message log (EQQMLOG)

Start the standby Z controller and browse the message log to:

- Ensure that the return code for all initialization options is 0 (message EQQZ016I).
- Check that the following message appears:

  Standby Z controller message

  ```
  EQQZ128I SCHEDULER ACTIVE IN STANDBY MODE
  ```

Verifying installation of the Restart and Cleanup function

To verify that the Restart and Cleanup function was installed and configured correctly, perform the following tasks:

- Verify that for each spool a Data Store was installed and started correctly (verify the message log EQQMLOG).
- Verify that the controller was started with the correct parameters (for a sample configuration, see SNA-only connection on page 33).

Checking the message log (EQQMLOG)

After the Z controller has been started, ensure that the following messages appear in the message log (this example shows messages for an SNA connection:

```
02/07 12.11.39 EQQZ015I INIT STATEMENT: RCLOPTS  CLNJOBPX(EQQCL)
02/07 12.11.39 EQQZ015I INIT STATEMENT:          DSTDEST(TWSFDEST)
02/07 12.11.43 EQQPS01I PRE SUBMITTER TASK INITIALIZATION COMPLETE
02/07 12.11.46 EQQFSF1I DATA FILE EQQSDF01 INITIALIZATION COMPLETED
02/07 12.11.46 EQQFSF1I DATA FILE EQQSDF02 INITIALIZATION COMPLETED
02/07 12.11.46 EQQFSF1I DATA FILE EQQSDF03 INITIALIZATION COMPLETED
02/07 12.11.46 EQQFSI1I SECONDARY KEY FILE INITIALIZATION COMPLETED
02/07 12.11.46 EQQFSD5I SYSOUT DATABASE INITIALIZATION COMPLETE
02/07 12.11.46 EQQFL01I JOBLOG FETCH TASK INITIALIZATION COMPLETE
02/07 12.11.46 EQQFSD1I SYSOUT DATABASE ERROR HANDLER TASK STARTED
02/07 12.11.46 EQQFV36I SESSION I9PC33A3-I9PC33Z3 ESTABLISHED
```

Note:

1. There should be an EQQFSF1I message for each EQQSDF*xx* file specified in the startup procedure.
2. There should be an EQQFV36I message for each SNA connection.
3. Verify that the DSTDEST for message EQQZ015I matches the SYSDEST in the Data Store message log.

After the server has been started, ensure that the following messages appear in the message log:

```
02/07 20.16.10 EQQZ015I INIT STATEMENT:         SYSDEST(TWSFDEST)
02/07 20.16.16 EQQFSK1I PRIMARY KEY FILE INITIALIZATION COMPLETED
02/07 20.16.16 EQQFCM2I Data Store COMMAND TASK IS BEING STARTED
02/07 20.16.16 EQQFCC1I Data Store COMMUNICATION TASK INITIALIZATION COMPLETED
02/07 20.16.16 EQQFV01I FN APPLICATION STARTED
02/07 20.16.16 EQQFV24I ACB SUCCESSFULLY OPENED
02/07 20.16.16 EQQFSF1I DATA FILE EQQSDF01 INITIALIZATION COMPLETED
02/07 20.16.16 EQQFV36I SESSION I9PC33Z3-I9PC33A3 ESTABLISHED
02/07 20.16.16 EQQFSF1I DATA FILE EQQSDF02 INITIALIZATION COMPLETED
02/07 20.16.16 EQQFSF1I DATA FILE EQQSDF03 INITIALIZATION COMPLETED
02/07 20.16.16 EQQFSF1I DATA FILE EQQUDF01 INITIALIZATION COMPLETED
02/07 20.16.16 EQQFSF1I DATA FILE EQQUDF02 INITIALIZATION COMPLETED
02/07 20.16.16 EQQFSF1I DATA FILE EQQUDF03 INITIALIZATION COMPLETED
02/07 20.16.16 EQQFSI1I SECONDARY KEY FILE INITIALIZATION COMPLETED
02/07 20.16.16 EQQFSD5I SYSOUT DATABASE INITIALIZATION COMPLETE
02/07 20.16.16 EQQFSD1I SYSOUT DATABASE ERROR HANDLER TASK STARTED
02/07 20.16.16 EQQFCU1I CLEAN UP TASK STARTED
02/07 20.16.16 EQQFSW1I Data Store WRITER TASK INITIALIZATION COMPLETED
02/07 20.16.16 EQQFSW1I Data Store WRITER TASK INITIALIZATION COMPLETED
02/07 20.16.16 EQQFSW1I Data Store WRITER TASK INITIALIZATION COMPLETED
02/07 20.16.16 EQQFJK3I Data Store JESQUEUE TASK INITIALIZATION COMPLETED
02/07 20.16.21 EQQFSR1I Data Store READER TASK INITIALIZATION COMPLETED
```

Note:

1. There should be an EQQFSF1I message for each EQQSDF*xx* file specified in the startup procedure.
2. Verify that the SYSDEST for message EQQZ015I matches the DSTDEST in the controller message log.
3. There should be an EQQFSW1I message for every writer task.
4. There should be an EQQFCC1I message to indicate that the communication completed successfully.

Verifying configuration

When you have installed your HCL Workload Automation for Z controlling system, or when you install a controlled system, review this section to complete the verification of HCL Workload Automation for Z. The following topics are described:

- Creating entries in the databases
- Running HCL Workload Automation for Z batch jobs
- Checking the message logs (EQQMLOG)
- Verifying workload submission
- Verifying takeover by a standby Z controller

Creating entries in the databases

You can completely verify HCL Workload Automation for Z *only* when you have created a current plan. Before creating a CP, you must create entries in the databases and produce a long-term plan. If you are not familiar with HCL Workload Automation for Z, see *Managing the Workload* for information about updating the databases and producing a long-term plan and current plan.

Sample HCL Workload Automation for Z databases come with HCL Workload Automation for Z. They are loaded into the SMP/E target library with ddname SEQQDATA when the tracker software CD is processed. You can load the sample databases by submitting the EQQSAMPI JCL, which is generated by EQQJOBS.

Running batch jobs

When you have created database entries, invoke the LTP dialog and create a long-term plan. Check that the batch job completed successfully, and browse the long-term plan to check that the entries are correct. Next, use the Daily Planning dialog to create a current plan. When this job has ended, browse the current plan to check that the expected application occurrences are present.

You can further test HCL Workload Automation for Z batch jobs by, for example, printing information about the entries you have created in the databases. For a lsit of the HCL Workload Automation for Z batch jobs, see Table 17: Controller skeleton JCL generated by the EQQJOBS dialog on page 79.

Checking the message logs (EQQMLOG)

When you have created a current plan and have started all HCL Workload Automation for Z address spaces in your configuration, check the message log of the Z controller and of all trackers.

Controller message log

Look for the following messages in the message log of the Z controller:

Active normal-mode manager messages

```
EQQZ005I SUBTASK NORMAL MODE MGR  IS BEING STARTED
EQQN013I NOW PROCESSING PARAMETER LIBRARY MEMBER EQQCP1DS
```

Note: Active job-tracking log archiver messages. In the preceding message, the active current plan ddname is either EQQCP1DS or EQQCP2DS.

```
EQQZ005I SUBTASK JT LOG ARCHIVER IS BEING STARTED
EQQN080I THE LOG ARCHIVER TASK HAS STARTED
```

Active job-tracking log archiver messages

Note: The preceding messages, EQQZ085I and EQQG001I, are repeated for each general service executor that is started. The number of executors started depends on the value you specified on the GSTASK keyword of the OPCOPTS initialization statement. The default is to start all five executors.

Active data-router-task messages

```
EQQZ005I SUBTASK DATA ROUTER TASK IS BEING STARTED
EQQF001I DATA ROUTER TASK INITIALIZATION IS COMPLETE
```

If you have specified that APPC support should be started, check that these messages appear:

Active APPC-task messages

```
EQQZ005I SUBTASK APPC TASK IS BEING STARTED
EQQO001I APPC TASK INITIALIZATION IS COMPLETE
```

This message must be issued for the first Z controller or server start after APPC starts; it is issued by APPC to the system log:

APPC scheduler active - system log messages

```
ATB050I LOGICAL UNIT IS4MEOP4 FOR TRANSACTION SCHEDULER EOP4 HAS BEEN
ADDED TO THE APPC CONFIGURATION.
```

If you have specified an event-reader function, check that these messages appear:

Active event-reader messages

```
EQQZ005I SUBTASK EVENT READER IS BEING STARTED
EQQR025I ERDR 01 STARTED
```

The numeric value on message EQQR025I indicates which event reader is started. The same value cannot be specified on more than one ERSEQNO keyword at the same node.

If the Z controller uses XCF connections, the XCF group is activated when the Z controller is started. Several messages can appear in the message log, indicating that a tracker or standby Z controller has started and that it has joined the group. If the Z controller communicates with a tracker using XCF, check for this message to verify the connection:

Active tracker-connection message

```
EQQF007I XCF MEMBER TRACK2 HAS JOINED THE GROUP. THE DESTINATION WILL BE
EQQF007I REPORTED ACTIVE
```

If a standby Z controller is started, check for this message:

Active standby-Z controller-connection message

```
EQQF008I XCF MEMBER CTRSTBY1 HAS JOINED THE GROUP AS STANDBY FOR THE
EQQF008I CONTROLLER
```

If the Z controller uses an NCF connection, check that these messages appear (where NCFCON01 is the VTAM® application ID of the Z controller, and NCFTRK01 is the VTAM® application ID of the tracker):

Active NCF-connection messages

```
EQQZ005I SUBTASK VTAM I/O TASK IS BEING STARTED
EQQV001I NCF APPLICATION STARTED
EQQV024I ACB SUCCESSFULLY OPENED
EQQV036I SESSION NCFCON01-NCFTRK01 ESTABLISHED
```

If the controller uses the Restart and Clean Up functionality check that the following messages appear in the Z controller MLOG:

```
EQQZ005I SUBTASK FL TASK IS BEING STARTED
```

```
EQQZ005I SUBTASK PRE-SUBMIT IS BEING STARTED
```

```
EQQFSD1I SYSOUT DATABASE ERROR HANDLER TASK STARTED
```

```
EQQFSK1I PRIMARY KEY FILE INITIALIZATION COMPLETED
```

```
EQQFSF1I DATA FILE EQQSDF01 INITIALIZATION COMPLETED
```

```
EQQFSF1I DATA FILE EQQSDF02 INITIALIZATION COMPLETED
```

```
...
```

```
EQQFSF1I DATA FILE EQQSDFnn INITIALIZATION COMPLETED
```

```
EQQFSI1I SECONDARY KEY FILE INITIALIZATION COMPLETED
```

```
EQQFSD5I SYSOUT DATABASE INITIALIZATION COMPLETE
```

```
EQQPS01I PRE SUBMITTER TASK INITIALIZATION COMPLETE
```

```
EQQFL01I JOBLOG FETCH TASK INITIALIZATION COMPLETE
```

If the XCF is used to connect with Data Store Following messages should occur:

```
EQQFCCAI XCF JOIN STARTED
```

```
EQQFCC9I XCF XCFCLC02 HAS JOINED XCF GROUP OPCGRPQ
```

If the SNA is used to connect with Data Store following messages should occur:

```
EQQFV01I FN APPLICATION STARTED
```

```
EQQFV24I ACB SUCCESSFULLY OPENED
```

```
EQQFV36I SESSION I9PC45RA-I9PC45AA ESTABLISHED
```

Sample Message Log for a controller on page 185 shows an example of the MLOG for a Z controller. The Z controller is connected to three trackers through XCF and NCF. A standby Z controller is also started in this configuration.

Tracker message log

Look for the following messages in the log of each tracker:

Active data-router-task messages

```
EQQZ005I SUBTASK DATA ROUTER TASK IS BEING STARTED
EQQF001I DATA ROUTER TASK INITIALIZATION IS COMPLETE
```

Active submit-task messages

```
EQQZ005I SUBTASK JOB SUBMIT TASK IS BEING STARTED
EQQSU01I THE SUBMIT TASK HAS STARTED
```

Also, verify that the tracker has started an event writer. You should see these messages:

Active event-writer messages

```
EQQZ005I SUBTASK EVENT WRITER IS BEING STARTED
EQQW065I EVENT WRITER STARTED
```

If the tracker forwards events to the Z controller, ensure that an event-reader function is specified. This can be a separate event-reader task or an event writer that has been started with the EWSEQNO keyword. In some configurations, both functions can be started in a tracker.

- Although no messages are issued if you use EWSEQNO to start the reader function, check that EWSEQNO appears on the EWTROPTS statement and that the return code for this statement is 0.
- If you have specified a separate event-reader function, check that these messages appear:

  Active event-reader messages

  ```
  EQQZ005I SUBTASK EVENT READER IS BEING STARTED
  EQQR025I ERDR 01 STARTED
  ```

  The numeric value on message EQQR025I indicates which event reader is being started. The same value cannot be specified on EWSEQNO and ERSEQNO keywords at the same node, and no more than 16 reader tasks can be specified at this node.

Note: If the tracker uses an XCF connection, no messages appear in the tracker message log unless an error has occurred. To verify XCF connection messages, check the message log of the Z controller.

If the tracker uses an NCF connection, check that these messages appear (where NCFTRK01 and NCFCON01 represent the VTAM® application IDs of the tracker and Z controller):

Active NCF-connection messages

```
EQQZ005I SUBTASK VTAM I/O TASK IS BEING STARTED
EQQV001I NCF APPLICATION STARTED
EQQV024I ACB SUCCESSFULLY OPENED
EQQV036I SESSION NCFTRK01-NCFCON01 ESTABLISHED
EQQV040I CURRENTLY RUNNING WITH 'NCFCON01' AS CONTROLLER
```

Sample message log for a tracker on page 186 shows an example of the MLOG for a tracker. The tracker is connected to the Z controller via a VTAM® link. The VTAM® application IDs are NCFTRK01 for the tracker and NCFCON01 for the Z controller. The Z controller is active.

Verifying workload submission

Now verify that HCL Workload Automation for Z can submit work and that the work is sent to the correct destination.

Use the following procedure for the controlling system:

1. Create a workstation description, leave the destination field blank. This means that operations will be submitted to the system where the Z controller is started.
2. Create an application description. Define at least one operation on the workstation you have created. Submit a daily planning extend or replan batch job to include the new workstation in the current plan.

   Add an occurrence of this application to the current plan.
3. Verify that the operations run successfully on this system and that they are reported as complete in the current plan.
4. Check that submit events are created in the event data set for the controlling system (for details, see Verifying job submission on page 187).

If you have controlled HCL Workload Automation for Z systems in your configuration, use he following procedure to check that work is sent to these destinations, is submitted, and that events are returned to the Z controller:

1. Create a workstation description for each destination. In the destination field, specify the XCF member name of the tracker, the tracker VTAM® application ID, or the tracker TCP/IP destination name, depending on the type of connection.
2. Add an application occurrence to the current plan with operations defined on each of the workstations.
3. Verify that the operations run successfully on the correct system and that they are reported as complete in the current plan.
4. Check that submit events are created in the event data set for each controlled system (see Verifying job submission on page 187).

Verifying job submission

When HCL Workload Automation for Z submits work, a submit event is written to the event data set. A submit event is prefixed with an I, and can be one of these:

IJ1

Job JCL. A job has been submitted.

IJ2

Started-task JCL. A started task has been started.

IWTO

An operation has been initiated for a general workstation with the WTO option. The submit task causes message EQQW775I to be issued.

Check each system on which HCL Workload Automation for Z is installed to ensure that the destination can be reached by the Z controller and that the relevant submit events are being created in the event data set. That is, if HCL Workload Automation for Z will submit jobs, start started tasks, and initiate WTO operations, verify that all the event types are created in the event data set. You need not test all these functions if HCL Workload Automation for Z will not be used for a particular operation.

To perform this test for all type I events, start an operation on each of three workstations, all of which specify the destination of the system you are testing. The workstations must be a computer automatic workstation for IJ1 events, a computer automatic workstation with the started-task option for IJ2 events, and a general WTO workstation for IWTO events. Follow this procedure to verify workload submission:

1. Ensure that you have the correct workstations specified in the workstation description database.
2. Create a test application with an operation for each workstation you want to test, and add it to the current plan.
3. When the application has completed, browse the event data set, and verify that the required type I events have been created.
4. If the operations are not started, check that the workstation status is ACTIVE and that the workstation is not WAITING FOR CONNECTION, which means that the Z controller is waiting for the corresponding tracker to communicate.

Verifying takeover by a standby controller

To verify that a standby Z controller can take over the functions of the Z controller, perform these actions:

1. Stop the Z controller.
2. Issue the following command:

   ```
   MODIFY ssname,TAKEOVER
   ```

   For a description about how to set up automatic takeover as a runtime option, see the XCFOPTS statement in *Customization and Tuning*.
3. Check that the following message appears in the message log:

Standby Z controller message

```
EQQZ129I TAKEOVER IN PROGRESS
```

When the standby Z controller has taken over the functions of the Z controller, more messages will appear in the message log. These are the same messages that appear when a Z controller is started. Verify that the takeover was successful by following the procedures used in the verification of the Z controller.

Chapter 6. Migrating

This topic provides information to help you plan your migration from HCL Workload Automation for Z versions 9.5 and 10.1 to version 10.2.

Before migrating, ensure that you have:

- Modified any shared DASD tracker-to-controller connection to a supported connection, as described in Changing a shared DASD tracker-to-controller connection to an NCF, XCF, or TCP/IP connection on page 194.
- Installed the required PTFs, which are listed in the EQQICNVS sample job.

Planning for migration

Before attempting to perform a migration, develop a migration plan to ensure a smooth and orderly transition. A well thought-out and documented migration plan can help minimize any interruption of service. Your migration plan should address topics such as:

- Identifying which required and optional products are needed.
- Evaluating new, changed, and deleted functions.
- Defining which HCL Workload Automation for Z functions you want to add, delete, or modify.
- Defining necessary changes to:
  - The configuration
  - The initialization statements
  - Installation modifications
  - Operational procedures
  - Other related products
- Determining any restrictions during the conversion period.
- Estimating the amount of time the conversion will take.
- Defining education requirements for operators and users.
- Preparing your staff and users for migration.

The content and extent of a migration plan can vary significantly from installation to installation. For example, installations that have many installation-specific modifications might require extensive planning due to the added complexity.

Migration considerations and strategies

Initially, you should install HCL Workload Automation for Z without taking any customization actions in order to achieve a stable environment.

For instructions about using System Modification Program/Extended (SMP/E) to install HCL Workload Automation for Z, see the *HCL Workload Scheduler for Z: Program Directory*.

Before migrating to the current release and to ensure that a proper fallback migration can be performed, consider that:

- You can migrate from or fall back to previous releases *without* IPLing z/OS.
- If you are performing a fallback because of problems experienced on HCL Workload Automation for Z, be sure to keep the HCL Workload Automation for Z data sets for diagnostic purposes.
- If you migrate to and fallback from HCL Workload Automation for Z to test the environment before your *official* migration, ensure that you reallocate all HCL Workload Automation for Z data sets before the next migration exercise.
- It might occur that in the Application Description database, an application shows a timestamp for its Last Update that corresponds to a date when no changes were made. This can occur when you migrate to a recent version of HCL Workload Automation for Z, because for the new fields introduced, the database conversion program adds them as blank fields; when you access a panel, *every displayed field* is verified and if a field is blank or null, the verification program initializes the field to its default. This is considered a change because different from the original record content. For this reason, the record changed message is issued, and the Last Update date is modified.
- Ensure that the Dynamic Workload Console is migrated to a new release before migrating the Z controller, because the Dynamic Workload Console must always be at a version later than the Z controller.

Customization considerations

HCL Workload Automation for Z is designed as a general-purpose workload automation subsystem. As such, it might not meet all the requirements for your specific installation. IBM® allows installations to implement installation exits and provides many callable services that can be used to supplement HCL Workload Automation for Z processing.

Carefully examine any customization that your site has installed. Determine whether the function is now provided in the product or if you need to modify the logic based on changes made to HCL Workload Automation for Z.

When new functions are added to HCL Workload Automation for Z, installation exits and macros used within installation exits can change. New installation exits or macros might also be introduced in an HCL Workload Automation for Z release. If a release provides a new installation exit, determine if your installation needs to implement the exit. A release can change an existing exit by modifying:

- What the installation exit expects on entry.
- Return codes HCL Workload Automation for Z expects when the exit returns control to HCL Workload Automation for Z.
- The function that the installation exit performs.
- The processing that is performed before or after the exit.

Establishing the required environment

You use SMP/E to install the HCL Workload Automation for Z software. For detailed instructions, see the *Program Directory*.

JES and SMF exits

JES and SMF exits supplied with HCL Workload Automation for Z can also track work for previous releases. The exits are *always downward compatible*.

Consider installing JES and SMF exits in your current production environment at least a week before you plan to migrate any of the address spaces to HCL Workload Automation for Z.

Migrating to existing subsystem definitions

You can migrate from or fall back to your current subsystems *without having to IPL the z/OS system*.

By continuing to use your current subsystem names, you do not need to consider the effect of migration on these users of HCL Workload Automation for Z services:

- Host dialog users
- PIF programs
- API programs
- Callable services
- Dynamic Workload Console

Keeping the same subsystem names reduces the installation effort of a new level of HCL Workload Automation for Z.

Migrating to new subsystem definitions

To parallel-test the new level of HCL Workload Automation for Z with your current level, you must create new subsystems for the HCL Workload Automation for Z address space.

Getting the correct software parts

The HCL Workload Automation for Z load modules, panels, messages, and other software parts have the same name as they had in previous HCL Workload Automation for Z releases. Make sure that the users of HCL Workload Automation for Z services run the same level of software as the subsystem address space.

Load modules

Decide if you want to use the same data set name for the HCL Workload Automation for Z load modules as your previous environment. However, consider the additional effort required on your part to coordinate the JCL changes required for callers of HCL Workload Automation for Z services such as:

EQQEVPGM
EQQUSIN*x*
EQQYCOM
EQQYLTOP

If the HCL Workload Automation for Z load library is not referenced in the STEPLIB DD statement, ensure that the HCL Workload Automation for Z library is listed first in the LNKLST concatenation *and* that the library remains empty until you are ready to cut over to HCL Workload Automation for Z on the production system. Then copy the load modules into the library and perform an LLA refresh.

Two HCL Workload Automation for Z load modules *must always* be in a LNKLST library:

EQQINIT*x*

The subsystem initialization module

EQQSSCM*x*

The subsystem communication module.

However, this does not mean that you must reinitialize the subsystem to cut over HCL Workload Automation for Z to production. The module names defined for EQQINIT*x* and EQQSSCM*x* in the SYS1.PARMLIB subsystem name table (IEFSSN*nn*) can be overridden when an HCL Workload Automation for Z address space is created.

The EQQMINO*x* load module requires special attention. EQQMINO*x* is the scheduler's dialog interface module, is invoked by TSO SERVICES, and passes dialog requests and data to the controller. EQQMINO*x* must run APF authorized, therefore it must reside in an authorized library. By this token, keep in mind that any unauthorized library in a STEPLIB or LIBDEF concatenation makes the entire concatenation unauthorized. So remember to identify the library where EQQMINO*x* resides.

Use the BUILDSSX parameter of the OPCOPTS initialization statement to rebuild the environment created during subsystem initialization at the new software level. When the address space is terminated, the previous environment is reinstated, thereby ensuring fallback to a previous release of HCL Workload Automation for Z.

SSCMNAME keyword of the OPCOPTS initialization statement can be used to permanently, or temporarily, replace the EQQSSCM*x* module that was loaded into common storage at IPL. When the TEMPORARY value is defined, the named module is loaded into private storage of the HCL Workload Automation for Z address space, therefore events created while the address space is not active will use the EQQSSCM*x* from the previous IPL. When PERMANENT is specified, the old EQQSSCM*x* in common storage is replaced.

Note: Create backup copies of the old load module library before you replace the objects.

The ISPF environment

HCL Workload Automation for Z ISPF dialog users must run software parts that are at the same level as the Z controller address space. Again, using the same data set names for software parts libraries, such as messages and panels, negates the requirement to change TSO logon procedures.

If you use the same data set names, instruct dialog users to return to the TSO `READY` prompt after you have replaced the software parts and before they try to communicate with an HCL Workload Automation for Z Z controller.

The HCL Workload Automation for Z ISPF profile is automatically reinitialized when the EQQOPCAP panel (the HCL Workload Automation for Z main menu) is first displayed for a new release. Dialog users must enter the HCL Workload Automation for Z options dialog and redefine required values, such as the subsystem name.

Be sure to create backup copies of the old libraries before you replace the objects.

When migrating from one release of HCL Workload Automation for Z to the next, the LOADLIB, PANELLIB, MSGLIB, CLIB, and SKELLIB for the right HCL Workload Automation for Z release must be invoked from the TSO ISPF dialogs. Remember to identify the library where EQQMINO*x* resides.

Installation and verification

If you are migrating to existing subsystem definitions, it is required that you perform the following installation tasks:

1. Load the tracker software (Step 1. Loading tracker software on page 62).
2. Load the Z controller software (Step 2. Loading controller software on page 63).
3. Run the EQQJOBS CLIST (Step 4. Using the EQQJOBS installation aid on page 63).
4. Install JES and SMF exits at HCL Workload Automation for Z level (Step 5. Adding SMF and JES exits for event tracking on page 87).
5. Update PARMLIB (Step 6. Updating SYS1.PARMLIB on page 90).
6. Import the new default security certificates for HTTP connections (Step 8. Securing communications on page 109).
7. Allocate VSAM and non-VSAM data sets (Step 9. Allocating data sets on page 112).
8. Update the JCL procedure for the HCL Workload Automation for Z address space (Step 10. Creating JCL procedures for address spaces on page 140).
9. Update the initialization statements (Step 11. Defining the initialization statements on page 146).
10. Update the ISPF environment (The ISPF environment on page 192).

Ensure that you follow the subsystem verification procedures described in Verifying your installation on page 170.

You can use the conversion program for both migration and fallback. You should consider testing your installation by migrating in the production environment and then falling back.

Note: Verify that all the HCL Workload Automation for Z parameters defined in the previous release are still valid in the current release.

Parallel testing

If you want to perform the migration and then continue immediately into parallel testing in a job-tracking environment,you can use the following procedure as a guide. However, you should carefully consider the applicability of this procedure in your own HCL Workload Automation for Z configuration.

1. Stop your production system.
2. Perform data set conversion and copying.
3. Start your production system.
4. Start HCL Workload Automation for Z, Version 10.2 SPE (Revised July 2025).

You should also consider:

- If you start the JCC in both the production system and HCL Workload Automation for Z, the two JCCs cannot delete or requeue SYSOUT from the same SYSOUT class.
- Do not specify HOLDJOB(YES) or HOLDJOB(USER) for more than one of the two systems. If you do, one system might incorrectly release jobs that are held by the other system.
- When you convert the VSAM data sets, it is recommended that you run the conversion of the JS file to verify that conversion has been done correctly. Then, before running the parallel test, reallocate empty JS files. (Otherwise, the test system might find valid production JCL on the active JS file and submit it to the JES subsystem.)
- You should start with an empty JCL library data set (EQQJBLIB). Otherwise, the test system might submit production JCL incorrectly. To test that HCL Workload Automation for Z submits jobs correctly, you should create test

applications with job names that are not known to the production system. JCL for those jobs could then safely be inserted into EQQJBLIB.
- On the test system you should specify TRACK(ALL) and SUBFAILACTION(R) on the JTOPTS initialization statement.
- TSO commands or subroutines that have a specific name for the subsystem parameter will not report events to the test system. You should update any procedures, which are dependent on TSO commands or subroutines, if events should also be reported to the test system.
- If you are migrating from a previous release of HCL Workload Automation for Z and you use NetView® or a similar product to intercept messages, make sure that WTO (write-to-operator) messages are not issued by the test system. Otherwise, the test system might trigger some processing that impacts the production system. You should not use alert WTOs, deadline WTOs, or WTO operations on the test system.
- If you want to use Restart and Cleanup when the old subsystem is running a JCC task, you must set the DSTCLASS parameter in the RCLOPTS statement of the new controller. The class specified in DSTCLASS must not be one of the classes specified in the JCC parameter CHKCLASS. This will prevent the JCC task from deleting the duplicate SYSOUT copy created for the Data Store before it has been successfully stored. For further details about the RCLOPTS statement, see *Customization and Tuning*.

Using the preceding notes as a guide, you will be able to run one production system and one HCL Workload Automation for Z test system in parallel. The work with the database dialogs and the long-term-planning functions can be fully tested this way. The job-tracking functions of the test system will be incomplete because only the specially created test jobs will be submitted by the test system. However, the tracking of work, including the tracking of applications and jobs in the production area, will be done normally.

Migrating the backup controller

After you migrated the primary controller, you can migrate the backup controller by completing the following procedure:

1. Allocate the VSAM clusters and non-VSAM data sets by running EQQPCS01 and EQQPCS02 jobs, respectively.
2. When the backup controller starts and establishes the connection with the primary controller, the primary controller sends all the required plans.
3. As soon as the backup controller receives the plans, it performs the corresponding restore.
4. To check that the restore process for all the plans has completed, look for the following message in EQQMLOG:

```
EQQN134I RE-SYNCHRONIZATION PLANS RESTORE COMPLETED
```

Changing a shared DASD tracker-to-controller connection to an NCF, XCF, or TCP/IP connection

To change a shared DASD tracker connection to an NCF (VTAM®), XCF (SYSPLEX), or TCP/IP connection, perform the following steps.

1. Delete the DASD connection as follows:
    a. In the controller started task procedure:
        i. Delete the `EQQEVDnn` DD statement pointing to the event data set of the specific tracker.
        ii. Delete the DD statement pointing to the Submit/Release data set of the tracker. Not every DASD-connected tracker has a Submit/Release data set, but if one exists, its DDNAME in the controller procedure is the same as the destination listed under the `DASD` keyword in the `ROUTOPTS` initialization statement of the controller.
    b. In the controller initialization parameters:
        i. Decrease the value of the `OPCOPTS ERDRTASK()` keyword by the number of DASD-connected trackers being removed. If there are no DASD-connected trackers, `ERDRTASK()` is 0.
        ii. Remove from the `ERDRPARM()` keyword the name of the `PARMLIB` member containing the parameters for the Event Reader task being deleted.
        iii. Remove from the `DASD` keyword in the `ROUTOPTS` initialization statement the DDNAME of the Submit/Release data set of the tracker.
    c. In the controller ISPF dialogs:
        i. Remove the DDNAME of the Submit/Release data set of the tracker from the workstation destination under dialog option 1.1.2, using the `M` (modify) row command.
        ii. Remove the workstation destination from `ROUTOPTS` and from the workstation definition.
    d. In the tracker started task procedure, remove the `EQQSUDS` DD statement.
    e. In the tracker initialization parameters:
        i. In the `EWTROPTS` statement, set the `SUREL()` keyword to `NO`.
        ii. In the `TRROPTS` statement, remove the `HOSTCON(DASD)` keyword.
2. To add an NCF connection:
    a. Define the VTAM® LUs for the controller and the tracker. If necessary, create cross-domain definitions. HCL Workload Automation for Z requires that the LU name be the same as the `ACBNAME` in the APPL. For details, refer to Step 14. Activating the network communication function on page 155.
    b. In the controller initialization parameters:
        i. In the `OPCOPTS` statement, set the `NCFTASK()` keyword to `YES` and write the LU name of the controller in the `NCFAPPL()` keyword.
        ii. In the `ROUTOPTS` statement, write the LU name of the tracker in the `SNA()` keyword.
    c. In the controller ISPF dialogs, write the LU name of the tracker in the workstation destination under dialog option 1.1.2, using the `M` (modify) row command.
    d. In the tracker initialization parameters:
        i. In the `OPCOPTS` statement, set the `NCFTASK()` keyword to `YES` and write the LU name of the tracker in the `NCFAPPL()` keyword.
        ii. In the `TRROPTS` statement, set the `HOSTCON()` keyword to `SNA` and write the LU name of the controller in the `SNAHOST()` keyword.
        iii. In the `EWTROPTS` statement, set the `EWSEQNO()` keyword to 1.
3. To add an XCF connection:

   a. In the `SYS1.PARMLIB(COUPLEnn)` member:
      i. Define the HCL Workload Automation for Z XCF transport class, as described in Updating XCF initialization options on page 98.
      ii. Define the XCF group that is to enable the controller to communicate with the trackers.
   b. In the controller initialization parameters:
      i. In the `XCFOPTS` statement, code the `GROUP()`, `MEMBER()`, and `TAKEOVER()` keywords.
      ii. In the `ROUTOPTS` statement, write the `XCF MEMBERNAME` of the tracker in the `XCF()` keyword.
   c. In the controller ISPF dialogs, write the `XCF MEMBERNAME` of the tracker in the workstation destination under dialog option 1.1.2, using the `M` (modify) row command.
   d. In the tracker initialization parameters:
      i. In the `XCFOPTS` statement, code the `GROUP()` and `MEMBER()` keywords.
      ii. In the `TRROPTS` statement, set the `HOSTCON()` keyword to `XCF`.
      iii. In the `EWTROPTS` statement, set the `EWSEQNO()` keyword to 1.
4. To add a TCP/IP connection:
   a. Define the IP addresses for the controller and tracker. For details, see Step 15. Using TCP/IP for communication on page 158.
   b. In the controller initialization parameters:
      i. In the `TCPOPTS` statement, set the values to define the details of the local controller. This statement is optional; if you do not specify it, the default values are taken.
      ii. In the `ROUTOPTS` statement, write the TCP/IP destination name and IP address of the remote tracker in the TCPIP keyword.
   c. In the controller ISPF dialogs, write the TCP/IP destination name of the tracker in the workstation destination under dialog option 1.1.2, using the `M` (modify) row command.
   d. In the tracker initialization parameters:
      i. In the `TCPOPTS` statement, set the values to define the details of the local tracker or leave the default values.
      ii. In the `TRROPTS` statement, set the `HOSTCON()` keyword to `TCPIP` and write the IP address of the controller in the `TCPHOSTNAME()` keyword.
      iii. In the `EWTROPTS` statement, set the `EWSEQNO()` keyword to 1.
5. Stop and restart the controller and tracker for the parmlib changes to take effect, and run the `CP extend` or the `CP replan` command to update the current plan with the changed workstation destinations.

Running on upgraded operating systems

To run the scheduler on a new version of the z/OS® operating system, you must reassemble the SMF and JES exits mentioned in Step 5. Adding SMF and JES exits for event tracking on page 87 with the libraries of the new operating system.

If you upgrade the SMP/E environment to a later version of z/OS® by using the SMP/E function BUILDMCS, the relink occurs automatically (ensure that the DDDEF entries for the new operating system are set up correctly by specifying the latest SEZACMTX and SCEELKED libraries). If you do not use BUILDMCS, relink the load modules by using the SMP/E function LINK LMODSCALLLIBS. With this function, all the HCL Workload Automation for Z modules are relinked to the latest SEZACMTX and SCEELKED libraries.

HOME directory requirements for started tasks

For any user IDs running the controller, server, output collector, or TCP connected tracker started task, you are required to define a HOME directory in their OMVS segment. The HOME directory must exist, be mounted, and the user ID must be assigned write permission to it.

To ensure that a HOME directory is defined for the user ID running the started task, issue the following RACF command:

```
LU <started_task_userID> OMVS NORACF
```

The output must return a row as in the following example:

```
 HOME= /u/<started_task_userID>
```

Migrating actions

The following topics describe the procedures for:

- Migrating data sets on page 197
- Migrating the production environment on page 204

To migrate the controller by using the job stream provided with the product, see Migrating the controller with the IWSZSELFUPGRADE application on page 207.

If after migrating you need to return to the previous version, follow the procedure described in Performing fallback on page 211.

Migrating data sets

For migration purposes, data sets fall into three categories:

- VSAM data sets that are copied and converted by the EQQICTOP migration program
- Non-VSAM data set that you can copy, or use unchanged, in the new version
- VSAM and non-VSAM data sets that must be empty when you migrate to HCL Workload Automation for Z

Each of these categories is described in the following topics.

EQQICTOP VSAM data set conversion program

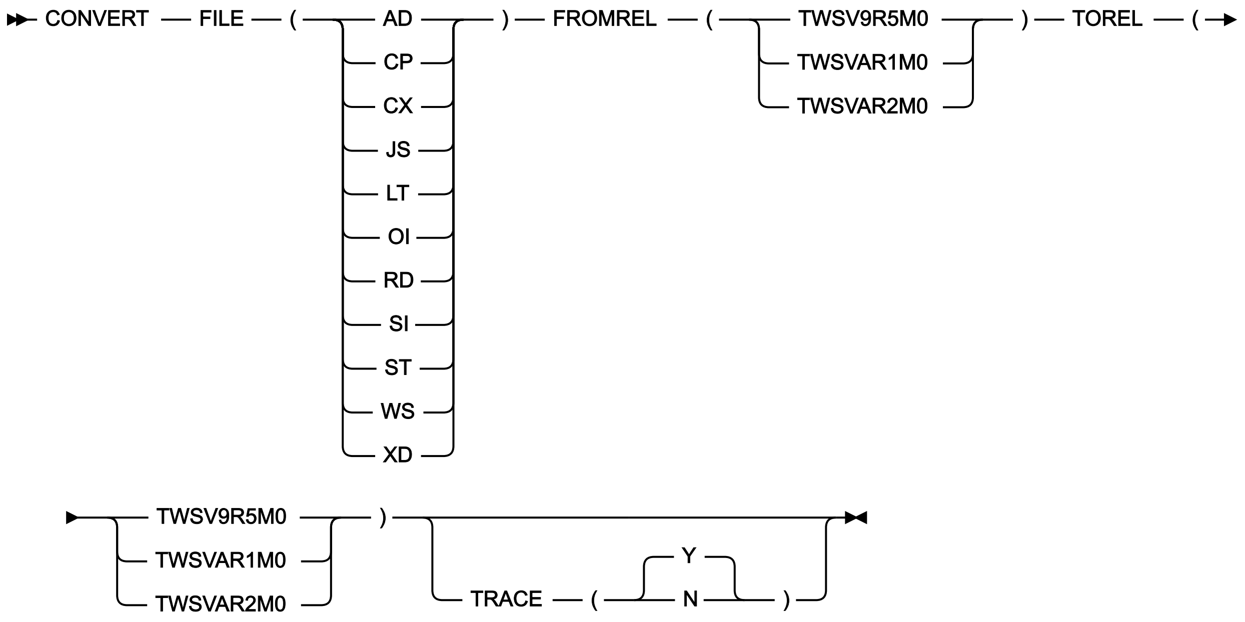
Purpose

With the EQQICTOP conversion program, you can migrate VSAM data sets from earlier releases of HCL Workload Automation for Z. You can also use the program to reverse the procedure in case you need to fall back to your old system.

The EQQJOBS program creates JCL tailored to your installation specifications in the EQQICNVS member.

EQQICTOP is controlled by CONVERT statements in the SYSIN file. You can supply any number of these statements to EQQICTOP.

Syntax



Parameters

FILE(*file identifier*)

Defines the data set to be converted. You can specify one of the following file identifiers on each CONVERT statement:

AD

Application descriptions and JCL variable tables

CP

The current plans, EQQCP*n*DS and EQQNCPDS

CX

The current plan extension, EQQCXDS and EQQNCXDS

JS

JCL repository and retrieved job logs

LT

Long-term plan

OI

Operator instructions

RD

Special resource definitions

SI

Side information file, ETT criteria and configuration information

ST

Step awareness (only if the Step Awareness feature is active on the primary controller)

WS

Workstation descriptions, calendars, and periods

XD

Extended dependencies

Your conversion JCL should contain DD names EQQ*xx*IN and EQQ*xx*OUT for each data set that you want to convert, where *xx* is the file identifier.

FROMREL(*product identifier*)

Defines the product and release level of the input data set. You can specify one of the following:

TWSV9R5M0

HCL Workload Automation for Z Version 9 Release 5.

TWSVAR1M0

HCL Workload Automation for Z Version 10 Release 1.

TWSVAR2M0

HCL Workload Automation for Z Version 10 Release 2.

TOREL(*product identifier*)

Defines the product and release level of the output data set. You can specify one of the following:

TWSV9R5M0

HCL Workload Automation for Z Version 9 Release 5.

TWSVAR1M0

HCL Workload Automation for Z Version 10 Release 1.

TWSVAR2M0

HCL Workload Automation for Z Version 10 Release 2.

Note:

1. Conversion stops if there is a VSAM I/O error on one of the files. One such error is a duplicate key on the output file. This can occur if the output data set is not empty.
2. Migrate the currently active JCL-repository data set. You can check whether the primary or alternate data set is in use by selecting option 6 in the Query Current Plan dialog. Do this when no work is running and before you stop the Z controller.

3. You can use one of two methods to convert the current plan:
    - If the last action performed on your production system was to extend or replan the current plan, use the new-current-plan data set that was created on this system as input to the conversion program. This is the preferred method as it ensures you will not lose any job-tracking records, this is relevant if you use the track log (EQQTROUT) as an audit trail.
    - If no error occurred when you stopped your production system, both primary and alternate current plans are the same. Use EQQCP1DS as input to the conversion program.

   In both cases, the output file **must** be the new-current-plan data set (EQQNCPDS) on your HCL Workload Automation for Z system.

   You can convert the current plan extension data set using the same methods. If the EQQCPIN DD card in the EQQICNVS conversion program refers to EQQNCPDS, the EQQCXIN DD card must refer to the corresponding EQQNCXDS created with the latest REPLAN or EXTEND job. If, instead, no REPLAN or EXTEND job was run before shutting down the system,the EQQCXIN DD card must refer to the latest EQQCXDS data set in use. In both cases, the EQQICNVS output file for CX must be the new-current-plan extension data set (EQQNCXDS) corresponding to the new-current-plan data set (EQQNCPDS).
4. In addition to input and output DD names for each VSAM file, the migration JCL should also contain the EQQMLOG and EQQMLIB DD names. EQQMLOG is an output file for messages. EQQMLIB is an input file that contains the product message library.

TRACE(Y|N)

Specifies if message:

```
EQQIC66I PROCESSING APPLICATION AD_data_set_name VALID FROM From_date
         STATUS status
```

is to be issued in the message log of the migration job as each data set of the Application Description database is migrated to the new product release.

Set TRACE to N specifically to inhibit the issue of message EQQIC66I; otherwise, the message is written to the MLOG as part of the migration process.

Note: Specifying TRACE(Y), or not specifying it at all (the default), will cause one occurrence of message EQQIC66I to be issued for each processed application. Consider using TRACE(N) if you do not want this message to be issued in the batch output MLOG.

Data sets that you need to migrate

Allocate new VSAM data sets for HCL Workload Automation for Z. Existing data can then be migrated by using EQQICTOP. Keep a copy of the old data sets for backup and fallback purposes. The following data sets must be migrated to HCL Workload Automation for Z format:

Table 31. Data sets that you need to migrate

DD Name	Description
EQQADDS	Application descriptions and JCL variable tables.
EQQJS*n*DS	JCL repository (currently active).
EQQLTDS	Long-term plan.
EQQNCPDS, or EQQCP*n*DS	The current plan, but use the NCP as input if a daily plan process created an NCP after the old system was shut down.
EQQNCXDS, or EQQCXDS	The current plan extension, but use the NCX as input if a daily plan process created an NCX after the old system was shut down.
EQQOIDS	Operator instructions.
EQQRDDS	Special resource definitions.
EQQSIDS	Side information, ETT criteria and configuration information.
EQQSTDS	Step Awareness (only if the feature is active on the primary controller).
EQQWSDS	Workstation descriptions, calendars, and periods.
EQQNXDDS and EQQXD*n*DS	Extended data.

Data sets that can be used without migrating

HCL Workload Automation for Z can use unchanged data from the following data sets:

Table 32. Data sets that HCL Workload Automation for Z can use

DD Name	Description
EQQEMAIL	Email data set
EQQEVLIB	Configuration file repository for event-triggered resource handling
EQQFLEX	License data sets

Table 32. Data sets that HCL Workload Automation for Z can use (continued)

DD Name	Description
EQQINCWK	JCC incident work file
EQQJBLIB[1]	JCL library
EQQJCLIB	Job-completion-checker (JCC) message-table library
EQQJTABL	Job table log file
EQQOUCEV	Data set used for controller-output collector communication
EQQPRLIB	Automatic-recovery-procedure library
—	JCC incident log
[1] If this library contains jobs for the scheduler planning, the JCL must be modified to reflect the new installation.	

Empty data sets

When you have completed your testing of HCL Workload Automation for Z and have performed data set migration, ensure that the following data sets are empty before you start the product for the first time in production:

EQQACPDS

Archive of old current plan

EQQBKPT

Backup checkpoint

EQQCKPT

Checkpoint

EQQCXDS

Current plan extension

EQQDBARC

Extended-auditing archive

EQQDB*nn*

Extended-auditing logs

EQQDBOUT

Extended-auditing tracklog data set

EQQDL*nn*

Dual job-tracking logs

EQQEVDS

Event data set in the controller

EQQHTTP0

Event data set for end-to-end scheduling with z-centric capabilities

EQQJTARC

Job-tracking archive

EQQJT*nn*

Job-tracking logs

EQQLOGRC

Job log and Restart Information pending requests Log data set

EQQMLOG

Message log

EQQMONDS

Monitoring Task Data Set

EQQNSTDS

New Step Awareness

EQQSCPDS

Secondary Current Plan Data Set

EQQSTC

Started-task submit

EQQTROUT

Job-tracking log copy created by the daily planning jobs. Input to the EQQAUDIT program, that is not downward compatible

EQQCP*n*DS, EQQXD*n*DS, EQQCXDS, and EQQSCPDS do not have to be empty, but they can be. When the product is started for the first time, you **must** specify CURRPLAN(NEW) on the JTOPTS statement. Therefore, any data in the EQQCP*n*DS and EQQCXDS data sets will immediately be replaced by the contents of EQQNCPDS and EQQNCXDS. Similarly, the inactive EQQJS*n*DS data set (EQQJS2DS in this example) does not have to be empty, although it can be.

Tracker, Data Store, and Output Collector considerations

When you migrate a tracker, it is not necessary for EQQEVDS to be empty. The migration enables you to use the subsystem with the new release and modifies the JCL trackers to point to the libraries used, for example EQQMLIB or STEPLIB.

The first time you start a tracker migrated to a new version, ensure that in the OPCOPTS statement you specified BUILDSSX(REBUILD) and SSCMNAME(EQQSSCM2,PERMANENT). However, this setting of OPCOPTS might cause the loss of some data set triggering events on the tracker; to prevent this from occurring, do not set BUILDSSX(REBUILD) and SSCMNAME(EQQSSCM2,PERMANENT) but set the INITRTN and INITPARM of the IEFSSN*nn* member in SYS1.PARMLIB as follows:

```
SUBSYS SUBNAME(subsysten name)
INITRTN(module name)
INITPARM('maxecsa,suffix')
```

In this way, after the IPL of the z/OS system, the tracker will be migrated without requiring any parameter change. Because the Z controller and trackers can communicate even if they are at different versions, you can migrate the trackers before or after migrating the Z controller.

When you migrate a Data Store, consider that:

- In the Data Store, it is not necessary for EQQPKI01, EQQSKI01, EQQSDF*nn*, EQQUDF*nn* to be empty.
- In the Z controller, it is not necessary for EQQPKI01, EQQSKI01, and EQQSDF*nn* to be empty.

The data store and controller tasks might be migrated at different times, provided that the maintenance level of the old and new release of HCL Workload Automation for Z is the same. This means that you should apply any PTF which affects both controller and data store code to both releases of the product. If this level of concurrent PTF maintenance cannot be maintained, it is best to keep the data store and controller on the same release of HCL Workload Automation for Z. If the migration is successfully performed, you should be able to use the Restart and Cleanup function on the new release for any operation which was on the error list on the old release of HCL Workload Automation for Z

If you change a datastore connection type and you want to reflect the naming convention in the FLOPTS destination name, keep the former destination name in the FLOPTS parameter that corresponds to the connection type to be used (SNADEST, XCFDEST, or TCPDEST). For example, suppose that you change the datastore connection from SNA to XCF and the former FLOPTS is `SNADEST(OPCTRK1.DST)`. If you want to use `XCFTRK1.DST` as new destination name, specify the following FLOPTS parameter: `XCFDEST(OPCTRK1.DST, XCFTRK1.DST)`. Omitting the former destination produces the messages EQQFL18E and EQQM643W in the controller message log, when retrieving any joblog stored with the former destination name.

Output collector is not to be migrated.

Migrating the production environment

To migrate your production system, perform the following steps:

1. Close down your production system on page 205
2. Convert VSAM files to the new system format on page 205
3. Initialize the new system on page 205
4. Produce a checkpoint data set containing data from the old production system on page 206
5. Start the new system on page 206
6. Validate the new system on page 207

Close down your production system

1. From the Daily Plan dialog on the production system on the controller, create a replan or plan-extend batch job. Change the job card to contain TYPRUN=HOLD, and submit the job. Save the JCL in a data set in case you have to resubmit it to correct an error.
2. If you specified CHECKSUBSYS(YES) in the BATCHOPT statement used by the batch job, change it to CHECKSUBSYS(NO). In the BATCHOPT statement used by the batch job comment out the JRUNHISTORY parameter, if it is used.
3. Using the Query Current Plan dialog on the controller production system, check which JS file is currently in use on this system.
4. Stop the controller and server, release the daily plan from hold, and make sure it runs successfully.

Convert VSAM files to the new system format

1. Create a backup copy of the HCL Workload Automation for Z VSAM files.
2. Back up the HCL Workload Automation for Z non-VSAM data sets.
3. Allocate VSAM and non-VSAM data sets for HCL Workload Automation for Z by using the EQQPCS01 and EQQPCS02 jobs.
4. Review the EQQICNVS sample job. Ensure that input and output data set names are correctly specified.

   Ensure that JS1 is the active JCL repository in option 6.6 on the previous Z controller. If JS2 is the active JCL repository, use that as input, but make sure that JS1 is used as output because HCL Workload Automation for Z by default uses JS1 as the active JCL repository when you start a subsystem with an empty checkpoint data set.

   The old NCP is used as input if a daily plan batch process was submitted on the previous system prior to shut down. Output is the HCL Workload Automation for Z NCP.
5. Run EQQICNVS to convert the VSAM data to HCL Workload Automation for Z format.
6. Verify that the conversion program ran successfully. If there are any problems converting the VSAM files, you should abandon the migration.

Initialize the new system

Before you perform the steps described in this section, ensure that the VSAM file conversion described in the preceding section was successful.

1. Ensure that the data sets referred to in Empty data sets on page 202 are empty. Use ISPF browse to ensure that all job-tracking logs (EQQJT*nn*), the job-tracking archive (EQQJTARC), and the checkpoint (EQQCKPT) data sets are empty. If you use dual job-tracking logs (EQQDL*nn*), they should also be empty.
2. Modify the JCL procedure for the controller to include the new DD names and data sets added in HCL Workload Automation for Z.
3. Modify initialization parameters for the controller. Because the CKPT data set is not yet initialized the first time you start the controller after migration, you must set CURRPLAN(NEW) in the JTOPTS statement. Specify BUILDSSX(REBUILD) and SSCMNAME(EQQSSCM2,TEMPORARY) in the OPCOPTS statement. Specify the PIFHD

parameter in the INTFOPTS statement. As soon as the controller has started, change back to CURRPLAN(CURRENT), to prevent the controller from recovering from the new current plan each time it starts.

> **Note:** You might find it useful to specify JOBSUBMIT(NO) in the JTOPTS statement so that work is not submitted when you start the controller. When you have checked that the controller has started without errors, you can activate job submission using the Service Functions dialog.

To initialize the checkpoint data set, specify OPCHOST(YES) in OPCOPTS. In this way, when the scheduler starts, the NMM task initializes the checkpoint data set with FMID and LEVEL corresponding to SSX. You can then change the OPCHOST value. For example, you can change the value to OPCHOST(PLEX) when the subsystem is used as the controlling system in XCF.

4. Start the controller. Verify that no errors occurred during initialization. If required, correct any errors and restart the controller.
5. Stop the controller.

Produce a checkpoint data set containing data from the old production system

Produce a checkpoint data set containing data from the old production system:

1. Merge OLD.CKPT, from the version you are migrating, and the newly allocated CKPT, created in the previous section, into CKPT.NEW using sample EQQPMCKP, which you customize for your environment.
2. Back up the current CKPT and then rename CKPT.NEW to the current CKPT.

Start the new system

1. Change JTOPTS CURRPLAN(NEW) to CURRPLAN(CURRENT).
2. In the BATCHOPT statement used by the batch job, uncomment the JRUNHISTORY parameter if you had commented it out.
3. Start the controller. The merged checkpoint data set will enable it to continue reading the event records.
4. The first time you start the trackers at the version to which you are migrating, in the OPCOPTS statement specify BUILDSSX(REBUILD) and SSCMNAME(EQQSSCM*n*,PERMANENT) then delete these parameters because no longer required.
5. Enter the Service Functions dialog on the controller, and activate job submission (if it is not already active).
6. Submit a daily plan replan or extend **as soon as possible** after migration. Until a new current plan is created, any references to special resources will cause the resource object to be copied from the EQQRDDS to the current-plan-extension data space. This processing has some performance overheads.

   The new-current-plan-extension data set (EQQNCXDS) is built during daily planning to contain all special resources referenced by operations in the new current plan.
7. After the first IPL of the z/OS system, delete the BUILDSSX and SSCMNAME parameters from the OPCOPTS statement in the Z controller and trackers, because they are no longer required.

Validate the new system

1. From the Ready List dialog, review the status of active operations.
2. Check that the operations that are becoming ready on the workstations representing the three z/OS systems are successfully submitted to the intended system. Also check that the ending status is correctly reflected in the ready lists.
3. Verify that the current plan and the long-term plan can be extended successfully.
4. Verify that other HCL Workload Automation for Z-related processes (for example, the dialogs, batch programs, and PIF-based programs) work as expected.

Migrating the controller with the IWSZSELFUPGRADE application

You can upgrade the HCL Workload Automation for Z controller in an automatic way, with only few manual steps, by using the `IWSZSELFUPGRADE` application that is provided with the product.

In `IWSZSELFUPGRADE`, the jobs requiring manual actions are defined as dummy operations on a manual start and completion workstation.

`IWSZSELFUPGRADE` is provided in batch loader and Workload Automation Programming Language formats that you can import by using the EQQUPGBL or EQQUPGWA sample, respectively.

After importing the `IWSZSELFUPGRADE` application, to migrate an HCL Workload Automation for Z controller complete the following procedure.

Note:

1. Before running `IWSZSELFUPGRADE`, ensure that you set `VARSUB=YES` in the OPCOPTS statement.
2. If you are using JES3 exit, modify the operations 004 and 005 in `IWSZSELFUPGRADE` to replace EQQJES2 and EQQJES21 with EQQJES3.

1. Copy `IWSZSELFUPGRADE` to the Application Description.
2. Customize the following jobs as required for your migration purposes:

   EQQALPDS

   To allocate the data sets required to run EQQJOBS.

   EQQNPJOB

   To mark as NOP all the operations that install or migrate optional functions (such as Restart and Cleanup or Reporting) that you do not use.

   EQQCPMOD

   To copy the load modules EQQSSCM*x* and EQQINIT*x* to the user library that needs to be APF authorized and added to the z/OS system LINKLIST.

EQQCPPAR

To copy the old PARMLIB to the new PARMLIB.

3. Add the `IWSZSELFUPGRADE` application to the current plan.

The following operations are run automatically or wait for your manual intervention:

Operation 001 (automatic)

The data sets required for EQQJOBS are allocated.

Operation 002 (manual)

You are required to run EQQJOBS and copy the generated sample JCLs to the job library data set (EQQJBLIB).

Note: Ensure that in the Create Sample Job JCL (EQQJOBS3) panel, you specify the `//&OJOBNAME JOB` card in the Job Statement Information field.

Operation 003 (automatic)

The NOPJOB job is run.

Operations 004, 005, and 006 (automatic)

The samples generated by EQQJOBS named EQQJES2, EQQJES21, and EQQSMF are run to link the JES2 and SMF exits.

Note: If you are using JES3 exit, you must have modified the operations running EQQJES2 and EQQJES21 to an operation running EQQJES3.

Operation 009 (manual)

You are required to manually update the following members:

- IEFSSN*nn* defined for the load modules EQQINIT*x* and EQQSSCM*x*.
- IKJTSO*nn* defined for the load module EQQMINO*x*.

For detailed information about the load modules, see Load modules on page 191.

Operations from 010 to 113 (automatic)

The following jobs are automatically run, unless you marked them as NOP.

Note: Because EQQPCS02 contains system symbols, if you want to use them you must make EQQPCS02 a started task or batch job by copying it to a procedure library. Then, define the workstation where this operation is run as `STARTED TASK, STC = Y`.

- EQQRCERT
- EQQPCS01

  - EQQPCS02
  - EQQPCS03
  - EQQPCS04
  - EQQPCS07
  - EQQPCS08
  - EQQPCS09
  - EQQPCS10
  - EQQPCS11
  - EQQPCS12
  - EQQPCS13

Operation 120 (manual)

You are required to update the controller startup procedure.

Operation 130 (automatic)

The EQQICNVS sample job is automatically run to migrate the VSAM data sets.

Operation 150 (automatic)

The COPYMOD job is automatically run to copy the load modules EQQSSCM*x* and EQQINIT*x* to the PARMLIB.

Operation 151 (automatic)

The COPYPARM job is automatically run to copy the old PARMLIB to the new PARMLIB.

Operation 152 (manual)

You are required to update the new PARMLIB as required.

After the `IWSZSELFUPGRADE` application completes successfully, you can start the subsystem that was migrated.

Installing or upgrading the HCL Workload Automation for Z agent automatically

You can automatically install or upgrade an HCL Workload Automation for Z agent (z-centric) or an agent with dynamic capabilities by customizing and running the following applications provided with the product. The applications are provided in both batch loader and Workload Automation Programming Language formats.

`IWSZZCENINSTALL`

To upgrade an HCL Workload Automation for Z agent. Run this application on a workstation where an HCL Workload Automation for Z agent instance exists, to automatically upgrade it.

Use the EQQZCEBL sample to import the batch loader format or the EQQZCEWA sample to import the Workload Automation Programming Language format.

`IWSZZREMINSTALL`

To install an HCL Workload Automation for Z agent by running a Remote Command job defined on the Dynamic Workload Console. Run this application on a workstation where an HCL Workload Automation for Z agent instance exists, to automatically install as many HCL Workload Automation for Z agents on as many workstations as you want.

Use the EQQZREBL sample to import the batch loader format or the EQQZREWA sample to import the Workload Automation Programming Language format.

`IWSZZDYNINSTALL`

To upgrade a dynamic agent. Run this application on a workstation where a dynamic agent instance exists, to automatically upgrade it.

Use the EQQZDYBL sample to import the batch loader format or the EQQZDYWA sample to import the Workload Automation Programming Language format.

Before importing the `IWSZZCENINSTALL`, `IWSZZREMINSTALL`, or `IWSZZDYNINSTALL` application to the AD database, complete the following customization steps:

1. The applications apply to the Linux operating system. To use them in a Windows environment, modify the paths and commands within each sample job as required.
2. The workstation names associated with the operations within the samples are `A130` and `Z130`. Ensure that you either create these workstations in your environment or modify the samples with your actual workstation names, as follows:

   `A130`

   The agent running the command (*driving* agent).

   `Z130`

   The agent that is to be installed (*target* agent).
3. All the jobs that are run by the `IWSZZCENINSTALL` or `IWSZZDYNINSTALL` application are defined in the EQQZCJCL sample. Create the corresponding jobs in the controller JOBLIB and customize them according to your environment, as follows:

   `/opt/IBM/TWA_twszc`

   Installation path of the target agent.

   `Buildzcen`

   Directory of the target agent where the agent will be installed.

   `/mnt/build`

   Source directory on the target agent from which the new agent code will be copied.

-uname `twszc`

Name of the user for which the HCL Workload Automation for Z agent is installed.

/JDBC_drivers

Installation path of the JDBC drivers on the target agent.

The `IWSZZCENINSTALL`, `IWSZZREMINSTALL`, and `IWSZZDYNINSTALL` applications comprise the following jobs, connected by internal dependencies:

Operation 001, workstation A130, job name JOBSTOP

Stops the target agent.

Operation 002, workstation A130, job name JOBUNINS

Uninstalls the target agent by running the `twsinst` command.

Operation 003, workstation A130, job name JOBDLBLD

Deletes the source directory containing the old compressed file for the target agent.

Operation 004, workstation A130, job name JOBDLINS

Deletes the target agent installation path.

Operation 005, workstation A130, job name JOBCOPY

Copies the compressed file to install the agent from the driving agent to the target agent installation path.

Operation 006, workstation A130, job name JOBUNTAR

Runs the uncompress command of the agent code.

Operation 007, workstation A130, job name JOBINSZC (in `IWSZZCENINSTALL`) or JOBINSDY (in `IWSZZDYNINSTALL`)

Installs the target agent by running the `twsinst` command.

Operation 008, workstation A130, job name JOBJDBC (automatic)

Copies the JDBC drivers to the target agent.

Operation 009, workstation WAIT

This operation waits for a few minutes, to ensure that the target agent is started.

Operation 010, workstation Z130, job name

Runs a sample job on the target agent to verify that the installation was successful.

Choose which application you want to run and use the appropriate sample to load it into your AD database.

Performing fallback

If a problem occurs after HCL Workload Automation for Z has been active as a production system for some time, and the problem is serious enough, you might need to stop the new system and return the workload to the previous system. You can do this by using a procedure called *fallback*, if the HCL Workload Automation for Z data sets are usable.

Note: If on the primary controller the Step Awareness feature was active and you want to keep it in the previous system, ensure that you convert the EQQSTDS data set.

The fallback procedure is as follows:

1. Run the EQQPCS01 and EQQPCS02 jobs to allocate new data sets for the old production system. The current data sets, or a copy, used by the HCL Workload Automation for Z systems should be kept for problem determination purposes.
2. If required, close down the systems in the same way as during migration. This is required if the current plan on the controller is intact and job tracking is working normally.
3. If possible, create up-to-date data sets for the long-term plan and new current plan for the controller.

   Do not submit a REPLAN job prior to shutdown, unless the PERMANENT option was used for SSCMNAME on the converted system, or if SSCMNAME was not specified.

   If SSCMNAME(EQQSSCM2,TEMPORARY) was used, message EQQX145E will be issued if a REPLAN job is started after the controller is shut down.
4. Build VSAM data sets for the old system by running the EQQICNVS job to convert HCL Workload Automation for Z files to their previous format.

   Note: Before running EQQICNVS job, check that a daily plan batch process was not submitted on the system before shutting the controller down. Then, in EQQICNVS set the HCL Workload Automation for Z CP1, CX, and XD1 as the input, and NCP, NCX, and NXD as the output.
5. Ensure that in the JTOPTS initialization statement you set CURRPLAN(NEW).

   Note: You might find useful to specify JOBSUBMIT(NO) in JTOPTS, so that work is not submitted when you start the controller. After checking that the old system has started without errors, you can activate job submission by using the Service Functions dialog.
6. Start the controller and trackers again using the converted files and start the server.
7. If the new current plan (NCP) data set is not fully up-to-date because you could not run the daily plan program, use the MCP dialog to update the status of operations to make the current plan up-to-date.
8. Start the trackers again. Use the SSCMNAME parameter on the JTOPTS initialization to load the current subsystem communication module for the release to which you are falling back.

Part III. Installing Dynamic Workload Console and Z connector

How to install, upgrade, configure, and uninstall the Dynamic Workload Console and Z connector, which is automatically installed with a Dynamic Workload Console instance.

The Dynamic Workload Console is a web-based user interface that is used with the following products:

- HCL Workload Automation
- HCL Workload Automation for Z

To use the Dynamic Workload Console you must configure the Z connector, which is automatically installed with a Dynamic Workload Console instance. For details about how to configure the Z connector, see Defining a z/OS engine in the Z connector on page 261.

When you install the Dynamic Workload Console, the following products are also installed (for more information, see Mobile Applications Users Guide):

- Self-Service Catalog
- Self-Service Dashboards

You can access HCL Workload Automation environments from any location in your network using one of the supported browsers connected to the Dynamic Workload Console. The Dynamic Workload Console must be installed on a system that can reach either the HCL Workload Automation using network connections.

Chapter 7. Preparing the installation

To install and use the Dynamic Workload Console, perform the following procedure.

1. Verify that your system is compliant by checking the installation prerequisites in the Dynamic Workload Console Detailed System Requirements.
2. Install the Dynamic Workload Console by following the instructions provided in Installing a Dynamic Workload Console server on page 247.
3. Log in to the Dynamic Workload Console.
4. To effectively manage the functions available in the Dynamic Workload Console, create *engine connections* to the HCL Workload Automation environments that you want to manage. Without defining engine connections, you can use only a limited set of Dynamic Workload Console functions. For more information, see Defining a z/OS engine in the Dynamic Workload Console on page 264 and Defining a z/OS engine in the Z connector on page 261.

Installation paths

This section describes the default installation paths of the Dynamic Workload Console.

UNIX ***DWC_DATA_dir* configuration directory**

To simplify administration, configuration, and backup and recovery on UNIX systems, a new default behavior has been implemented with regard to the storage of product data and data generated by HCL Workload Automation, such as logs and configuration information. These files are now stored by default in the `<data_dir>` directory, which you can optionally customize at installation time.

By default, this directory is *DWC_home*`/DWC_DATA` for the Dynamic Workload Console. The product binaries are stored instead, in the installation directory.

You can optionally customize the `<data_dir>` directory at installation time by setting the **--data_dir** argument when you install using the command-line installation. If you want to maintain the previous behavior, you can set the **--data_dir** argument to the HCL Workload Automation installation directory.

If you deploy the product components using Docker containers, the `<data_dir>` is set to the default directory name and location, and it cannot be modified.

To retrieve the *DWC_DATA_dir* location in case you have modified the default path, check the values for the DWC_datadir properties stored in the `twainstance<instance_number>.TWA.properties` file. The file is located in `/etc/TWA`.

***DWC_home* installation directory**

The Dynamic Workload Console can be installed in the path of your choice, but the default installation directory is as follows:

Windows **On Windows™ operating systems**

```
%ProgramFiles%\wa\DWC
```

UNIX **On UNIX™ operating systems**

/opt/wa/DWC

z/OS **On z/OS operating system**

/opt/wa/DWC

Downloading installation images

About this task

You can download installation images from Flexnet or from HCL Software.

1. Ensure that your workstation has sufficient space to store the compressed file containing the installation images. For more information about system requirements, see the product requirements in the online documentation..
2. From Flexnet or from HCL Software, download the compressed file, containing the latest product image, to a temporary directory.
3. Extract the installation image from the downloaded file and verify that the installation image is complete.
   Extract the content of the ZIP files into a directory, using one of the extraction tools available on your system or that can be downloaded from the internet. The tool you use must be able to keep the file permissions on the extracted files, for example, `infozip`.

   Windows On Windows™ systems, ensure that you extract the image into a path that is not very long, otherwise, the file name might be truncated. The maximum length allowed is 255 characters.

   UNIX If you are installing on a UNIX™ operating system, run the following command:

   ```
   chmod -R 755 <imagesDir>
   ```

   Windows **Note:** To extract the **.zip** file onto a Windows™ 64-bit system, ensure that the image is not located on the desktop because the Windows™ operating system extract tool might encounter a problem. Choose another directory into which to extract the product image.

Note: DB2 is available for download from HCL License Portal only. The latest versions of Open Liberty can be downloaded from Get started with Open Liberty. For further details, see the HWA_10.2.5_QuickStartGuide.zip available from Flexnet or from HCL Software.

Dynamic Workload Console prerequisites

The Dynamic Workload Console installation has the following prerequisites.

WebSphere Application Server Liberty Base

To install it, download the appropriate image from Recommended updates for WebSphere Application Server Liberty.

WebSphere Application Server for z/OS Liberty

Ensure that your system meets the operating system and Java requirements. For more information, see WebSphere Application Server for z/OS Liberty detailed system requirements.

Ensure that you set the following z/OS UNIX system services variables according to your environment requirements:

- MAXMMAPAREA (minimum required value: 122880)
- CPUTIMEMAX (minimum required value: 915827882)
- ASSIZEMAX (minimum required value: 2147483647)

Before you install the Dynamic Workload Console for the first time, ensure you have a supported database installed. You can choose to use any of the supported relational database management systems (RDBMS).

If you are installing on a z/OS system

DB2 for z/OS

See Creating and populating the database for DB2 for z/OS for the Dynamic Workload Console on page 224

If you are *not* installing on a z/OS system

DB2

See Creating and populating the database for DB2 for the Dynamic Workload Console on page 222

DB2 for z/OS

See Creating and populating the database for DB2 for z/OS for the Dynamic Workload Console on page 224

Oracle

See Creating the database for Oracle for the Dynamic Workload Console on page 228

MSSQL

See Creating and populating the database for MSSQL for the Dynamic Workload Console on page 230

MSSQL cloud-based databases

See Creating and populating the database for MSSQL cloud-based databases for the Dynamic Workload Console on page 232

PostgreSQL

See Creating and populating the database for PostgreSQL for the Dynamic Workload Console on page 234.

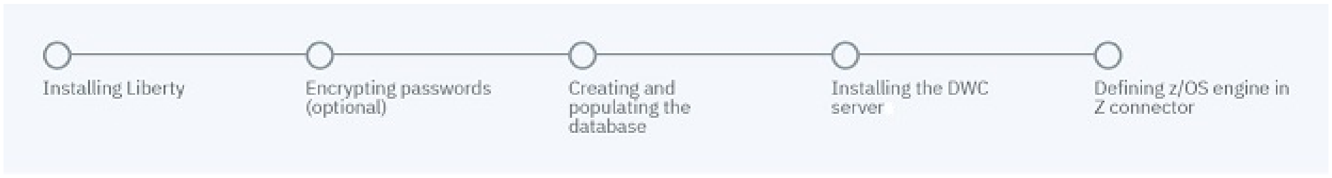
If you purchase a new DB2 license, you can typically use your existing DB2 installation without needing to reinstall the software. You simply need to apply the new license certificate to your current DB2 setup using the db2licm command to update your license compliance. For more information, see Applying Db2 licenses. If you have any further questions regarding your license on your account, contact your sales representative.

Chapter 8. Installing the Dynamic Workload Console

Install the Dynamic Workload Console to manage your static and dynamic workload by using a web interface.

By default the Dynamic Workload Console installation process also installs the Z connector component.

The following scenario shows a fresh typical installation of the Dynamic Workload Console at the latest product version.


Installing Liberty
Encrypting passwords (optional)
Creating and populating the database
Installing the DWC server
Defining z/OS engine in Z connector

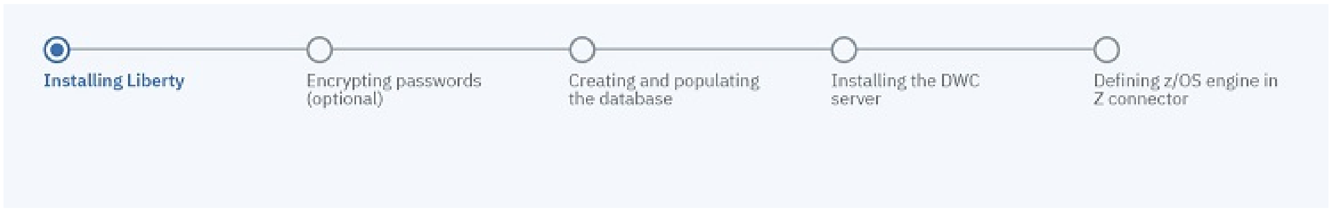
Installing WebSphere Application Server Liberty Base

WebSphere Application Server Liberty Base is required on all workstations where you plan to install the master components and the Dynamic Workload Console.

Before you begin

Ensure that your system meets the operating system and Java requirements. For more information, see WebSphere Application Server Liberty Base detailed system requirements.

About this task


Installing Liberty
Encrypting passwords (optional)
Creating and populating the database
Installing the DWC server
Defining z/OS engine in Z connector

You can quickly install WebSphere Application Server Liberty Base by extracting an archive file on all supported platforms.

Install WebSphere Application Server Liberty Base on all of the following workstations, which comprise a typical installation:

- Two Dynamic Workload Console installations on two separate workstations

On UNIX workstations, you can install WebSphere Application Server Liberty Base using a user of your choice. In this case, assign the HCL Workload Automation administrative user read and write access to the WebSphere Application Server Liberty Base installation directory.

Ensure you install WebSphere Application Server Liberty Base in the `../current` path. This prevents problems when installing a new WebSphere Application Server Liberty Base level. By default, WebSphere Application Server Liberty Base installs in the ../current path.

To extract the archive, you can use your own Java Ext or use the Java Ext provided with the HCL Workload Automation image. The provided Java Ext is located in the `/TWS/JavaExt` folder in the image for your operating system.

To install WebSphere Application Server Liberty Base, perform the following steps:

1. Find out which version of Open Liberty is required, by checking the required version of the Application server in the **Supported Software Report**, available in Product Requirements.
2. Download Open Liberty from Get started with Open Liberty. Download the package named **All GA Features**
3. Perform one of the following actions:
    a. Extract Open Liberty using the root user:

       Windows **On Windows operating systems**

       ```
       unzip <openliberty_download_dir>\openliberty-<version>.zip
        -d <install_dir>
       ```

       UNIX **On UNIX operating systems**

       ```
       unzip <openliberty_download_dir>/openliberty-<version>.zip
        -d <install_dir>
       ```

    b. Run the following command to assign permissions:

       ```
       chmod 755 -R "wlp_directory"
       ```

   OR

   Extract Open Liberty using the user who is going to install the product, as follows:

   ```
   su - "wauser"
   unzip
   ```

   where:

   <openliberty_download_dir>

   The directory where you downloaded Open Liberty.

   install_dir

   The directory where you want to install Open Liberty.

   Note: Install the new Open Liberty in the exact location of the previous WebSphere Application Server Liberty Base installation.

4. Ensure the HCL Workload Automation administrative user has the rights to run Open Liberty and full access to the installation directory. If Open Liberty is shared between the master domain manager and the Dynamic Workload Console, ensure also the Dynamic Workload Console user has the same rights.

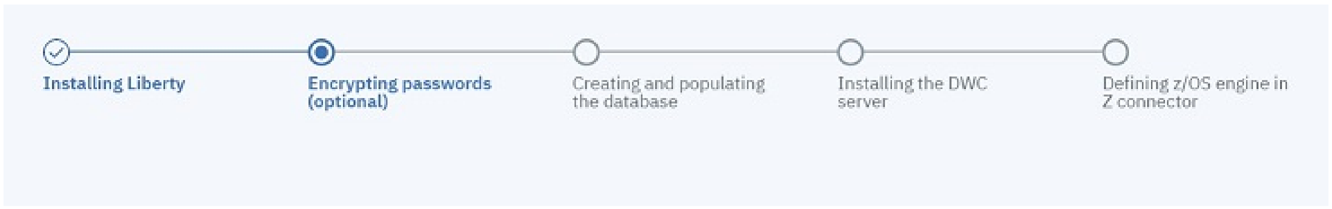
Results

You have now successfully installed WebSphere Application Server Liberty Base. You can proceed to install the Dynamic Workload Console.

Encrypting passwords (optional)

How to encrypt passwords required by the installation, upgrade, and management processes.

About this task


Installing Liberty
Encrypting passwords (optional)
Creating and populating the database
Installing the DWC server
Defining z/OS engine in Z connector

You can optionally encrypt the passwords that you will use while installing, upgrading, and managing HCL Workload Automation. The secure command uses the AES method and prints the encrypted password to the screen or saves it to a file.

Note: It is important you understand the limits to the protection that this method provides. The custom passphrase you use to encrypt the passwords is stored in clear format in the `passphrase_variables.xml` file, stored in `configureDropin`. To fully understand the implications of this method, it is recommended you read the information provided by Open Liberty at the link Password encryption limitations.

You can perform a typical procedure, which uses a custom passphrase, as described in the following scenario. For more information about all secure arguments and default values, see Optional password encryption - secure script on page 284.

Encrytping the password

1. Browse to the folder where the secure command is located:
   - Before the installation, the command is located in the product image directory, `<image_directory>/TWS/<op_sys>/Tivoli_LWA_<op_sys>/TWS/bin`
   - After the installation, the command is located in `TWA_home/TWS/bin`
2. Depending on your operating system, encrypt the password as follows:

   Windows **Windows operating systems**

   ```
   secure -password password -passphrase passphrase
   ```

   UNIX **UNIX operating systems**

   ```
   ./secure -password password -passphrase passphrase
   ```

   z/OS **z/OS operating systems**

   ```
   ./secure -password password -passphrase passphrase
   ```

   where

   -password

   Specifies the password to be encrypted.

   -passphrase

   Specifies the custom passphrase that is used to generate the key with which the command encrypts the password. If you set this parameter, inform the user who installs HCL Workload Automation that they must define the **SECUREWRAP_PASSPHRASE** environment variable in the same shell from which they run the installation command, and set it to the same value as the **passphrase** parameter. On Windows operating systems, the passphrase must be at least 8 characters long. This argument generates a password which can be reused for all HCL Workload Automation components. This parameter is mutually exclusive with the -useaeskeystore on page 287 parameter, which generates a password which can be decrypted only on the local workstation and not reused for other components.

3. Provide both the encrypted password and custom passphrase to the user in charge of installing HCL Workload Automation. You can use encrypted passwords only in association with the specific passphrase used to encrypt them.

Installing with the encrypted password

The user in charge of installing HCL Workload Automation must set the **SECUREWRAP_PASSPHRASE** environment variable by performing the following steps:

1. Open a brand new shell session.
2. Ensure that no value is set for the **SECUREWRAP_PASSPHRASE** environment variable.
3. Define the **SECUREWRAP_PASSPHRASE** environment variable and set it to the passphrase defined by the user who ran the secure command, as follows:

   ```
   SECUREWRAP_PASSPHRASE=<passphrase>
   ```

   You can use encrypted passwords only in association with the specific passphrase used to encrypt them.

4. In the same shell session, provide the encrypted passwords when running any command that uses a password. An encrypted password looks like the following example:

   ```
   {aes}AFC3jj9cROYyqR+3CONBzVi8deLb2Bossb9GGroh8UmDPGikIkzXZzid3nzY0IhnSg=
   ```

Creating and populating the database

Before you install the Dynamic Workload Console for the first time, ensure you have a supported database installed. You can choose to use any of the supported relational database management systems (RDBMS).

Installing Liberty
Encrypting passwords (optional)
Creating and populating the database
Installing the DWC server
Defining z/OS engine in Z connector

Create and populate the database tables for the Dynamic Workload Console by following the procedure appropriate for your RDBMS:

- Creating and populating the database for DB2 for the Dynamic Workload Console on page 222
- Creating and populating the database for DB2 for z/OS for the Dynamic Workload Console on page 224
- Creating the database for Oracle for the Dynamic Workload Console on page 228
- Creating and populating the database for MSSQL for the Dynamic Workload Console on page 230
- Creating and populating the database for MSSQL cloud-based databases for the Dynamic Workload Console on page 232
- Creating and populating the database for PostgreSQL for the Dynamic Workload Console on page 234

Note: If you use Db2 for z/OS with the Dynamic Workload Console version 10.2.4 or later, transfer the drivers in binary mode from the directory where you installed Db2 for z/OS to a directory of your choice. When you run the configuredb or dwcinst script, set the directory you chose in the **dbdriverspath** parameter.

Next, install the Dynamic Workload Console servers, as described in Dynamic Workload Console installation - dwcinst script on page 252.

Creating and populating the database for DB2 for the Dynamic Workload Console

Instructions for creating and populating the Dynamic Workload Console database for DB2.

Before you begin

Ensure a DB2 database is installed.

About this task

Installing Liberty
Encrypting passwords (optional)
Creating and populating the database
Installing the DWC server
Defining z/OS engine in Z connector

You can perform a typical database procedure, as described in the following scenarios, or you can customize the database parameters, as described in the section about FAQ - Database customizations in *HCL Workload Automation: Planning and Installation*.

DB2 requires a specific procedure in which you first create the database and then create and populate the database tables. To simplify the database creation, a customized SQL file named `create_database.sql` is provided containing the specifics for creating the Dynamic Workload Console database. The database administrator can use this file to create the database. After the database has been created, you can proceed to create and populate the database tables.

You can run the configureDb command specifying a typical set of parameters. In this case, default values are used for all remaining parameters.

For more information about all parameters and supported values of the `configureDb` command, see Database configuration - configureDb script on page 237.

Default values are stored in the `configureDb.properties` file, located in *`image_location`*. If you need to modify any of the default values, edit the `configureDb.properties` file, but do not modify the `configureDb.template` file located in the same path.

To create and populate the Dynamic Workload Console database and schema for DB2, perform the following steps:

1. On the workstation where you plan to install the Dynamic Workload Console, extract the Dynamic Workload Console package to a directory of your choice.
2. Browse to the *`image_location`*`/DWC_`*`interp_name`*`/tools` path.
3. Edit the `create_database.sql` file by replacing the default value for the database name (**DWC**) with the name you intend to use.
4. Provide the `create_database.sql` file to the DB2 administrator to run on the DB2 database.
   The following command creates the Dynamic Workload Console database:

   ```
   db2 -tvf <file_location>/create_database.sql
   ```
5. Instruct the DB2 administrator to create the DB2 user on the server hosting the DB2 database. You will then specify this user with the `dbuser` parameter when creating and populating the database with the configureDb command on the Dynamic Workload Console. When you run the configureDb command, this user is automatically granted access to the Dynamic Workload Console tables on the database server.
6. On the server where you plan to install the Dynamic Workload Console,browse to *image_location*/DWC_*interp_name*.
7. Type the following command to create and populate the Dynamic Workload Console database tables with typical settings:

   Windows **On Windows operating systems**

   ```
   cscript configureDb.vbs --rdbmstype DB2 --dbhostname DB_hostname
           --dbport db_port --dbname db_name --dbuser db_user
           --dbadminuser DB_admin_user --dbadminuserpw DB_admin_pwd
   ```

   UNIX **On UNIX operating systems**

   ```
   ./configureDb.sh --rdbmstype DB2 --dbhostname DB_hostname
           --dbport db_port --dbname db_name --dbuser db_user
           --dbadminuser DB_admin_user --dbadminuserpw DB_admin_pwd
   ```

   where:

   --rdbmstype

      The database vendor.

--dbhostname ***db_hostname***

The host name or IP address of database server.

--dbport ***db_port***

The port of the database server.

--dbname ***db_name***

The name of the Dynamic Workload Console database.

--dbuser ***db_user***

The database user you must create before running the configureDb command. When you run the configureDb command, this user is automatically granted access to the Dynamic Workload Console tables on the database server.

--dbadminuser ***db_admin_user***

The database administrator user that creates the Dynamic Workload Console schema objects on the database server.

--dbadminuserpw ***db_admin_password***

The password of the DB administrator user that creates the Dynamic Workload Console schema objects on the database server. Special characters are not supported.

Note: The following parameters specified with the configureDb command are also required when installing theDynamic Workload Console and their values must be the same:

- **--rdbmstype**
- **--dbhostname**
- **--dbport**
- **--dbname**
- **--dbuser**

Results

You have now successfully created and populated the Dynamic Workload Console database.

Creating and populating the database for DB2 for z/OS for the Dynamic Workload Console

Instructions for creating and populating the database for DB2 for z/OS for Dynamic Workload Console.

Before you begin

Ensure a DB2 for z/OS database is installed.

About this task

Installing Liberty
Encrypting passwords (optional)
Creating and populating the database
Installing the DWC server
Defining z/OS engine in Z connector

You can perform a typical database procedure, as described in the following scenarios, or you can customize the database parameters, as described in the section about FAQ - Database customizations in *HCL Workload Automation: Planning and Installation*.

DB2 for z/OS requires a specific procedure in which you first create the database and then create and populate the database tables. To simplify the database creation, a sample JCL named `EQQINDWC` is provided with Package HWAZ_950_APAR_HC00001 containing the specifics for creating the Dynamic Workload Console database. The database administrator can use this file to create the database. After the database has been created, you can proceed to create and populate the database tables.

You can run the **configureDb** command specifying a typical set of parameters. In this case, default values are used for all remaining parameters.

You can optionally configure DB2 in SSL mode on UNIX operating systems by specifying the **sslkeysfolder** and **sslpassword** parameters when you run the configureDb command. For more information, see the topic about using certificates when DB2 or PostgreSQL is in SSL mode in *HCL Workload Automation: Planning and Installation*.

For more information about all parameters and supported values of the `configureDb` command, see Database configuration - configureDb script on page 237.

Default values are stored in the `configureDb.properties` file, located in *`image_location`*.

If you need to modify any of the default values, edit the `configureDb<`*`database_vendor`*`>.properties` file, but do not modify the `configureDb<`*`database_vendor`*`>.template` file located in the same path.

To create and populate the Dynamic Workload Console database and schema for DB2 for z/OS, perform the following steps:

1. From the SEQQSAMP library, edit the `EQQINDWC` sample JCL as required.

   Note: The `EQQINDWC` sample JCL is provided with the Package HWAZ_950_APAR_HC00001. If you did not install this APAR, create a JCL named `EQQINDWC` that looks like the following example:

```
//JOBCARD
//*********************************************************************/
//*                                                                   */
//*  SECURITY CLASSIFICATION:                                         */
//*  Licensed Materials - Property of HCL 5698-T08                  */
//*  Copyright HCL Technologies Ltd. 2020 All rights reserved.       */
//*  US Government Users Restricted Rights - Use, duplication        */
//*  or disclosure restricted by GSA ADP Schedule Contract           */
//*                                                                  */
//* CREATES DB2 STORAGE GROUP AND DATABASE for DWC                   */
//* NOTE1:You must tailor this JCL sample to conform to              */
//*       installation standards defined at your location.           */
//*       - Add a JOB card                                           */
//*       - Change following DB/2 values according to your           */
//*         current environment:                                     */
//*         - DSN.V11R1M0.SDSNLOAD      DB/2 library                 */
//*         - DSN111.RUNLIB.LOAD        DB/2 run library             */
//*         - DBB1                      DB/2 system name             */
//*         - DSNTIA11                  DB/2 DSNTIAD plan name       */
//*         - volname                   volume name                  */
//*         - catname                   catalog name                 */
//*       - Change all the occurrences of                            */
//*         TWSSDWC if you need a storage group with a different name*/
//*                                                                  */
//* Flag Reason   Rlse   Date   Origin Flag Description              */
//* ---- -------- ------ ------ ------ ----------------------------  */
//* $EGE=PH22448  950    200121 ZLIB: DB2 on zLiberty                */
//* $ETA=PH53936  101 220418 MR: EQQINDWC MEMBER OF SEQQSAMP FOR     */
//*                              CREATION OF DB2 DATABASE FOR        */
//*                              DWCFAILS FOR DB2 V12R1M504 OR       */
//*                              higher levels                       */
//******************************************************************/
//EQQINDWC EXEC PGM=IKJEFT01,DYNAMNBR=20
//STEPLIB  DD  DISP=SHR,DSN=DSN.V11R1M0.SDSNLOAD
//SYSTSPRT DD  SYSOUT=*
//SYSTSIN  DD  *
  DSN SYSTEM(DBB1)
  RUN PROGRAM(DSNTIAD) PLAN(DSNTIA11) LIB('DSN111.RUNLIB.LOAD')
//SYSPRINT DD  SYSOUT=*
//SYSUDUMP DD  SYSOUT=*
//SYSIN    DD *
SET CURRENT APPLICATION COMPATIBILITY = 'V10R1';
CREATE STOGROUP TWSSDWC VOLUMES(volname) VCAT catname;
CREATE DATABASE DWC
BUFFERPOOL BP0
INDEXBP BP16K0
STOGROUP TWSSDWC
CCSID UNICODE;
COMMIT;
```

2. Instruct the DB2 for z/OS administrator to create the DB2 for z/OS user on the server hosting the DB2 for z/OS database. You will then specify this user with the `dbuser` parameter when creating and populating the database with the `configureDb` command on the Dynamic Workload Console. When you run the `configureDb` command, this user is automatically granted access to the Dynamic Workload Console tables on the database server.
3. On the server where you plan to install the Dynamic Workload Console, browse to the directory where you extracted the Dynamic Workload Console image.
4. Type the following command to create and populate the Dynamic Workload Console database tables with typical settings:

Windows **On Windows operating systems**

```
cscript configureDb.vbs --rdbmstype DB2Z --dbhostname DB_hostname
--dbport db_port --dbname db_name --dbuser db_user
--dbadminuser DB_admin_user --dbadminuserpw DB_admin_pwd
--zlocationname zOS_location_containing_db --zbufferpoolname buffer_pool_in_zOS_location
```

UNIX **On UNIX operating systems**

```
./configureDb.sh --rdbmstype DB2Z --dbhostname DB_hostname
--dbport db_port --dbname db_name --dbuser db_user
--dbadminuser DB_admin_user --dbadminuserpw DB_admin_pwd
--zlocationname zOS_location_containing_db --zbufferpoolname buffer_pool_in_zOS_location
```

z/OS **On z/OS operating systems**

```
./configureDb.sh --rdbmstype DB2Z --dbhostname DB_hostname
--dbport db_port --dbname db_name --dbuser db_user
--dbadminuser DB_admin_user --dbadminuserpw DB_admin_pwd
--zlocationname zOS_location_containing_db --zbufferpoolname buffer_pool_in_zOS_location
```

where:

--rdbmstype

The database vendor.

--dbhostname *db_hostname*

The host name or IP address of database server.

--dbport *db_port*

The port of the database server.

--dbname *db_name*

The name of the Dynamic Workload Console database.

--dbuser *db_user*

The database user you must create before running the `configureDb` command. When you run the `configureDb` command, this user is automatically granted access to the Dynamic Workload Console tables on the database server.

--dbadminuser *db_admin_user*

The database administrator user that creates the Dynamic Workload Console schema objects on the database server.

--dbadminuserpw *db_admin_password*

The password of the DB administrator user that creates the Dynamic Workload Console schema objects on the database server. Special characters are not supported.

--zlocationname *zos_location_containing_db*

The name of an already existing location in the z/OS environment that will contain the new database. The default value is LOC1.

--zbufferpoolname ***buffer_pool_in zos_location***

The name of an already existing buffer pool created in the location specified by `-zlocationname`. The default value is BP32K.

Note: The following parameters specified with the **configureDb** command are also required when installing the Dynamic Workload Console and their values must be the same:

- **--rdbmstype**
- **--dbhostname**
- **--dbport**
- **--dbname**
- **--dbuser**
- **--zlocationname**

Results

You have now successfully created and populated the Dynamic Workload Console database.

Creating the database for Oracle for the Dynamic Workload Console

Instructions for creating and populating the Dynamic Workload Console database for Oracle.

About this task

Installing Liberty
Encrypting passwords (optional)
Creating and populating the database
Installing the DWC server
Defining z/OS engine in Z connector

You can perform a typical database procedure, as described in the following scenarios, or you can customize the database parameters, as described in the section about FAQ - Database customizations in *HCL Workload Automation: Planning and Installation*.

You can run the **configureDb** command specifying a typical set of parameters. In this case, default values are used for all remaining parameters.

For more information about all parameters and supported values of the `configureDb` command, see Database configuration - configureDb script on page 237.

Default values are stored in the `configureDbOracle.properties` file, located in *`image_location`*. If you need to modify any of the default values, edit the `configureDbOracle.properties` file, but do not modify the `configureDbOracle.template` file located in the same path.

To create and populate the Dynamic Workload Console database, perform the following steps:

1. On the server where you plan to install the Dynamic Workload Console, extract the Dynamic Workload Console package to a directory of your choice.
2. Browse to the directory where you extracted the package.
3. Type the following command to populate the Dynamic Workload Console database with typical settings:

   Windows **On Windows operating systems**

   ```
   cscript configureDb.vbs --rdbmstype ORACLE --dbname service_name
   --dbuser db_user --dbpassword DB_password --dbhostname DB_hostname
   --dbadminuser DB_administrator --dbadminuserpw DB_administrator_password
   --iwstsname USERS
   ```

   UNIX **On UNIX operating systems**

   ```
   ./configureDb.sh --rdbmstype ORACLE --dbname service_name
   --dbuser db_user --dbpassword DB_password --dbhostname DB_hostname
   --dbadminuser DB_administrator --dbadminuserpw DB_administrator_password
   --iwstsname USERS
   ```

   where:

   --rdbmstype

   The database vendor.

   --dbname *db_name*

   The service name of the Dynamic Workload Console database.

   dbuser *db_user*

   The user to be granted access to the Dynamic Workload Console tables on the database server.

   --dbpassword *db_password*

   The password for the user that has been granted access to the Dynamic Workload Console tables on the database server. Special characters are not supported.

   --dbhostname *db_hostname*

   The host name or IP address of database server.

   --dbadminuser *db_admin_user*

   The database administrator user that creates the Dynamic Workload Console schema objects on the database server.

   --dbadminuserpw *db_admin_password*

   The password of the DB administrator user that creates the Dynamic Workload Console schema objects on the database server. Special characters are not supported.

--iwstsname|-tn ***table_space_name***

The name of the tablespace for Dynamic Workload Console data. This parameter is required.

Note: The following parameters specified with the configureDb command are also required when installing the Dynamic Workload Console and their values must be the same:

- **rdbmstype**
- **dbhostname**
- **dbport**
- **dbname**
- **dbuser**
- **dbpassword**

Results

You have now successfully created and populated the Dynamic Workload Console database.

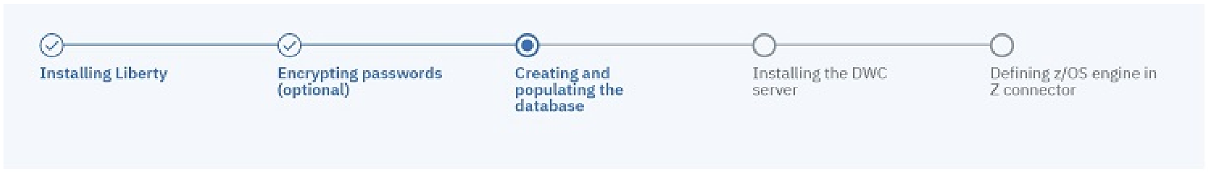
What to do next

You can now proceed to Installing a Dynamic Workload Console server on page 247.

Creating and populating the database for MSSQL for the Dynamic Workload Console

Instructions for creating and populating the Dynamic Workload Console database for MSSQL.

About this task


Installing Liberty
Encrypting passwords (optional)
Creating and populating the database
Installing the DWC server
Defining z/OS engine in Z connector

You can perform a typical database procedure, as described in the following scenarios, or you can customize the database parameters, as described in the section about FAQ - Database customizations in *HCL Workload Automation: Planning and Installation*.

You can run the **configureDb** command specifying a typical set of parameters. In this case, default values are used for all remaining parameters.

For more information about all parameters and supported values of the `configureDb` command, see Database configuration - configureDb script on page 237.

Default values are stored in the `configureDbMSSQL.properties` file, located in *`image_location`*.

To create the Dynamic Workload Console database and schema, perform the following steps:

1. Windows Only on Windows systems hosting an MSSQL database, create the path for hosting the following tablespace, if the path is not already existing:
    - TWS_DATA
2. Only on Windows systems hosting an MSSQL database, specify the path to the folder when running the configureDb.vbs command or when filling in the `configureDbMSSQL.properties` properties file with the following parameter:
    - --iwstspath
3. On the server where you plan to install the Dynamic Workload Console, extract the Dynamic Workload Console package to a directory of your choice.
4. To populate the Dynamic Workload Console database with typical settings, type the following command:

    Windows **On Windows operating systems**

    ```
    cscript configureDb.vbs --rdbmstype MSSQL --dbname db_name
    --dbhostname db_hostname --dbadminuser db_administrator
    --dbadminuserpw db_administrator_password
    --iwstspath DATA_tablespace_path
    ```

    UNIX **On UNIX operating systems**

    ```
    ./configureDb.sh --rdbmstype MSSQL --dbname db_name
    --dbhostname db_hostname --dbadminuser db_administrator
    --dbadminuserpw db_administrator_password
    --iwstspath DATA_tablespace_path
    ```

    where:

    --rdbmstype

    The database vendor.

    --dbname *db_name*

    The name of the Dynamic Workload Console database.

    --dbhostname *db_hostname*

    The host name or IP address of database server.

    --dbadminuser *db_admin_user*

    The database administrator user that creates the Dynamic Workload Console schema objects on the database server.

    --dbadminuserpw *db_admin_password*

    The password of the DB administrator user that creates the Dynamic Workload Console schema objects on the database server. Special characters are not supported.

--iwstspath|-tp *table_space*

The path of the tablespace for HCL Workload Automation or Dynamic Workload Console data. This parameter is optional. The default value for all databases other than Oracle is:

For all operating systems, except z/OS

TWS_DATA

For z/OS operating system

TWSDATA

Only on Windows systems hosting an MSSQL database, ensure the folder for the tablespace is already existing before running the configureDb command and specify the path using this parameter. Specify the path using forward slashes (/), for example: `c:/<my_path>/TWS_DATA`.

Note: The following parameters specified with the configureDb command are also required when installing the Dynamic Workload Console and their values must be the same:

- **rdbmstype**
- **dbhostname**
- **dbport**
- **dbname**

When **--rdbmstype** is set to `MSSQL`, the default value is **sa**. To install a Dynamic Workload Console with a user different from **sa**, you must create a new user in `MSSQL` and grant all the required permissions before running the configureDb command.

Results

You have now successfully created and populated the Dynamic Workload Console database.

What to do next

You can now proceed to Installing a Dynamic Workload Console server on page 247.

Creating and populating the database for MSSQL cloud-based databases for the Dynamic Workload Console

Instructions for creating and populating the Dynamic Workload Console database for MSSQL cloud-based databases.

About this task

Installing Liberty
Encrypting passwords (optional)
Creating and populating the database
Installing the DWC server
Defining z/OS engine in Z connector

MSSQL cloud-based databases include the following:

- Azure SQL
- Google Cloud SQL for SQL server
- Amazon RDS for MSSQL

You can perform a typical database procedure, as described in the following scenarios, or you can customize the database parameters, as described in the section about FAQ - Database customizations in *HCL Workload Automation: Planning and Installation*.

You can run the **configureDb** command specifying a typical set of parameters. In this case, default values are used for all remaining parameters.

For more information about all parameters and supported values of the `configureDb` command, see Database configuration - configureDb script on page 237.

If you need to modify any of the default values, edit the `configureDbMSSQL.properties` file, but do not modify the `configureDbMSSQL.template` file located in the same path. Default values are stored in the `configureDbMSSQL.properties` file, located in *`image_location`*.

To create the Dynamic Workload Console database and schema, perform the following steps:

1. Specify the path to the folder when running the configureDb command or when filling in the `configureDbMSSQL.properties` properties file with the following parameter:
   - --iwstsname PRIMARY
2. On the server where you plan to install the Dynamic Workload Console, extract the Dynamic Workload Console package to a directory of your choice.
3. To populate the Dynamic Workload Console database with typical settings, type the following command:

   Windows **On Windows operating systems**

   ```
   cscript configureDb.vbs --rdbmstype MSSQL --dbname db_name
   --dbhostname db_hostname --dbadminuser db_administrator
   --dbadminuserpw db_administrator_password
   --iwstsname PRIMARY
   ```

iwstsname *DATA_tablespace_name*

The name of the tablespace for Dynamic Workload Console data. The default value is PRIMARY. Do not modify this value.

Note: The following parameters specified with the **configureDb** command are also required when installing the Dynamic Workload Console and their values must be the same:

- **rdbmstype**
- **dbhostname**
- **dbport**
- **dbname**
- **dbuser**

Results

You have now successfully created and populated the Dynamic Workload Console database.

What to do next

You can now proceed to Installing a Dynamic Workload Console server on page 247.

Creating and populating the database for PostgreSQL for the Dynamic Workload Console

Instructions for creating and populating the Dynamic Workload Console database for PostgreSQL.

Before you begin

Ensure you have performed the following tasks:

- Create the PostgreSQL database and ensure it is configured to allow remote connections. To create the database, use the following command:

  ```
  create database <database_name> with lc_collate='C' template=template0;
  ```

  This command creates the database with the collation feature enabled.
- Create a user dedicated specifically to the new database schema and do not use the administrator user (`postgres`) for this purpose.

For more information about allowing remote connections and creating users, see the PostgreSQL documentation.

About this task

Installing Liberty
Encrypting passwords (optional)
Creating and populating the database
Installing the DWC server
Defining z/OS engine in Z connector

You can perform a typical database procedure, as described in the following scenarios, or you can customize the database parameters, as described in the section about FAQ - Database customizations in *HCL Workload Automation: Planning and Installation*.

You can optionally configure PostgreSQL in SSL mode on UNIX operating systems by specifying the **sslkeysfolder** and **sslpassword** parameters when you run the configureDb command. For more information, see the FAQ about How can I use default certificates when DB2 or PostgreSQL is in SSL mode? in *HCL Workload Automation: Planning and Installation*.

You can run the configureDb command specifying a typical set of parameters. In this case, default values are used for all remaining parameters.

For more information about all parameters and supported values of the `configureDb` command, see Database configuration - configureDb script on page 237.

Default values are stored in the `configureDbPostgresql.properties` file, located in *`image_location`*. If you need to modify any of the default values, edit the `configureDbPostgresql.properties` file, but do not modify the `configureDbPostgresql.template` file located in the same path. For an example of a properties file, see What is the content of a database properties file?.

To create and populate the Dynamic Workload Console database and schema for PostgreSQL, perform the following steps:

1. On the workstation where you plan to install the Dynamic Workload Console, extract the Dynamic Workload Console package to a directory of your choice.
2. Browse to the path *`image_location`*`/TWS/`*`interp_name`*.
3. Type the following command to create and populate the Dynamic Workload Console database tables with typical settings:

   Windows **On Windows operating systems**

   ```
   cscript configureDb.vbs --rdbmstype POSTGRESQL --dbhostname DB_hostname
   --dbport db_port --dbname db_name --dbuser db_user
   --dbadminuser DB_admin_user --dbadminuserpw DB_admin_pwd
   ```

UNIX **On UNIX operating systems**

```
./configureDb.sh --rdbmstype POSTGRESQL --dbhostname DB_hostname
--dbport db_port --dbname db_name --dbuser db_user
--dbadminuser DB_admin_user --dbadminuserpw DB_admin_pwd
```

where:

--rdbmstype

The database vendor.

--dbhostname *db_hostname*

The host name or IP address of database server.

--dbport *db_port*

The port of the database server.

--dbname *db_name*

The name of the Dynamic Workload Console database.

--dbuser *db_user*

The database user you must create before running the configureDb command. When you run the configureDb command, this user is automatically granted access to the Dynamic Workload Console tables on the database server.

--dbadminuser *db_admin_user*

The database administrator user that creates the Dynamic Workload Console schema objects on the database server.

--dbadminuserpw *db_admin_password*

The password of the DB administrator user that creates the Dynamic Workload Console schema objects on the database server. Special characters are not supported.

Note: The following parameters specified with the configureDb command are also required when installing theDynamic Workload Console and their values must be the same:

- **--rdbmstype**
- **--dbhostname**
- **--dbport**
- **--dbname**
- **--dbuser**

Results

You have now successfully created and populated the Dynamic Workload Console database.

What to do next

You can now proceed to Installing a Dynamic Workload Console server on page 247.

Database configuration - configureDb script

The configureDb script is typically used by the database administrator for creating and populating the HCL Workload Automation database.

For typical scenarios, see Creating and populating the database on page 221.

This section lists and describes the parameters that you can use to create and populate the HCL Workload Automation database.

You can specify values in the properties file, type them in the command line, or use both methods. If a parameter is specified both in the properties file and in the command line, the command line value takes precedence.

The log files generated from this command are located in the following path:

Windows On Windows operating systems

DWC_home`\logs`

UNIX On UNIX operating systems

DWC_DATA_dir`/installation/logs`

z/OS On z/OS operating systems

DWC_DATA_dir`/installation/logs`

Windows

Syntax for Windows operating systems

Show command usage

```
configureDb -? | --usage | --help
```

Retrieve the command parameters and values from a file

```
configureDb --propfile | -f  [property_file]
```

General information

```
[--lang lang_id]
[--work_dir working_directory]
[--wlpdir wlp_directory]
[--componenttype MDM | DDM | DWC ]
[--dbadminuser db_admin_user]
--dbadminuserpw db_admin_password
--rdbmstype|-r DB2 | DB2Z | ORACLE | MSSQL | POSTGRESQL
[--dbname db_name]
[--dbuser db_user]
[--dbport db_port]
--dbhostname db_hostname
```

```
        [--dbdriverpath db_driver_path]
        --auth_type aythentication_type ]
        [--iwstsname table_space_name]
        [--iwstspath table_space_path]
        [--iwslogtsname log_table_space]
        [--iwslogtspath log_path_table_space]
        [--iwsplantsname plan_table_space]
        [--iwsplantspath plan_path_table_space]
        [--execsql execute_sql]
```

Oracle-only configuration options

```
        --dbpassword db_password
        [--usePartitioning true | false ]
        [--Usage_TsTempName IWS_temp_path]
        [--skipdbcheck true | false]
```

Db2 for z/OS-only configuration options

```
        [--zlocationname zOS_location_containing_db]
        [--zbufferpoolname buffer_pool_in_zOS_location]
```

UNIX

Syntax for UNIX operating systems

Show command usage

```
configureDb -? | --usage | --help
```

Retrieve the command parameters and values from a file

```
configureDb --propfile | -f  [property_file]
```

General information

```
        [--lang lang_id]
        [--work_dir working_directory]
        [--wlpdir wlp_directory]
        [--componenttype MDM | DDM | DWC ]
        [--dbadminuser db_admin_user]
        --dbadminuserpw db_admin_password
        --rdbmstype|-r DB2 | DB2Z | ORACLE | MSSQL  | POSTGRESQL
        [--dbname db_name]
        [--dbuser db_user]
        [--dbport db_port]
        --dbhostname db_hostname
        [--dbdriverpath db_driver_path]
        [--iwstsname table_space_name]
        [--iwstspath table_space_path]
        [--iwslogtsname log_table_space]
        [--iwslogtspath log_path_table_space]
        [--iwsplantsname plan_table_space]
        [--iwsplantspath plan_path_table_space]
        [--execsql execute_sql ]
```

Oracle-only configuration options

```
        --dbpassword db_password
        [--usePartitioning true | false ]
        [--Usage_TsTempName IWS_temp_path]
        [--skipdbcheck true | false]
```

Db2- and PostgreSQL-only security options

```
    [--sslkeysfolder  keystore_truststore_folder]
    [--sslpassword ssl_password]
    [--dbsslconnection true | false]
```

Db2 for z/OS-only configuration options

```
        [--zlocationname zOS_location_containing_db]
        [--zbufferpoolname buffer_pool_in_zOS_location]
```

z/OS

Syntax for z/OS operating system

Show command usage

```
configureDb -? | --usage | --help
```

Retrieve the command parameters and values from a file

```
configureDb --propfile | -f  [properties_file]
```

General information

```
        [--lang lang_id]
        [--work_dir working_directory]
        [--wlpdir wlp_directory]
        [--dbadminuser db_admin_user]
        [--componenttype DWC ]
        --dbadminuserpw db_admin_password
        --rdbmstype|-r DB2 | DB2Z | ORACLE | MSSQL | POSTGRESQL
        [--dbname db_name]
        [--dbuser db_user]
        [--dbport db_port]
        --dbhostname db_hostname
        [--dbdriverpath db_driver_path]
        [--iwstsname table_space_name]
        [--iwstspath table_space_path]
        [--iwslogtsname log_table_space]
        [--iwslogtspath log_path_table_space]
        [--iwsplantsname plan_table_space]
        [--iwsplantspath plan_path_table_space]
        [--execsql execute_sql ]
```

Db2 for z/OS-only configuration options

```
        [--zlocationname zOS_location_containing_db]
        [--zbufferpoolname buffer_pool_in_zOS_location]
```

Database configuration parameters

-? | --usage | --help

Displays the command usage and exits.

--propfile|-f [*properties_file*]

Optionally specify a properties file containing custom values for `configureDb` parameters. The default file for the Dynamic Workload Console is `image_location/configureDbdatabase_vendor.properties`. Specifying a properties file is suggested if you have a high number of parameters which require custom values. You can also reuse the file with minimal modification for several installations. If you create a custom properties file, specify its name and path with the **-f** parameter.

--lang *lang_id*

The language in which the messages returned by the command are displayed. If not specified, the system LANG is used. If the related catalog is missing, the default C language catalog is used. If neither **--lang** nor LANG are used, the default codepage is set to SBCS. For a list of valid values for these variables, see the following table:

Table 33. Valid values for -lang and LANG parameter

Language	Value
Brazilian Portuguese	pt_BR
Chinese (traditional and simplified)	zh_CN, zh_TW
English	en
French	fr
German	de
Italian	it
Japanese	ja
Korean	ko
Russian	ru
Spanish	es

Note: This is the language in which the installation log is recorded and not the language of the installed component instance. The command installs all languages as default.

--work_dir

The working directory where you extract the installation image. It also contains the output produced by the command, such as the SQL statements if you set the **execsql** parameter to **false**. The default value is `/tmp` on UNIX operating systems and `C:\tmp` on Windows operating systems.

[--wlpdir *wlp_directory*]

The path to Open Liberty installation directory. Open Liberty is used to decrypt the passwords you provide in encrypted form. This parameter is required only if you encrypt your passwords with the {**xor**} or {**aes**} encoding.

--componenttype MDM | DDM | DWC

The HCL Workload Automation component for which the database is installed. This parameter is optional. Supported values are:

MDM

master domain manager. Applies only to distributed operating systems.

DDM

dynamic domain manager. Applies only to distributed operating systems.

DWC

Dynamic Workload Console (it comprises the Federator).

--dbadminuser *db_admin_user*

The database administrator user who creates the HCL Workload Automation or Dynamic Workload Console schema objects on the database server. This parameter is optional. Depending on the database vendor, the default values are as follows:

db2admin

when **--rdbmstype** is set to `DB2`

sysadm

when **--rdbmstype** is set to `DB2Z`

system

when **--rdbmstype** is set to `ORACLE`

sa

when **--rdbmstype** is set to `MSSQL`

--dbadminuserpw *db_admin_password*

The password for the DB administrator user who creates the HCL Workload Automation schema objects on the database server. This parameter is required. You can optionally encrypt the password. For more information, see Encrypting passwords (optional) on page 219.

--rdbmstype|-r *rdbms_type*

The database type. Supported databases are:

- **DB2**
- **ORACLE**
- **MSSQL** This value applies to MSSQL and supported MSSQL cloud-based databases.
- **POSTGRESQL**
-

This parameter is required and has no default value.

--dbname *db_name*

The name of the Dynamic Workload Console database. This parameter is optional and case sensitive. Depending on the component that you are installing and the database vendor, the default values are as follows:

TDWC

when **--rdbmstype** is set to `DB2`

TDWC

when **--rdbmstype** is set to `DB2Z`

orcl

when **--rdbmstype** is set to `ORACLE`

TDWC

when **--rdbmstype** is set to `MSSQL`

TDWC

when **--rdbmstype** is set to `POSTGRES`

--dbuser *db_user*

The database user that has been granted access to the HCL Workload Automation or Dynamic Workload Console tables on the database server. This parameter is optional. The default values are as follows:

db2dwc

when **--rdbmstype** is set to `DB2`

root

when **--rdbmstype** is set to `DB2Z`

twsora

when **--rdbmstype** is set to `ORACLE`

sa

when **--rdbmstype** is set to `MSSQL`

--dbport *db_port*

The port of the database server. This parameter is optional. Depending on the database vendor, the default values are as follows:

50000

when **--rdbmstype** is set to `DB2`

446

when **--rdbmstype** is set to `DB2Z`

1521

when **--rdbmstype** is set to `ORACLE`

1433

when **--rdbmstype** is set to `MSSQL`

--dbhostname *db_hostname*

The host name or IP address of database server. This parameter is required. If you are configuring the database in SSL mode, ensure you specify the fully qualified hostname used in the database client CN. This value is the same as the fully qualified hostname of the workstation where the database is installed.

--dbdriverpath *db_driver_path*

The path where the database drivers are stored. This parameter is optional, but becomes required when you set the **rdbmstype** parameter to **DB2Z**. If you use Db2 for z/OS with the Dynamic Workload Console version 10.2.4 or later, transfer the drivers in binary mode from the directory where you installed Db2 for z/OS to a directory of your choice. Specify the driver path using this parameter.

By default, the configuration script references the JDBC drivers supplied with the product images. If your database server is not compatible with the supplied drivers, then contact your database administrator for the correct version to use with your database server and specify the driver path using this parameter. Ensure that you provide the same path in the configureDb, serverinst, and dwcinst commands.

--iwstsname|-tn *table_space_name*

The name of the tablespace for HCL Workload Automation data. This parameter is optional for all databases with the exception of the Oracle database. The default value for all databases other than Oracle is:

For all operating systems, except z/OS

TWS_DATA

For z/OS operating system

TWSDATA

--iwstspath|-tp *table_space*

The path of the tablespace for HCL Workload Automation data. This parameter is optional. The default value for all databases other than Oracle is:

For all operating systems, except z/OS

TWS_DATA

For z/OS operating system

TWSDATA

Only on Windows systems hosting an MSSQL database, ensure the folder for the tablespace is already existing before running the configureDb command and specify the path using this parameter. Specify the path using forward slashes (/), for example: `c:/<my_path>/TWS_DATA`.

--iwslogtsname|-ln *log_table_space*

The name of the tablespace for HCL Workload Automation log. This parameter is optional for all databases with the exception of the Oracle database. The default value for all databases other than Oracle is **TWS_LOG**. This parameter applies only to the server components.

--iwslogtspath|-lp *log_path_table_space*

The path of the tablespace for HCL Workload Automation log. This parameter is optional. The default value for all databases other than Oracle is **TWS_LOG**. This parameter applies only to the server components. Only on Windows systems hosting an MSSQL database, ensure the folder for the tablespace is already existing before running the configureDb command and specify the path using this parameter. Specify the path using forward slashes (/), for example: `c:/<my_path>/TWS_LOG`.

--iwsplantsname|-pn *plan_table_space*

The name of the tablespace for HCL Workload Automation plan. This parameter is optional for all databases with the exception of the Oracle database. The default value for all databases other than Oracle is **TWS_PLAN**. This parameter applies only to the server components.

--iwsplantspath|-pp *plan_path_table_space*

The path of the tablespace for HCL Workload Automation plan. This parameter is optional. The default value for all databases other than Oracle is **TWS_PLAN**. This parameter applies only to the server components.

Only on Windows systems hosting an MSSQL database, ensure that the folder for the tablespace is already existing before running the configureDb command and specify the path using this parameter. Specify the path using forward slashes (/), for example: `c:/<my_path>/TWS_PLAN`.

--execsql|-es *execute_sql*

Set to **true** to generate and run the SQL file, set to **false** to generate the SQL statement without running it. The resulting files are stored in the path defined in the **--work_dir** parameter. This option is useful if you wan to review the file before running it. This parameter is optional. The default value is **true**.

Windows **--auth_type**

This parameter applies only to Windows operating systems. Specify the authentication type. Supported values are as follows:

SQLSERVER

Enables MSSQL authentication type. Only the user specified with the **--dbadminuser** parameter has the grants to administer the HCL Workload Automation or Dynamic Workload Console database.

WINDOWS

Enables Windows authentication type. The Windows user you used to log on to the workstation is assigned the grants to administer the HCL Workload Automation or Dynamic Workload Console database.

The default value is **SQLSERVER**.

Oracle-only configuration syntax

--dbpassword *db_password*

The password for the user that has been granted access to the HCL Workload Automation or Dynamic Workload Console tables on the database server. This parameter is required only if you are using an Oracle database. You can optionally encrypt the password. For more information, see Encrypting passwords (optional) on page 219.

--usePartitioning

Only applies when installing the master domain manager. Set to **true** if you want to use the Oracle partitioning feature, otherwise set it to **false**. This parameter is optional. The default value is **true**.

--Usage_TsTempName *IWS_temp_path*

Only applies when installing the master domain manager. The path of the tablespace for HCL Workload Automation temporary directory. This parameter is optional. The default value is **TEMP**.

--skipdbcheck

This parameter specifies whether the check on the existence of the Workload Automation schema for the Oracle user is performed or not. By default, the parameter is set to **false** and a check is performed on the Oracle user. If the user does not exist, the script then proceeds to create the user and the Workload Automation schema.

If you have already created your Oracle user, set this parameter to **true**. As a result, the check is skipped and the schema creation is performed also if the Oracle user is already existing.

This parameter is optional.

UNIX **DB2- and PostgreSQL-only configuration syntax**

--sslkeysfolder ***keystore_truststore_folder***

The name and path of the folder containing certificates in PEM format. The installation program automatically processes the keystore and truststore files using the password you specify with the **--sslpassword** parameter. The folder must contain the following files:

- **ca.crt**

  The Certificate Authority (CA) public certificate. Note that if certificates being installed are part of a chain consisting of 3 or more certificates (one Root CA, followed by one or more Intermediate CAs, followed by the end user certificate), then this file must contain the Root CA certificate only. Any Intermediate CA certificates must be stored in the `additionalCAs` subfolder, which therefore becomes a mandatory subfolder. Each Intermediate CA must be stored in the `additionalCAs` subfolder in its own file.

  Note: From V10.2.3, if certificates being installed are part of a chain, the ca.crt can contain also the intermediate CAs. In this case, it must begin with one or more intermediate CA certificates and end with the Root ca.

- **tls.key**

  The private key of the end user certificate for the instance to be installed.

- **tls.crt**

  The public part of the previous key, that is the end user certificate.

For UNIX systems, ensure that all the files have the ownership of the user who installed the master domain manager and the correct permissions (644).

You can optionally create a subfolder to contain one or more `*.crt` files to be added to the server truststore as trusted CA, whose name must be `additionalCAs`. This can be used for example to add to the list of trusted CAs the certificate of the LDAP server or DB2 server. Additionally, you can store here any intermediate CA certificate to be added to the truststore. The subfolder must be named **additionalCAs**. Note that if the end user certificate being installed in the instance is part of a chain consisting of 3 or more certificates (one Root CA, followed by one or more Intermediate CAs, followed by the end user certificate), then the Intermediate CAs certificates must be stored in the `additionalCAs` subfolder, which therefore becomes a mandatory subfolder. Each Intermediate CA must be stored in the `additionalCAs` subfolder in its own file.

For further information about how to generate custom certificates, see the topic about managing certificates using Certman in *HCL Workload Automation: Planning and Installation*.

--sslpassword ***ssl_password***

The password for the certificates.

For more information, see sslkeysfolder on page 246.

You can optionally encrypt the password using the secure script. For more information, see Optional password encryption - secure script on page 284.

UNIX **--dbsslconnection**

UNIX Specify whether you want to enable SSL connection to the database. Supported values are `true` and `false`. The default value is `false`. If you set this parameter to `true`, the **sslkeysfolder** and **sslpassword** parameters become mandatory. Ensure the database client certificate specifies, as its Common Name (CN), the fully qualified hostname of the workstation where the database is installed.

DB2 for z/OS-only configuration syntax

--zlocationname *zos_location_containing_db*

The name of an already existing location in the z/OS environment that will contain the new database. The default value is **LOC1**.

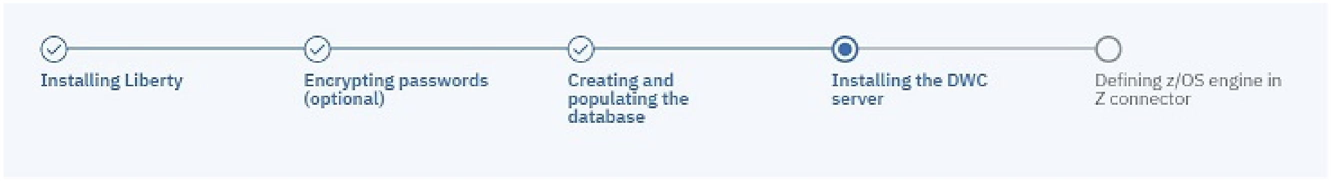
--zbufferpoolname *buffer_pool_in zos_location*

The name of an already existing buffer pool created in the location specified by `-zlocationname`. The default value is BP32K.

Installing a Dynamic Workload Console server

You can choose between installing a Dynamic Workload Console version 10.2.1 or later, or a Dynamic Workload Console version 10.2.0 or earlier. The main differences among the versions are described hereafter.

About this task


Installing Liberty
Encrypting passwords (optional)
Creating and populating the database
Installing the DWC server
Defining z/OS engine in Z connector

Dynamic Workload Console version 10.2.1, or later

SSL configuration is required and is provided with the custom certificates generated during the installation process. For more information, see Installing a Dynamic Workload Console version 10.2.1, or later on page 248.

Note: If you use Db2 for z/OS with the Dynamic Workload Console version 10.2.4 or later, transfer the drivers in binary mode from the directory where you installed Db2 for z/OS to a directory of your choice. When you run the configuredb or dwcinst script, set the directory you chose in the **dbdriverspath** parameter.

Note: If you are installing the Dynamic Workload Console version 10.2.3 or later, the Federator is also automatically installed. This component enables you to monitor your objects through the Orchestration Monitor page of the Dynamic Workload Console. For detailed information about how to configure and use the Federator, see Mirroring the z/OS current plan to enable the Orchestration Monitorthe section about mirroring the z/OS current plan to enable the Orchestration Monitor in the *Dynamic Workload Console User's Guide*.

.

Dynamic Workload Console version 10.2.0, or earlier

SSL configuration is optional. For more information, see Installing Dynamic Workload Console version 10.2.0, or earlier on page 251.

Installing a Dynamic Workload Console version 10.2.1, or later

About this task

The HCL Workload Automation administrator installs the Dynamic Workload Console by running the **dwcinst** command. You can run the **dwcinst** specifying a typical set of parameters. In this case, default values are used for all remaining parameters.

Default values are stored in the `dwcinst.properties` file, located in the root directory of the installation image. If you need to modify any of the default values, edit the `dwcinst.properties` file, but do not modify the `dwcinst.template` file located in the same path.

You have to configure your environment in SSL mode, by using the **sslkeysfolder** and **sslpassword** parameters. These parameters process automatically the certificates for each workstation in your environment. For details about these parameters, see the topic about Dynamic Workload Console installation - dwcinst script in *HCL Workload Automation: Planning and Installation*, version 10.2.1.

To install the Dynamic Workload Console, perform the following steps:

1. Generate a z/OS certificate.

   Note: The owner of the SAF key ring (that is, the user ID who runs the EQQINCRT sample JCL located in SEQQSAMP) must be the same user who runs the controller and server started tasks.

2. Export the z/OS certificate by using the RACF utility panel RACF - SERVICES OPTION MENU (options 7.1.4) to a data set that you have previously allocated. For example:

```
Data Set Name . . . . : <dataset_name>
 Space units . . . . . TRKS
 Primary quantity  . . 5
 Secondary quantity    1
 Directory blocks  . . 0
 Record format . . . . VB
 Record length . . . . 84
 Block size  . . . . . 27998
```

3. Generate the distributed certificates on a distributed environment as follows.

If the certification authority (CA) does not exist

Run the following commands:

```
openssl genrsa -out ca.key 4096
```

```
openssl req -x509 -new -nodes -key ca.key -subj "/CN=<common_name>" -days <dddd> -out ca.crt
```

```
openssl genrsa -des3 -out tls.key 4096
```

```
openssl req -new -key tls.key -out tls.csr
```

```

openssl x509 -req -in tls.csr -CA ca.crt -CAkey ca.key
-CAcreateserial -out tls.crt -extfile /etc/pki/tls/openssl.cnf -extensions v3_req
```

If the certification authority (CA) already exists

a. Run the following command to generate the private key and unsigned certificate:
   - `openssl genrsa -des3 -out tls.key 4096`
   - `openssl req -new -key tls.key -out tls.csr`
b. Send the `tls.csr` file to your certificate administrator to have it signed with your CA and be resent to you.

The `ca.crt, tls.crt`, and `tls.key` files are now ready for use.

4. Store the resulting `ca.crt`, `tls.crt`, and `tls.key` files in a folder of your choice on the workstation where you plan to install the Dynamic Workload Console.
5. Import the certificate from the z/OS to the distributed environment in ASCII mode.
6. Convert the certificates to the p12 format, as follows:
   a. Navigate to the folder where you stored the `ca.crt`, `tls.key`, and `tls.crt` files.
   b. Issue the following command:

   ```
   openssl pkcs12 -export -out TWSServerTrustFile.p12 -inkey tls.key -in tls.crt -certfile ca.crt
   ```

   You are prompted to enter the password used for generating the certificates, and verify it.
   c. Issue the following command:

   ```
   openssl pkcs12 -export -out TWSServerKeyFile.p12 -inkey tls.key -in tls.crt -certfile ca.crt
   ```

   You will be prompted to insert the same password of the creation of the certificates, and verify it.
   d. Now that you have created the databases, issue the following commands to add the z/OS certificates into them:

   ```
   keytool -importcert -keystore TWSServerTrustFile.p12 -trustcacerts -alias <alias>
   -keypass <key_pass> -storepass <store_pass> -file /path/to/zos/certificate -storetype PKCS12
   -keyalg RSA
   ```

   ```
   keytool -importcert -keystore TWSServerKeyFile.p12 -trustcacerts -alias <alias>
   -keypass <key_pass> -storepass <store_pass> -file /path/to/zos/certificate -storetype PKCS12
   -keyalg RSA
   ```

   If prompted, confirm the insertion of the certificates.
   e. To display a list of all the certificates that are stored in the databases, issue the following commands:

```
keytool -list -keystore TWSServerTrustFile.p12 -storetype PKCS12 -storepass <store_pass>
```

```
keytool -list -keystore TWSServerKeyFile.p12 -storetype PKCS12 -storepass <store_pass>
```

7. Import the `ca.crt` and `tls.crt` certificates from the distributed to the z/OS environment in ASCII mode and add them to a data set that you have previously allocated.
8. Import the distributed certificate in the RING created in step 1 on page 248.
9. Transfer the certificate database created in step 6 to z/OS, as follows:

```
zip -q certs.zip TWSServerKeyFile.p12 TWSServerTrustFile.p12 //to create a zip file with the
 certificates

ftp               //to open ftp connection
open <z/OS_ip_address>       //to connect to a specific mainframe ip
<user_id>         //TSO user enabled to OMVS
<passwd>          //TSO password for the user enabled to OMVS
bin          //to chose the right format for the transfer
put certs.zip /path/to/certs/certs.zip       //to actually transfer the zip file inside uss path
close             //to close the ftp connection
quit              //to go back to terminal

From OMVS launch the following commands:

cd /path/to/certs/          //to navigate to the unzipped certs file
unzip -q certs.zip /path/to/certificates_unzipped      //to unzip the files in a specific directory
```

10. When running the dwcinst script, specify the path to the database with the **sslkeysfolder** parameter. This folder must contain two files named `TWSServerKeyFile.p12` and `TWSServerTrustFile.p12`, in which your certificates are already stored.
11. Start the installation specifying a typical set of parameters. Specify the user defined in step 1 in the **--user** parameter. Default values are used for all remaining parameters:

Windows On Windows operating systems

```
cscript dwcinst.vbs --acceptlicense yes --rdbmstype db_type
--user dwc_admin_user --password dwc_pwd --dbname db_name
--dbuser db_user --dbpassword db_pwd --dbhostname db_hostname
--dbport db_port --wlpdir Liberty_installation_dir\wlp
--sslpassword keystore_password --sslkeysfolder distributed_certificates_path
```

UNIX On UNIX operating systems

```
./dwcinst.sh --acceptlicense yes --rdbmstype db_type
--user dwc_admin_user --password dwc_pwd --dbname db_name
--dbuser db_user --dbpassword db_pwd --dbhostname db_hostname
--dbport db_port --wlpdir Liberty_installation_dir/wlp
--sslpassword keystore_password --sslkeysfolder distributed_certificates_path
```

z/OS On z/OS operating system

```
./dwcinst.sh --acceptlicense yes --rdbmstype DB2z
--user non-root_user --password dwc_pwd --dbname db_name
--dbuser db_user --dbpassword db_pwd --dbhostname db_hostname
--dbport db_port --wlpdir Liberty_installation_dir/wlp
--zlocationname zOS_location_containing_db
--sslpassword keystore_password --sslkeysfolder distributed_certificates_path
```

where:

user

The administrator of the Dynamic Workload Console. This user is added to the group of the Dynamic Workload Console administrators at installation time. You can use this account to log in to the Dynamic Workload Console and manage your environment.

password

The password of the Dynamic Workload Console user.

sslpassword

The keystore password.

sslkeysfolder

The name and path of the folder containing the distributed certificates.

12. Stop and start WebSphere Application Server Liberty Base for the Dynamic Workload Console, as described in Starting and stopping the WebSphere Application Server Liberty Base on page 266.
13. In the TCPOPTS statement of the server started task, set the following parameters:
    - SSLKEYSTORETYPE(SAF)
    - SSLKEYSTORE(MDMCSTMRING)
    - SSLLEVEL(FORCE)
14. In the `connectionFactory.xml` file, set `useSsl="true"`.

For more information about all **dwcinst** parameters and default values, see the topic about Dynamic Workload Console installation - dwcinst script in *HCL Workload Automation: Planning and Installation*, version 10.2.1 or later.

Installing Dynamic Workload Console version 10.2.0, or earlier

About this task

The HCL Workload Automation administrator installs the Dynamic Workload Console by running the **dwcinst** command. You can run the **dwcinst** specifying a typical set of parameters. In this case, default values are used for all remaining parameters.

Default values are stored in the `dwcinst.properties` file, located in the root directory of the installation image. If you need to modify any of the default values, edit the `dwcinst.properties` file, but do not modify the `dwcinst.template` file located in the same path.

You can optionally configure your environment in SSL mode, by using the **sslkeysfolder** and **sslpassword** parameters and generating automatically the certificates for each workstation in your environment. For details about these parameters, see .

In a typical installation scenario, it is recommended that you install the Dynamic Workload Console as a **non-root user** on UNIX systems and as a **local administrator** on Windows systems.

This user is automatically created by the installation process in the WebSphere Application Server Liberty Base repository. Ensure that the user has full access to the WebSphere Application Server Liberty Base installation directory.

To install the Dynamic Workload Console, perform the following steps:

1. Start the installation specifying a typical set of parameters. Specify the user defined in step 1 in the **--user** parameter. Default values are used for all remaining parameters:

   Windows **On Windows operating systems**

   ```
   cscript dwcinst.vbs --acceptlicense yes --rdbmstype db_type
   --user dwc_admin_user --password dwc_pwd --dbname db_name
   --dbuser db_user --dbpassword db_pwd --dbhostname db_hostname
   --dbport db_port --wlpdir Liberty_installation_dir\wlp
   ```

   UNIX **On UNIX operating systems**

   ```
   ./dwcinst.sh --acceptlicense yes --rdbmstype db_type
   --user dwc_admin_user --password dwc_pwd --dbname db_name
   --dbuser db_user --dbpassword db_pwd --dbhostname db_hostname
   --dbport db_port --wlpdir Liberty_installation_dir/wlp
   ```

   z/OS **On z/OS operating system**

   ```
   ./dwcinst.sh --acceptlicense yes --rdbmstype DB2z
   --user non-root_user --password dwc_pwd --dbname db_name
   --dbuser db_user --dbpassword db_pwd --dbhostname db_hostname
   --dbport db_port --wlpdir Liberty_installation_dir/wlp
   --zlocationname zOS_location_containing_db
   ```

   where:

   password

   The password of the Dynamic Workload Console user.

For more information about all **dwcinst** parameters and default values, see .

Dynamic Workload Console installation - dwcinst script

This script installs the Dynamic Workload Console

This section lists and describes the parameters that are used when running a **dwcinst** script to install the Dynamic Workload Console. For a typical installation scenario, see Installing the Dynamic Workload Console servers. If you are installing in a z/OS environment, see the topic about installing the Dynamic Workload Console in *HCL Workload Scheduler for Z: Planning and Installation*.

Certificates are now required when installing or upgrading HCL Workload Automation. You can no longer install nor upgrade HCL Workload Automation without securing your environment with certificates. The required certificates are:

- ca.crt
- tls.key
- tls.crt

For UNIX systems, ensure that all the files have the ownership of the user who installed the master domain manager and the correct permissions (644).

You can specify values in the properties file, type them in the command line, or use both methods. If a parameter is specified both in the properties file and in the command line, the command line value takes precedence.

Note: Ensure that the **inst_dir** parameter is different from the directory of the installation image and it does not contain any HCL Workload Automation instances.

The log files generated from this command are located in the following path:

Windows **On Windows operating systems**

DWC_home`\logs`

UNIX **On UNIX operating systems**

DWC_DATA_dir`/installation/logs`

z/OS **On z/OS operating system**

DWC_DATA_dir`/installation/logs`

Windows

Syntax for Windows operating systems

Show command usage

```
dwcinst -? | --usage | --help
```

Retrieve the command parameters and values from a properties file

```
dwcinst --file | -f  [properties_file]
```

General information

```
 dwcinst
 --acceptlicense yes|no
[--lang lang_id]
[--inst_dir install_dir]
[--skipcheckprereq true|false]
[--componenttype DWC | FED]
```

Configuration information for the data source

```
--rdbmstype|-r DB2 | DB2Z | ORACLE | MSSQL | POSTGRESQL
[--dbname db_name]
[--dbuser db_user]
[--dbpassword db_password]
[--dbport db_port]
[--dbhostname db_hostname]
[--dbdriverpath db_driver_path]
[--dbsslconnection true | false]
```

Db2 for z/OS-only configuration options

```
[--zlocationname  zOS_location_containing_db]
```

SSL configuration options

```
--sslkeysfolder  keystore_truststore_folder
--sslpassword ssl_password
```

User information

```
 --user | -u dwc_user
 --password | -p dwc_password
```

Configuration information for the application server

```
--wlpdir|-w wlp_directory
```

Security configuration

```
[--httpsport https_port]
[--bootstrapport bootstrap_port]
[--bootsecpport bootstrap_sec_port]
```

UNIX

Syntax for UNIX operating systems

Show command usage

```
dwcinst -? | --usage | --help
```

Retrieve the command parameters and values from a properties file

```
dwcinst --file | -f  [properties_file]
```

General information

```
 dwcinst
 --acceptlicense yes|no
[--lang lang_id]
[--inst_dir install_dir]
[--data_dir dwc_datadir]
[--skipcheckprereq true|false]
[--componenttype DWC | FED]
```

Configuration information for the data source

```
 --rdbmstype|-r DB2 | DB2Z | ORACLE | MSSQL | POSTGRESQL
```

```
    [--dbname db_name
    [--dbuser db_user]
    [--dbpassword db_password]
    [--dbport db_port]
    [--dbhostname db_hostname]
    [--dbdriverpath db_driver_path]
    [--dbsslconnection true | false]
```

Db2 for z/OS-only configuration options

```
[--zlocationname  zOS_location_containing_db]
```

SSL configuration options

```
   --sslkeysfolder  keystore_truststore_folder
   --sslpassword ssl_password
```

User information

```
     --user | -u dwc_user
     --password | -p dwc_password
```

Configuration information for the application server

```
    --wlpdir|-w wlp_directory
```

Security configuration

```
    [--httpsport https_port]
    [--bootstrapport bootstrap_port]
    [--bootsecpport bootstrap_sec_port]
```

z/OS

Syntax for z/OS operating systems

Show command usage

```
dwcinst -? | --usage | --help
```

Retrieve the command parameters and values from a properties file

```
dwcinst --file | -f  [properties_file]
```

General information

```
    dwcinst
    --acceptlicense yes|no
   [--lang lang_id]
   [--inst_dir install_dir]
   [--data_dir dwc_datadir]
```

```
[--componenttype DWC | FED]
```

Configuration information for the data source

```
 --rdbmstype|-r DB2 | DB2Z | ORACLE | MSSQL | POSTGRESQL
 [--dbname db_name]
 [--dbuser db_user]
 [--dbpassword db_password]
 [--dbport db_port]
 [--dbhostname db_hostname]
 [--dbdriverpath db_driver_path]
```

Db2 for z/OS-only configuration options

```
 [--zlocationname  zOS_location_containing_db]
```

SSL configuration options

```
--sslkeysfolder  keystore_truststore_folder
--sslpassword ssl_password
```

User information

```
  --user | -u dwc_user
  --password | -p dwc_password
```

Configuration information for the application server

```
 --wlpdir|-w wlp_directory
```

Security configuration

```
 [--httpsport https_port]
 [--bootstrapport bootstrap_port]
 [--bootsecpport bootstrap_sec_port]
```

Parameters

-? | -usage | -help

Displays the command usage and exits.

--propfile | -f [*properties_file*]

Optionally specify a properties file containing custom values for `dwcinst` parameters. The default file is located in the root directory of the installation image.

Specifying a properties file is suggested if you have a high number of parameters which require custom values. You can also reuse the file with minimal modification for several installations. If you create a custom properties file, specify its name and path with the **-f** parameter.

General information

--acceptlicense *yes|no*

Specify whether to accept the License Agreement.

--lang *lang_id*

The language in which the messages returned by the command are displayed. If not specified, the system LANG is used. If the related catalog is missing, the default C language catalog is used. If neither **--lang** nor LANG are used, the default codepage is set to SBCS. For a list of valid values for these variables, see the following table:

Table 34. Valid values for -lang and LANG parameter

Language	Value
Brazilian Portuguese	pt_BR
Chinese (traditional and simplified)	zh_CN, zh_TW
English	en
French	fr
German	de
Italian	it
Japanese	ja
Korean	ko
Russian	ru
Spanish	es

Note: This is the language in which the installation log is recorded and not the language of the installed component instance. The command installs all languages as default.

--inst_dir

Specify the directory where the Dynamic Workload Console is to be installed. This parameter is optional. The default values varies based on the operating system, as follows:

Windows **On Windows operating systems**

```
%ProgramFiles%\wa\DWC
```

UNIX **On UNIX operating systems**

```
/opt/wa/DWC
```

z/OS **On z/OS operating system**

```
/opt/wa/DWC
```

After installing, you can find this value in the `twainstance<instance_number>.TWA.properties` file, by checking the **DWC_basePath** parameter. For more information, see the topic about Finding out what has been installed in which HCL Workload Automation instances in *Administration Guide*.

--data_dir *dwc_datadir*

Specify the path to a directory where you want to store the logs and configuration files produced by Dynamic Workload Console. This parameter is optional. If you do not specify this parameter, all data files generated by the Dynamic Workload Console are stored in *DWC_home*/`DWC_DATA`. This path is called, in the publications, *DWC_DATA_dir*.

--skipcheckprereq true | false

If you set this parameter to `true`, Dynamic Workload Console does not scan system prerequisites before starting the installation. This parameter is optional. The default value is `false`. For more information about the prerequisite check, see Scanning system prerequisites for HCL Workload Automation.

Configuration information for the data source

--rdbmstype|-r *rdbms_type*

The database type. Supported databases are:

- **DB2**
- **ORACLE**
- **MSSQL** This value applies to MSSQL and supported MSSQL cloud-based databases.
- **POSTGRESQL**
-

This parameter is required and has no default value.

--dbname *db_name*

The name of the Dynamic Workload Console database. This parameter is optional. The default value is **DWC**.

--dbuser *db_user*

The user that has been granted access to the Dynamic Workload Console tables on the database server. This parameter is required.

--dbpassword *db_password*

The password for the user that has been granted access to the Dynamic Workload Console tables on the database server. This parameter is required. Special characters are not supported. You can optionally encrypt the password. For more information, see Encrypting passwords (optional).

--dbport *db_port*

The port of the database server. This parameter is required.

--dbhostname *db_hostname*

The host name or IP address of database server. This parameter is required.

--dbdriverpath *db_driver_path*

The path where the database drivers are stored. This parameter is optional, but becomes required when you set the **rdbmstype** parameter to **DB2Z**. If you use Db2 for z/OS with the Dynamic Workload Console version 10.2.4 or later, transfer the drivers in binary mode from the directory where you installed Db2 for z/OS to a directory of your choice. Specify the driver path using this parameter.

By default, the configuration script references the JDBC drivers supplied with the product images. If your database server is not compatible with the supplied drivers, then contact your database administrator for the correct version to use with your database server and specify the driver path using this parameter. Ensure you provide the same path in the configureDb, serverinst, and dwcinst commands.

--dbsslconnection true | false

Enables or disables the SSL connection to the database. This value must always be **false** when `--rdbmstype` is **DB2Z**.

The default value is **false**.

SSL configuration options

--sslkeysfolder *keystore_truststore_folder*

The name and path of the folder containing certificates in PEM format. The installation program automatically processes the keystore and truststore files using the password you specify with the **--sslpassword** parameter. The folder must contain the following files:

- **ca.crt**

  The Certificate Authority (CA) public certificate. Note that if certificates being installed are part of a chain consisting of 3 or more certificates (one Root CA, followed by one or more Intermediate CAs, followed by the end user certificate), then this file must contain the Root CA certificate only. Any Intermediate CA certificates must be stored in the `additionalCAs` subfolder, which therefore becomes a mandatory subfolder. Each Intermediate CA must be stored in the `additionalCAs` subfolder in its own file.

> **Note:** From V10.2.3, if certificates being installed are part of a chain, the ca.crt can contain also the intermediate CAs. In this case, it must begin with one or more intermediate CA certificates and end with the Root ca.

- **tls.key**

  The private key of the end user certificate for the instance to be installed.

- **tls.crt**

  The public part of the previous key, that is the end user certificate.

For UNIX systems, ensure that all the files have the ownership of the user who installed the master domain manager and the correct permissions (644).

You can optionally create a subfolder to contain one or more `*.crt` files to be added to the server truststore as trusted CA, whose name must be `additionalCAs`. This can be used for example to add to the list of trusted CAs the certificate of the LDAP server or DB2 server. Additionally, you can store here any intermediate CA certificate to be added to the truststore. The subfolder must be named **additionalCAs**. Note that if the end user certificate being installed in the instance is part of a chain consisting of 3 or more certificates (one Root CA, followed by one or more Intermediate CAs, followed by the end user certificate), then the Intermediate CAs certificates must be stored in the `additionalCAs` subfolder, which therefore becomes a mandatory subfolder. Each Intermediate CA must be stored in the `additionalCAs` subfolder in its own file.

For further information about how to generate custom certificates, see the topic about managing certificates using Certman in *HCL Workload Automation: Planning and Installation*.

--sslpassword *ssl_password*

The password for the certificates.

For more information, see sslkeysfolder on page 259.

You can optionally encrypt the password using the secure script. For more information, see Optional password encryption - secure script on page 284.

--enablefips false

Specify whether you want to enable FIPS. The default is `false`. This parameter is optional.

Db2 for z/OS-only configuration syntax

--zlocationname *zos_location_containing_db*

The name of an already existing location in the z/OS environment that will contain the new database. The default value is LOC1.

User information

--user

Specify the administrator of the Dynamic Workload Console. You can use this account to log in to the Dynamic Workload Console and manage your environment. This parameter is optional. The default value is `dwcadmin`.

--password

Specify the password for the Dynamic Workload Console user. This parameter is required. You can optionally encrypt the password. For more information, see Encrypting passwords (optional).

Windows **On Windows operating systems**

Supported characters for the password are alphanumeric, dash (-), underscore (_) characters, and ()|?*~+.@!^

UNIX **On UNIX operating systems**

Supported characters for the password are any alphanumeric, dash (-), underscore (_) characters, and ()|?=*~+.

Configuration information for the application server

--wlpdir

Specify the path where Open Liberty is installed. This parameter is required.

z/OS **On z/OS operating system**

Specify the path where WebSphere Application Server for z/OS Liberty is installed. This parameter is required.

Security configuration

--httpsport

Specify the HTTPS port, to be used in the Dynamic Workload Console URL. This parameter is optional. The default value is `9443`.

--bootstrapport

Specify the bootstrap port. This parameter is optional. The default value is `12809`.

--bootsecport

Specify the bootstrap security port, to be used for connecting to the Z connector. This parameter is optional. The default value is `19402`.

Defining a z/OS engine in the Z connector

To define a z/OS engine, perform the following steps.

Installing Liberty
Encrypting passwords (optional)
Creating and populating the database
Installing the DWC server
Defining z/OS engine in Z connector

1. Depending on your operating system, browse to the `connectionFactory.xml` template:

   Windows **Windows operating systems**

   <DWC_home>`\usr\servers\dwcServer\configDropins\templates\zconnectors`

   UNIX **UNIX operating systems**

   <DWC_home>`/usr/servers/dwcServer/configDropins/templates/zconnectors`

   z/OS **z/OS operating systems**

   <DWC_home>`/usr/servers/dwcServer/configDropins/templates/zconnectors`

2. Copy the `connectionFactory.xml` template to a temporary directory.

   z/OS On z/OS system, it is required that you copy the file on your workstation in binary mode.

3. Edit the file by specifying the required connection details.

   The following example shows the `connectionFactory.xml` file, as it is created in the `zconnectors` folder at installation time:

```
<server description="zonnector_configuration">

        <connectionFactory id="$(zServerConnectionName)"
        jndiName="eis/tws/zconn/$(zServerConnectionName)">
        <properties.ZOSConnectorAdapter hostName="$(zServerHostName)"
                                        portNumber="$(zServerPortNumber)"
                                        useSsl="$(zConnUseSsl)"
                                        connectionTimeoutCleanup="10"/>
                                        </connectionFactory>
                                        </server>
```

   Set the following values:

   zServerConnectionName

   Name of the connection between the Z connector and z/OS server started task.

Starting from the Dynamic Workload Console V10.2.3, if you enabled the mirroring of data to a database, this value must be coincident with the value set in the CONNECTIONFACTORYID parameter of the MIRROPTS statement.

zServerHostName

The host name or TCP/IP address of the remote z/OS system where the Z controller is installed.

Note: If the Dynamic Workload Consoleand Z controller are installed on the same z/OS system, you must specify the TCP/IP address of the z/OS system.

zServerPortNumber

The TCP/IP port that is used to communicate with the z/OS server started task, hence with the Z controller. This is the value specified in the SRVPORTNUMBER parameter of the TCPOPTS statement in the server started task.

zConnUseSsl

Set `true` to enable the SSL communication between the Z connector and the remote z/OS system, or `false` to disable this property. This property is optional. The default is `false`.

connectionTimeoutCleanup

The connection timeout cleanup for the z/OS connection.

Note: In the `TWSZOSConnConfig.properties` file located in `<INSTALL_DIR>\usr\servers\dwcServer\resources\properties`, set the name of the host where you installed the Dynamic Workload Console as follows:

```
com.ibm.tws.zconn.usr.mapping.hostName=DWC_hostname
```

4. Optionally, create a backup copy of the configuration file in a different directory, if the file is already present.
5. Depending on your operating system, copy the edited `connectionFactory.xml` file to the following path. Changes are immediately effective.

   Windows **Windows operating systems**

   <DWC_home>`\usr\servers\dwcServer\configDropins\overrides`

   UNIX **UNIX operating systems**

   <DWC_DATA_dir>`/DWC_DATA/usr/servers/dwcServer/configDropins/overrides`

   z/OS **z/OS operating systems**

   Ensure that you copy the file in binary mode. *<DWC_DATA_dir>*`/usr/servers/dwcServer/configDropins/overrides`.
6. Replicate the change on all the Dynamic Workload Console instances in the domain.

Defining a z/OS engine in the Dynamic Workload Console

Steps to create a z/OS engine connection in the Dynamic Workload Console

After logging to the Dynamic Workload Console using the administrator user ID or another user ID with the appropriate roles assigned, perform the following steps to define a connection to one of your supported HCL Workload Automation for Z engines.

For detailed information about the roles required to run the Dynamic Workload Console, see how to configure security roles to users and groups in the *HCL Workload Automation: Administration Guide*.

According to the version of the Dynamic Workload Console you are using, perform the following procedure:

Dynamic Workload Console version 10.2.0, or later

1. From the navigation toolbar, click **Administration** > **Engine Connections**.
2. From the displayed panel you can create, edit, delete, or share an engine connection, and test the connection to the remote server where HCL Workload Automation for Z is installed.
3. Click **Create new** > **z/OS**.
4. From the **Engine Connection - New Engine z/OS** window, in the Engine Info tab assign a name to the engine connection and specify the required information. For more details about fields and options, see the online help by clicking the sandwich menu next to the Engine Connections page and select **Help Page**.

   To share the engine, select the **Enable sharing** check box in the Manage Sharing tab, and specify the required information.

   To test the connection to the HCL Workload Automation for Z database (mandatory for managing reporting and event management functions), from the reporting tab select **Enable reporting** and specify the user credentials.
5. Click **Test Connection** to check that the configuration was successful and that the Dynamic Workload Console is communicating with the selected engine. .

Dynamic Workload Console version earlier than 10.2.0

1. From the navigation toolbar, click **Administration** > **Manage Engines**.
2. From the displayed panel you can create, edit, delete, or share an engine connection, and test the connection to the remote server where HCL Workload Automation for Z is installed. You can order the list of engine connections displayed in this panel by using sorting criteria that you select with the buttons at the upper left corner of the table.
3. Click **New Engine**.
4. In the Engine Connection Properties window, assign a name to the engine connection and specify the required information. For more details about fields and options, see the online help by clicking the "**?**" in the top right corner. If you want to test the connection to the HCL Workload Automation for

Z database (mandatory for managing reporting and event management functions), you must select **Enable reporting** and specify the user credentials.

5. Click **Test Connection** to check that the configuration was successful and that the Dynamic Workload Console is communicating with the selected engine. .

Encrypting the connection between the Z connector and the server started task

An overview of encrypting the connection between the Z connector and the server started task.

SSL encryption can be enabled to protect communication between the Z connector and the HCL Workload Automation for Z server started task.

To enable the use of the SSL default certificates provided with HCL Workload Automation for Z or your own certificates, you must:

1. On the UNIX System Services of the z/OS system where the server runs, use the `gskkyman` utility of z/OS® Cryptographic Services System SSL to create the keystore database and generate the password file. Following this, you can import the default SSL certificates from the Z connector or from the Dynamic Workload Console.
2. Configure HCL Workload Automation for Z by specifying the TCPOPTS statement for the server started task.

For details, see Security for TCP/IP connections on page 109.

Navigating the Dynamic Workload Console

An overview to the Dynamic Workload Console.

After you have installed and configured the Dynamic Workload Console, you can start defining and scheduling your workload. Log in by connecting to:

```
https://<your_ip_address>:<https_port>/console/login.jsp
```

You can access the Dynamic Workload Console from any computer in your environment using a web browser through the secure HTTPS protocol.

To have a quick and rapid overview of the portal and of its use, after logging in, the Welcome page for the Dynamic Workload Console is displayed in the console window. This window has a navigation menu across the top, organized in categories. Each category drops down to display a number of options that when clicked, display a page in the work area on the left. Each page displays with a title in its tabbed window in the work area.

Several products can be integrated in this portal and their related entries are listed together with those belonging to the Dynamic Workload Console in the navigation bar displayed at the top of the page.

The navigation bar at the top of the page is your entry point to the Dynamic Workload Console.

Starting and stopping the WebSphere Application Server Liberty Base

Use the startAppServer and stopAppServer commands or the equivalent from the Dynamic Workload Console to start or stop the WebSphere Application Server Liberty. For a description of these commands, see *HCL Workload Automation: User's Guide and Reference*.

These commands also stop appservman, the service that monitors and optionally restarts WebSphere Application Server Liberty.

If you do not want to stop appservman, you can issue startAppServer or stopAppServer, supplying the –direct argument. These scripts are located in `TWA_home/appservertools`.

The complete syntax of startAppServer and stopAppServer is as follows:

UNIX **UNIX™**

Start the application server

```
./startAppServer.sh  [-direct]
```

Stop the application server

```
./stopAppServer.sh  [-direct]
```

Note: If your WebSphere Application Server Liberty is installed for the Dynamic Workload Console, use the following syntax:

```
./stopAppServer.sh [-direct]
            [-user <user_ID>
           -password <password>]
```

The user ID and password are optional only if you have specified them in the `soap.client.props` file located in the properties directory of the WebSphere Application Server Liberty profile. Unlike the master domain manager installation, when you install the Dynamic Workload Console the `soap.client.props` file is not automatically customized with these credentials.

Windows **Windows™**

Start the application server

```
startAppServer.bat [-direct]
```

Stop the application server

```
stopAppServer.bat [-direct
            [-wlpHome <installation_directory>]
            [-options <parameters>]]
```

z/OS **z/OS**

Start the application server

```
./startAppServer.sh  [-direct]
```

Stop the application server

```
./stopAppServer.sh  [-direct]
```

where the arguments are as follows:

–direct

Optionally starts or stops the application server without starting or stopping the application server monitor appservman.

For example, you might use this after changing some configuration parameters. By stopping WebSphere Application Server Liberty without stopping appservman, the latter will immediately restart WebSphere Application Server Liberty, using the new configuration properties.

UNIX This argument is mandatory on UNIX™ when the product components are not integrated.

–options ***parameters***

Optionally supplies parameters to the WebSphere Application Server Liberty startServer or stopServer commands. See the WebSphere Application Server Liberty documentation for details.

–wlpHome ***installation_directory***

Defines the WebSphere Application Server Liberty installation directory, if it is not the default value.

Installation log files

The type of log files you find on your system depends on the type of installation you performed.

About this task

UNIX To simplify administration, configuration, and backup and recovery, a new default behavior has been implemented with regard to the storage of product data and data generated by HCL Workload Automation, such as logs and configuration information. On UNIX operating systems, these files are stored by default in the `DATA_DIR` directory, which you can optionally customize at installation time. By default, this directory is `INST_DIR/DWC_DATA` for the Dynamic Workload Console. The product binaries are stored, instead, in the installation directory.

Note: If you deployed the product components using Docker containers, this is the default behavior and it cannot be modified. However, if you installed the product components using the command-line installation, the **--data_dir** parameter can be used to change the path.

Dynamic Workload Console

`INST_DIR/DWC_DATA/installation/logs`

Windows On Windows operating systems, installation log files for the Dynamic Workload Console are stored in *`INST_DIR`*`\logs`.

z/OS On z/OS operating system, installation log files for the Dynamic Workload Console are stored in *`INST_DIR`*`/DWC_DATA`.

Chapter 9. Configuring the Dynamic Workload Console

Some links and pointers on configuration tasks that are needed for the Dynamic Workload Console.

The following list shows the links and pointers to sections that document the configuration tasks needed for the Dynamic Workload Console. You can perform the following optional configuration steps at any time after the installation.

- Configuring new users to access the Dynamic Workload Console: see the section about configuring access to the Dynamic Workload Console in the *HCL Workload Automation: Administration Guide*.
- Configuring the Dynamic Workload Console to use a user registry:
  - For configuring the Dynamic Workload Console with LDAP - RACF®, see the *Configuring Lightweight Directory Access Protocol user registries* section in the WebSphere® documentation at: https://www.ibm.com/support/knowledgecenter/SSEQTP/mapfiles/product_welcome_was.html.
  - For configuring access to the Dynamic Workload Console, see the corresponding section in the *HCL Workload Automation: Administration Guide*.
- Configuring roles to access the Dynamic Workload Console: see the corresponding section in the *HCL Workload Automation: Administration Guide*.
- Configuring the Dynamic Workload Console to use Single Sign-On: see the corresponding section in the *HCL Workload Automation: Administration Guide*.
- Securing your communication with the Secure Socket Layer protocol: see the section about customizing the SSL connection between the Dynamic Workload Console and components with a distributed connector in the *HCL Workload Automation: Administration Guide*.
- Configuring the Dynamic Workload Console to launch in context: see the corresponding section in the *HCL Workload Automation: Administration Guide*.

Note: If, after installing, you have more than one instance of WebSphere Application Server managing any HCL Workload Automation products, you must ensure that they have the same LTPA token_keys.

For detailed information about how to configure the Dynamic Workload Console, see the *HCL Workload Automation: Administration Guide*.

For more information about configuring authentication using the Lightweight Directory Access Protocol (LDAP), see the *HCL Workload Automation: Administration Guide*.

Configuring a user registry

In this topic you can find information about how to configure a user registry.

About this task

By default, the dynamic domain manager, the Dynamic Workload Console, and the master domain manager are configured to use a local file-based user repository. For more information about supported authentication mechanisms, see the topic about available configurations in the *Administration Guide*.

You can implement an OpenID Connect (OIDC) user registry, a Lightweight Directory Access Protocol (LDAP) user registry, or a basic user registry by configuring the sample authentication templates provided in XML format. You can further customize the templates by adding additional elements to the XML files. For a full list of the elements that you can configure to complement or modify the configuration, see the related Open Liberty documentation, for example LDAP User Registry (ldapRegistry).

To configure an OIDC user registry, see Configuring an OIDC user registry on page 270 .

To configure an LDAP user registry, for example as Active Directory, see Configuring an LDAP user registry on page 271.

To configure a basic user registry, see Configuring a basic user registry on page 273.

Configuring an OIDC user registry

About this task

You can implement an OIDC user registry by configuring the sample authentication template provided in XML format: `openid_connect.xml`.

To configure an OIDC user registry, complete the following steps:

1. Copy the following template to a working directory:

```
<server>
    <featureManager>
        <feature>openidConnectClient-1.0</feature>
    </featureManager>

    <authFilter id="restFilterOpenID">
        <requestUrl id="restUrl" urlPattern="jwt/ibm/api|/dwc/rest/roles|/dwc/
ServiceDispatcherServlet?ServiceName=PrefExport|/metrics" matchType="notContain"/>
    </authFilter>

    <openidConnectClient id="keycloak"
        clientId="wa-service"
        clientSecret="put_oidc_secret_here"
        httpsRequired="true"
        userIdentifier="preferred_username"
        signatureAlgorithm="RS256"
        scope="openid"
        authFilterRef="restFilterOpenID"
        inboundPropagation="supported"
        groupIdentifier="groups"
        accessTokenAttributeName="groups"
        groupNameAttribute="groups"
        hostNameVerificationEnabled="false"
        realmName = "your_realm_name"
        redirectToRPHostAndPort="https://dwc_ingress_hostname"
        discoveryEndpointUrl="https://oidc ingress hostname/realms/wa
                    /.well-known/openid-configuration">
    </openidConnectClient>
</server>
```

2. Edit the template file in the working folder with the desired configuration by adding users and groups as necessary.
3. Optionally, create a backup copy of the configuration file in the `overrides` folder, if already present.
4. Copy the updated template file to the `overrides` folder.
5. To upload the certificates of the OIDC provider, browse to `<DWC_home>/java/jre/bin` and run the following command:

   ```
   keytool -importcert -file ingress-cert.pem
    -keystore <DWC_home>/usr/servers/dwcServer/resources/security/TWSServerTrustFile.p12 -alias
    ingress-cert -storepass <password_keystore>
   ```

   where

   ingress-cert.pem

   The certificates file to be imported into the Dynamic Workload Console.

   ingress-cert

   The alias defined during the import of the certificate.
6. On Keycloak, ensure you define the group with the same name present in the OpenID file.

Note: If one or more messages similar to the following are displayed, perform the steps listed below.

```
CWPKI0819I: The default keystore is not created because a password is not configured on the
<keyStore id="defaultKeyStore"/> element, and the 'keystore_password' environment variable is not set.
CWOAU0073E: An authentication error occurred. Try closing the web browser and authenticating again,
or contact the site administrator if the problem persists.
```

1. On the workstation where the Dynamic Workload Console is installed, browse to the `server.xml` file located in `<dwc_installation_directory>/usr/servers/dwcServer`.
2. Open the file with a text editor and change the value of the **sameSiteCookie** parameter from `Strict` to `lax`.
3. Optionally, trust the Dynamic Workload Console certificate with the keycloak certificate.

Configuring an LDAP user registry

About this task

You can implement an LDAP based user repository by configuring the following sample authentication templates provided in XML format. The following are the supported authentication methods and the corresponding sample template that can be configured to replace the configuration file currently in use:

- OpenLDAP: `auth_OpenLDAP_config.xml`
- IBM® Directory Server: `auth_IDS_config.xml`
- Windows Server Active Directory: `auth_AD_config.xml`

To configure a common authentication provider for both the HCL Workload Automation and the Dynamic Workload Console, complete the following steps:

1. Assign a role to your authentication provider user or group.
    a. Log in to the Dynamic Workload Console as administrator and access the **Manage Roles** page.
    b. Add a new **Entity** of type **Group** to the role you want to assign to your authentication provider user or group and click **Save**.
2. Update the authentication configuration template file with the details about your authentication provider server.
    a. Copy the template file to a working directory. The templates are located in the following path:

       UNIX

       Dynamic Workload Console

       ```
       DWC_DATA_dir/usr/servers/dwcServer/configDropins/templates/authentication
       ```

       master domain manager

       ```
       TWA_DATA_DIR/usr/servers/engineServer/configDropins/templates/authentication
       ```

       Windows

       Dynamic Workload Console

       ```
       DWC_home\usr\servers\dwcServer\configDropins\templates\authentication
       ```

       master domain manager

       ```
       TWA_home\usr\servers\engineServer\configDropins\templates\authentication
       ```

    b. Edit the template file in the working directory with the desired configuration.
    c. Optionally, create a backup copy of the configuration file in a different directory, if the file is already present. To avoid conflicts, ensure the backup copy is in a directory different from the following directories: `configDropins/templates` and `configDropins/overrides`.
    d. Copy the updated template file to the `overrides` directory.
    e. The `overrides` directory is located in the following path:

       UNIX

       Dynamic Workload Console

       ```
       DWC_DATA_dir/usr/servers/dwcServer/configDropins/overrides
       ```

       master domain manager

       ```
       TWA_DATA_DIR/usr/servers/engineServer/configDropins/overrides
       ```

       Windows

       Dynamic Workload Console

       ```
       DWC_home\usr\servers\dwcServer\configDropins\overrides
       ```

master domain manager

```
TWA_home\usr\servers\engineServer\configDropins\overrides
```

f. Stop and restart Open Liberty using the stopappserver and startappserver commands located in `TWA_home/appservertools`.

For more information about configuring an LDAP registry, see the Open Liberty documentation, for example: Configure an LDAP user registry and Federated user registries.

Configuring a basic user registry

About this task

You might want to use a basic user registry by defining the users and groups information for authentication on Open Liberty, even though this type of authentication is not recommended. This type of authentication cannot be used for production, but only for test purposes.

To configure basic user registry, complete the following steps:

1. Copy the `auth_basicRegistry_config.xml` template from the `templates` folder to a working folder.
2. Edit the template file in the working folder with the desired configuration by adding users and groups as necessary.

   To add a user, add an entry similar to the following in the **basicRegistry** section:

   ```
   <user name="nonadminuser" password="{xor}Ozo5PiozKw=="/>
   ```

   To add a group, add an entry similar to the following in the **basicRegistry** section:

   ```
   <group name="TWSUsers">
           <member name="nonadminuser"/>
           </group>
   ```
3. Store the password in xor, aes, or hash formats using the Open Liberty securityUtility command, as described in securityUtility command.

   This utility requires the JAVA_HOME environment variable to be set. If you do not have Java installed, you can optionally use the Java version provided with the product and available in:

   HCL Workload Automation

   `<INST_DIR>/TWS/JavaExt/jre/jre`

   Dynamic Workload Console

   `<DWC_INST_DIR>/java/jre/bin`
4. Create a backup copy of the configuration file in the `overrides` folder, if already present.
5. Copy the updated template file to the `overrides` folder. Maintaining the original folder structure is not required.

Configuring the Dynamic Workload Console for Single Sign-On

Single Sign-On (SSO) is a method of access control that allows a user to authenticate once and gain access to the resources of multiple applications sharing the same user registry.

This means that using SSO you can run queries on the plan or manage object definitions on the database accessing the engine without authenticating, automatically using the same credentials you used to log in to the Dynamic Workload Console.

The same is true when working with the Self-Service Catalog and Self-Service Dashboards apps from a mobile device. If the Dynamic Workload Console has been configured to use SSO, then these apps automatically use the same credentials used to log in to the Dynamic Workload Console.

After completing the installation or upgrade, you can set up Single Sign-On (SSO) for the Dynamic Workload Console and master domain manager using either an LTPA registry or a basic user registry. To achieve this, Dynamic Workload Console and master domain manager need to use the same user registry. Additionally, ensure that the contents of the `ltpa.keys` file are identical on both the Dynamic Workload Console and the master domain manager. The `ltpa.keys` file is located in the following path:

```
usr/servers/engineServers/resources/security
```

The Lightweight Directory Access Protocol (LDAP) is an application protocol for querying and modifying directory services running over TCP/IP - see the information about configuring a common LDAP for both the master and console in the post-installation section of the *Planning and Installation Guide* for more details.

If you configured Dynamic Workload Console to use Single Sign-On with an engine, then, the following behavior is applied:

If engine connection has the user credentials specified in its definitions

These credentials are used. This behavior regards also engine connections that are shared along with their user credentials.

If the user credentials are not specified in the engine connection

The credentials you specified when logging in to Dynamic Workload Console are used. This behavior regards also shared engine connections having unshared user credentials.

For detailed information about how to configure SSO using an LTPA token or an MP-JWT token, see How to configure the Dynamic Workload Console and the master domain manager for Single Sign-On.

Chapter 10. Upgrading the Dynamic Workload Console

This section describes how to upgrade the Dynamic Workload Console from v 9.5.*x* to 10.2.*x*.

When upgrading your HCL Workload Automation environment, it is a good practice to start with the upgrade of the Dynamic Workload Console first. If you upgrade the console to the new product version level, you can then use it to verify that your environment is working after upgrading the remaining components.

With Version 9.5, the Dynamic Workload Console is based on a new architectural foundation that does not include Jazz for Service Management nor Dashboard Application Services Hub, therefore, no direct upgrade procedure is supported, but you perform a fresh installation of the Dynamic Workload Console at version 9.5.0.*x* or 10.2.*x*.

Perform the following tasks:

1. Upgrade the WebSphere Application Server Liberty Base, as described in the related section in *HCL Workload Automation: Planning and Installation*.
2. Perform a direct upgrade of the Dynamic Workload Console and its database, as described in the related section in *HCL Workload Automation: Planning and Installation*.

Chapter 11. Uninstalling the Dynamic Workload Console

Before you begin

Ensure that all HCL Workload Automation processes, services and the WebSphere Application Server Liberty process are stopped, and that there are no active or pending jobs. For information about stopping the processes and services see the topic about starting and stopping processes on a workstation in the *HCL Workload Automation: User's Guide and Reference*.

About this task

To uninstall the Dynamic Workload Console, perform the following steps:

1. Change directory to the folder containing the uninstallation script:

   ```
   cd DWC_INST_DIR/tools
   ```

2. Run the uninstallation process by running the script as follows:

   Windows **Windows™ operating systems**

   ```
   cscript uninstall.vbs --prompt no
   ```

   UNIX **UNIX™ and Linux™ operating systems**

   ```
   ./uninstall.sh --prompt no
   ```

   The procedure runs without prompting the user to confirm the uninstallation.

Results

The log file generated by this command are located in:

Windows **On Windows operating systems**

<*DWC_home*>`\logs`

UNIX **On UNIX operating systems**

<*DWC_DATA_dir*>`/installation/logs`

Part IV. Deploying with containers

Docker is a state-of-the-art technology that creates, deploys, and runs applications by using containers. Packages are provided containing an application with all of the components it requires, such as libraries, specific configurations, and other dependencies, and deploy it in no time on any UNIX, Windows, and Linux on Z workstation, regardless of any different settings between the source and the target workstation.

Docker adoption ensures standardization of your workload scheduling environment and provides an easy method to replicate environments quickly in development, build, test, and production environments, speeding up the time it takes to get from build to production significantly. Install your environment using Docker to improve scalability, portability, and efficiency.

To deploy HCL Workload Automation using a Docker container, proceed as follows:

1. Access and download the Docker image from the entitled registry. For details about the complete procedure, see Deploying containers with Docker on page 304.
2. You can choose to deploy all product containers with a single command, or you can deploy each product component container individually. Start and configure the HCL Workload Automation containers. For details about the complete procedure, see Deploying containers with Docker on page 304.

   More detailed technical information for each component can be found in the sample readme files:
   - HCL Workload Automation Server
   - HCL Workload Automation Console
   - HCL Workload Automation dynamic agent
   - HCL Workload Automation z-centric agent
   - Workload Automation FileProxy
3. Access the container to verify the status and run HCL Workload Automation commands. For details, see Accessing the Docker containers on page 304.

You can also use docker containers to store all the latest integrations available on Automation Hub. For further information, see the Container plug-in.

Deploying z-centric and dynamic agents on Kubernetes

To ensure a fast and responsive experience when using HCL Workload Automation for Z, you can deploy z-centric and dynamic agents on a Kubernetes cluster. Kubernetes is a portable, extensible, open-source platform for managing containerized workloads and applications.

A cloud deployment ensures access anytime anywhere and is a fast and efficient way to get up and running quickly. It also simplifies maintenance, lowers costs, provides rapid upscale and downscale, minimizes IT requirements and physical on-premises data storage.

Orchestrating your Kubernetes workload from a containerized agent optimizes allocations of resources in your cluster and reduces the chance of human errors. Find out more about the advantages of a stable containerized agent in the Orchestrating your workload on cloud from IBM Z Workload Scheduler video, available on the Workload Automation YouTube channel.

Deployment of the z-centric and dynamic agent containers on Kubernetes is supported on the following cloud provider infrastructures. For details and deployment instructions, see the HCL Workload Automation z-Centric Chart readme file.

Amazon Web Services (AWS) Elastic Kubernetes Service (EKS) (Amazon EKS)

The z-centric and dynamic agents are offered on the Amazon Web Services cloud. Within just a few minutes, access the product Helm chart and container images and easily launch an instance to deploy an agent with full on-premises capabilities.

Note: At agent installation completion, the AWS cluster returns a host name 77 characters long. Because in the ROUTOPTS statement the maximum allowed length for host names is 52 characters, ensure that you map the host name returned by AWS to a name no longer than 52 characters.

Azure Kubernetes Service (AKS)

Deploy and manage HCL Workload Automation containerized product components on the Azure AKS, a container orchestration service available on the Microsoft Azure public cloud. You can use Azure AKS to deploy and manage containers in the cluster environment.

Google GKE

Google Kubernetes Engine (GKE) provides a managed environment for deploying, managing, and scaling your containerized applications using Google infrastructure. The Google GKE environment consists of multiple machines grouped together to form a cluster.

Red Hat OpenShift

The HCL Workload Automation product components can be deployed onto a Red Hat OpenShift, V4.*x* environment by using helm charts.

Part V. Deploying AI Data Advisor

You can deploy AI Data Advisor (AIDA) by using Docker or Kubernetes.

AIDA is composed of two major components: AIDA Exporter and AIDA Engine. Each component contains a number of services:

AIDA Exporter

Exporter

Through HCL Workload Automation APIs, exports KPIs metrics from HCL Workload Automation (according to OpenMetrics standard) and stores them into AIDA OpenSearch database.

Also, it exports Alert definitions from HCL Workload Automation and imports them into OpenSearch.

AIDA Engine

Predictor

Calculates the expected values of each KPI, also considering special days.

Anomaly detection and alert generation

Detects anomalies in KPIs trend by comparing observed KPI data points with expected values, and generates alerts when trigger conditions are met.

Email notification

Sends email notification when alerts are generated.

Orchestrator

Orchestrates KPI prediction and anomaly detection.

UI

AIDA User Interface.

Internal event manager

Manages communication among AIDA services.

Also, AIDA uses:

OpenSearch (an Elasticsearch technology)

To store and analyze data.

Keycloak

To manage security and user access in AIDA (Docker deployment only). If not deployed, the Dynamic Workload Console user authentication roles will be used.

Nginx

As a reverse proxy for its components.

Deploying AIDA using Docker

To monitor HCL Workload Automation and HCL Workload Automation for Z engines, you can deploy AIDA with Docker by using AIDA.sh script. This script provides options to run Docker Compose operations and AIDA configuration steps.

For details, see the following readme file for Docker:

- HCL AI Data Advisor for HCL Workload Automation

Deploying AIDA using Kubernetes

To monitor HCL Workload Automation engines only (not HCL Workload Automation for Z engines), you can deploy AIDA by using an **helm chart**. This helm chart is included in the Workload Automation product helm chart that allows you to deploy Workload Automation and all its components in one shot.

For details, see the following readme file for Kubernetes:

- HCL AI Data Advisor for HCL Workload Automation

Note: Horizontal pod autoscaling based on memory and network usage is supported for **Anomaly detection and alert generation** and **Predictor** services. In case of high memory utilization, Kubernetes replicates pods. When memory usage decreases, pods are deleted.

Deploying AIDA on Amazon Web Services (AWS) Marketplace

You can use Amazon Web Services (AWS) Marketplace to subscribe to HCL Workload Automation and deploy your environment on the AWS secure cloud platform, including AIDA as optional component.

For more information see Deploying from Amazon Web Services (AWS) Marketplace.

Part VI. Installing HCL Workload Automation Agent (z-centric agent)

The following topics describe how to install, migrate, and uninstall the HCL Workload Automation Agent (also known as z-centric agent).

Note: If you are installing on IBM i systems, see Installing HCL Workload Automation Agent on IBM i systems on page 317.

Chapter 12. Installing the HCL Workload Automation Agent (z-centric)

You install the HCL Workload Automation Agent (z-centric) to run workload from the mainframe to distributed systems with a low cost of ownership.

With the z-centric agent you can run your workload:

Statically

To run existing job types, for example scripts, on a specific HCL Workload Automation Agent. In this case, you install the HCL Workload Automation Agent on the distributed systems and connect it to the z/OS system through the HCL Workload Automation for Z controller.

Statically including job types with advanced options

In this case, you install the HCL Workload Automation Agent on the distributed systems adding the Java™ run time, and connect it to the z/OS system through the HCL Workload Automation for Z controller.

Dynamically

To run existing job types allowing the product to assign them to the workstation that best meets both the hardware and software requirements needed to run them. In this case, you install the HCL Workload Automation Agent on the distributed systems adding the dynamic capabilities, and connect it to the dynamic domain manager. For a detailed description about how to install a dynamic domain manager for a Z controller, see the *HCL Workload Automation: Planning and Installation*.

During the installation of the dynamic domain manager for a Z controller, you must provide the **master domain manager** and **HCL Workload Automation Netman port** values.

Dynamically including job types with advanced options

To run existing job types and job types with advanced options allowing the product to assign them to the workstation that best meets both the hardware and software requirements needed to run them. In this case, you install the HCL Workload Automation Agent on the distributed systems adding the dynamic capabilities and the Java™ run time, then connecting it to the dynamic domain manager. For a detailed description about how to install a dynamic domain manager for a Z controller, see the *HCL Workload Automation: Planning and Installation*.

During the installation of the dynamic domain manager for a Z controller, you must provide the **master domain manager** and the **HCL Workload Automation Netman port** values.

For information about how to use the z-centric agent, see *Scheduling End-to-end with z-centric Capabilities*.

Authorization roles for running the twsinst script

Before you start to install, upgrade, or uninstall with **twsinst**, ensure that you have the following authorization requirements:

Windows **Windows™ operating system**

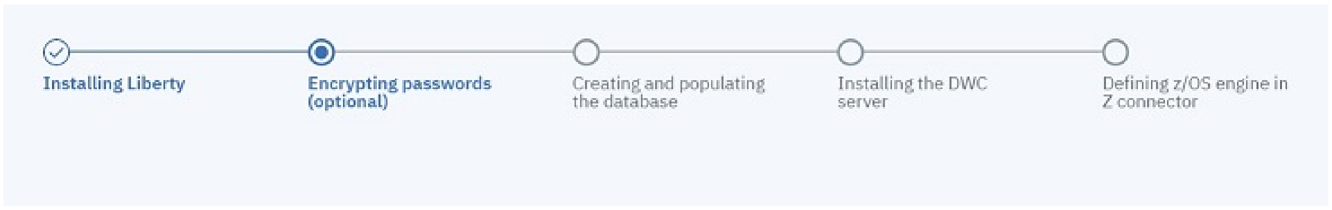
If you set the Windows User Account Control (UAC), your login account must be a member of the Windows™ **Administrators** group or domain administrators with the rights **Act as Part of the Operating System**.

If you set the Windows User Account Control (UAC) on the workstation, you must run the installation as **administrator**.

Encrypting passwords (optional)

How to encrypt passwords required by the installation, upgrade, and management processes.

About this task


Installing Liberty
Encrypting passwords (optional)
Creating and populating the database
Installing the DWC server
Defining z/OS engine in Z connector

You can optionally encrypt the passwords that you will use while installing, upgrading, and managing HCL Workload Automation. The secure command uses the AES method and prints the encrypted password to the screen or saves it to a file.

Note: It is important you understand the limits to the protection that this method provides. The custom passphrase you use to encrypt the passwords is stored in clear format in the `passphrase_variables.xml` file, stored in `configureDropin`. To fully understand the implications of this method, it is recommended you read the information provided by Open Liberty at the link Password encryption limitations.

You can perform a typical procedure, which uses a custom passphrase, as described in the following scenario. For more information about all secure arguments and default values, see Optional password encryption - secure script on page 284.

Encrytping the password

1. Browse to the folder where the secure command is located:
   - Before the installation, the command is located in the product image directory, `<`*`image_directory`*`>/TWS/<`*`op_sys`*`>/Tivoli_LWA_<`*`op_sys`*`>/TWS/bin`
   - After the installation, the command is located in `TWA_home/TWS/bin`
2. Depending on your operating system, encrypt the password as follows:

   Windows **Windows operating systems**

   ```
   secure -password password -passphrase passphrase
   ```

   UNIX **UNIX operating systems**

   ```
   ./secure -password password -passphrase passphrase
   ```

   z/OS **z/OS operating systems**

   ```
   ./secure -password password -passphrase passphrase
   ```

   where

-password

Specifies the password to be encrypted.

-passphrase

Specifies the custom passphrase that is used to generate the key with which the command encrypts the password. If you set this parameter, inform the user who installs HCL Workload Automation that they must define the **SECUREWRAP_PASSPHRASE** environment variable in the same shell from which they run the installation command, and set it to the same value as the **passphrase** parameter. On Windows operating systems, the passphrase must be at least 8 characters long. This argument generates a password which can be reused for all HCL Workload Automation components. This parameter is mutually exclusive with the -useaeskeystore on page 287 parameter, which generates a password which can be decrypted only on the local workstation and not reused for other components.

3. Provide both the encrypted password and custom passphrase to the user in charge of installing HCL Workload Automation. You can use encrypted passwords only in association with the specific passphrase used to encrypt them.

Installing with the encrypted password

The user in charge of installing HCL Workload Automation must set the **SECUREWRAP_PASSPHRASE** environment variable by performing the following steps:

1. Open a brand new shell session.
2. Ensure that no value is set for the **SECUREWRAP_PASSPHRASE** environment variable.
3. Define the **SECUREWRAP_PASSPHRASE** environment variable and set it to the passphrase defined by the user who ran the secure command, as follows:

   ```
   SECUREWRAP_PASSPHRASE=<passphrase>
   ```

   You can use encrypted passwords only in association with the specific passphrase used to encrypt them.
4. In the same shell session, provide the encrypted passwords when running any command that uses a password. An encrypted password looks like the following example:

   ```
   {aes}AFC3jj9cROYyqR+3CONBzVi8deLb2Bossb9GGroh8UmDPGikIkzXZzid3nzY0IhnSg=
   ```

Optional password encryption - secure script

Optionally encrypt the passwords you use to install, upgrade, and manage HCL Workload Automation.

This section lists and describes the parameters of the secure script. The secure command uses the AES method and prints the encrypted password to the screen or saves it to a file. This command is available by default on all HCL Workload Automation components.

Note: Use this script only to encrypt passwords used during the installation and upgrade processes.

You can either:

- Define a custom passphrase by using the **passphrase** argument and defining the **SECUREWRAP_PASSPHRASE** environment variable in the same shell session in which you run the command using the encrypted password. Ensure you set the **SECUREWRAP_PASSPHRASE** environment variable to the same value as the **passphrase** argument. You can use encrypted passwords only in association with the specific passphrase used to encrypt them.
- Use the standard encryption method provided with the secure command. In this case, you simply specify the **password** parameter.

Note: It is important you understand the limits to the protection that this method provides. The custom passphrase you use to encrypt the passwords is stored in clear format in the `passphrase_variables.xml` file, stored in `configureDropin`. To fully understand the implications of this method, it is recommended you read the information provided by Open Liberty at the link Password encryption limitations.

Syntax

Windows **Windows operating systems:**

```
secure -fips on|weak|off | -checksecurity | -updatesecurity | -securitystatus | {-password password
| -in file} [-fipscompliance true|false] [-des3toaes]
[[-passphrase passphrase] | [-useaeskeystore]] [-out file]
```

UNIX **UNIX operating systems:**

```
./secure -fips on|weak|off | -checksecurity | -updatesecurity | -securitystatus | {-password password
| -in file} [-fipscompliance true|false] [-des3toaes]
[[-passphrase passphrase] | [-useaeskeystore]] [-out file]
```

z/OS **z/OS operating systems:**

```
secure -fips on|weak|off | -checksecurity | -updatesecurity | -securitystatus | {-password password
| -in file} [-fipscompliance true|false] [-des3toaes]
[[-passphrase passphrase] | [-useaeskeystore]] [-out file]
```

Arguments

-fips

Specifies your FIPS settings:

on

Select `on` to enable FIPS in **full** mode. In this mode, FIPS enforces the most rigorous cryptographic levels defined by the FIPS 140-3 standard.

weak

Select `weak` to enable FIPS in **weak** mode. In this mode, FIPS enforces the most rigorous cryptographic levels defined by the FIPS 140-3 standard, but supports also the SHA-1 and 3DES algorithms.

off

Select `off` to disable FIPS. In this mode, FIPS standards are not enforced, but the product is still robust and secure.

-checksecurity

Checks the encryption level for password encryption. If user passwords are encrypted with the AES algorithm, the command modifies the **useAESEncryptionAlgorithm** optman option to `yes`. If the encryption algorithm is different from AES, the command displays a warning message.

-updatesecurity

Changes the encryption algorithm for user passwords from 3DES to AES. To prevent communication problems, this change requires that all components in your environment are at version 10.2.5 or later.

The command checks and modifies the **useAESEncryptionAlgorithm** optman option, based on the encryption algorithm used in your environment for user passwords, as follows:

users with passwords in 3DES

the **useAESEncryptionAlgorithm** option is set to `no`.

users with passwords in AES

the **useAESEncryptionAlgorithm** option is set to `yes`.

For more information about optman options, see the topic about global options in *Administration Guide*.

-securitystatus

Displays security settings in your environment, for example the current FIPS mode.

-password

Specifies the password to be encrypted. This parameter is mutually exclusive with the -**in** parameter.

-in

Specifies the name and path of the file where you have stored the password to be encrypted. This parameter is mutually exclusive with the -**password** parameter.

-fipscompliance

Allows overriding the product's FIPS mode. For instance, if a master domain manager, agent, or Dynamic Workload Console has FIPS enabled and the option **-fipscompliance** `=false` is specified, FIPS is selectively disabled for the secure command. Conversely, passing **-fipscompliance** `=true` enforces FIPS for that command, regardless of the global setting.

-des3toaes

Converts the specified password from the Triple DES to the AES format.

-passphrase

Specifies the custom passphrase that is used to generate the key with which the command encrypts the password. If you set this parameter, inform the user who installs HCL Workload Automation that they must define the **SECUREWRAP_PASSPHRASE** environment variable in the same shell from which they run the installation command, and set it to the same value as the **passphrase** parameter. On Windows operating systems, the passphrase must be at least 8 characters long. This argument generates a password which can be reused for all HCL Workload Automation components. This parameter is mutually exclusive with the -useaeskeystore on page 287 parameter, which generates a password which can be decrypted only on the local workstation and not reused for other components.

-useaeskeystore

Specifies that the secure command runs the encryption process using the AES keystore specified in the **encrypt keystore file** option and associated to the **encrypt label** alias. Both options are defined in the `localopts` file. The keystore is created automatically at installation time. Using this parameter ensures that passwords are encrypted with a unique key for each installation. Consequently, files encrypted on one component cannot be decrypted on another component due to differing encryption keys. For more information about the **encrypt keystore file** option and the **encrypt label** alias, see the topic about localopts details in *Administration Guide*. This parameter is mutually exclusive with the -passphrase on page 286 parameter, which generates a password which can be reused for other components.

-base64 e

Specifies that the encoding process uses the **base64** format.

-out

Specifies the path and name of a file where the command stores the encrypted password. If you do not specify this parameter, the encrypted password is printed to the screen.

Examples

To encrypt password `MyPassword` with a strong passphrase, run the following command:

```
./secure -password MyPassword -passphrase de85pU!Mb5G2xewPgdVa
```

To encrypt the password stored in file `MyFile` using the default passphrase and save the encrypted password to file `OutputFile`, run the following command:

```
secure -in C:\info\MyFile -out C:\info\OutputFile
```

Installing z-centric agent using twsinst

You use the **twsinst** script to install the agent with z-centric capabilities.

Optionally, you can add to the agent with z-centric capabilities:

- Dynamic capabilities, to provide your distributed environment with dynamic scheduling capabilities.
- Java™ run time to run job types with advanced options, both those supplied with the product and the additional types implemented through the custom plug-ins. The run time environment also enables the capability to remotely run, from the agent, the dynamic workload broker resource command on the server.

For a complete list of the supported operating systems, see the Detailed System Requirements document at HCL Workload Automation Detailed System Requirements.

twsinst

Before you begin

Ensure that you have the required authorization requirements, as described in Authorization roles for running the twsinst script on page 282.

About this task

During the installation process, if you do not specify the installation directory in the command, twsinst creates files in the following directories for each of the installation steps:

On Windows™ operating systems

`%ProgramFiles%\HCL\TWA_`*`TWS_USER`*

On UNIX™ and Linux™ operating systems

`/opt/HCL/TWA_`*`TWS_USER`*

Where *`TWS_USER`* is the user for which you are installing the HCL Workload Automation instance that you specify in the command. If you stop and restart the installation, the installation process starts from the installation step where it was stopped.

To install the HCL Workload Automation Agent, perform the following steps:

On Windows™ operating systems

1. Download the HCL Workload Automation Agent eImage related to your operating system.
2. Log in as administrator on the workstation where you want to install the product.
3. In the *`image_directory`*`\TWS\`*`operating_system`*`\ACTIONTOOLS` directory, delete the `.distributed` file and create an empty file named `.zcentric`
4. From *`image_directory`*`\TWS\`*`operating_system`*, run **twsinst** using the syntax described in the following section.

   Note: twsinst for Windows™ operating systems is a Visual Basic Script (VBS) that you can run in CScript and WScript mode.

   The HCL Workload Automation for Z user is automatically created. The software is installed by default in the HCL Workload Automation for Z installation directory. The default value is `%ProgramFiles%\HCL \TWA`.

On UNIX™ and Linux™ operating systems

1. Download the HCL Workload Automation Agent eImage related to your operating system.
2. Create the HCL Workload Automation for Z user. The software is installed by default in the user's home directory, referred to as `/installation_dir/TWS`

   User:

   TWS_user

   Home:

   `/installation_dir/TWS` (for example: `/home/user1/TWS` where `user1` is the name of the HCL Workload Automation for Z user.)
3. Log in as root on the workstation where you want to install the product.
4. In the `image_directory/TWS/operating_system/ACTIONTOOLS` directory, delete the `.distributed` file and create an empty file named `.zcentric`
5. From the `image_directory/TWS/operating_system` directory, run **twsinst** using the syntax described in the following section.

A successful installation using the **twsinst** script issues the return code RC = 0. If the installation fails to understand the cause of the error, see Analyzing return codes for agent installation, upgrade, restore, and uninstallation on page 313.

Synopsis

On Windows™ operating systems

Show command usage and version

```
twsinst.vbs -u | -v
```

Install a new instance

```
twsinst.vbs -new -uname user_name
    -password user_password
    -agent zcentric
    [-addjruntime true|false]
    [-displayname agentname]
    [-domain user_domain]
    [-hostname hostname]
    [-inst_dir install_directory]
    [-jmport port_number]
    [-jmportssl true|false]
    [-lang lang_id]
    -sslkeysfolder
    -sslpassword
    [-skip_usercheck]
    [-stoponcheckprereq]
    [-work_dir working_directory]
    [-zhostname zconn_hostname
    [-zport zconn_portnumber]
```

On UNIX™ and Linux™ operating systems

Show command usage and version

```
twsinst -u | -v
```

Install a new instance

```
twsinst -new -uname user_name
    -agent zcentric
    [-addjruntime true|false]
    [-displayname agentname]
    [-hostname hostname]
    [-inst_dir install_directory]
    [-jmport port_number]
    [-jmportssl true|false]
    [-lang lang_id]
    -sslkeysfolder
    -sslpassword
    [-reset_perm]
    [-skip_usercheck]
    [-stoponcheckprereq]
    [-work_dir working_directory]
    [-zhostname zconn_hostname
    [-zport zconn_portnumber]
```

Parameters

-agent zcentric

Installs the HCL Workload Automation Agent.

-addjruntime *true|false*

Adds the Java™ run time to run job types with advanced options, both those types that are supplied with the product and the additional types that are implemented through the custom plug-ins. Valid values are **true** and **false**. The default for a fresh installation is **true**. Set this parameter to `true` if you use the **sslkeysfolder** and **sslpassword** parameters to define custom certificates in PEM format.

-domain *user_domain*

Windows™ operating systems only. The domain name of the HCL Workload Automation user. The default is the name of the workstation on which you are installing the HCL Workload Automation Agent.

-displayname *agentname*

The name to be assigned to the HCL Workload Automation Agent. The default is the host name.

-hostname *hostname*

The fully qualified host name or IP address on which the agent will be contacted by the dynamic workload broker.

-inst_dir *install_directory*

The directory where to install the HCL Workload Automation Agent. On UNIX™ and Linux™, this path cannot contain blanks. On Windows™ operating systems, if you specify a path that contains blanks, enclose it in double quotes. On any operating system, specify an absolute path. If you do not manually specify a path, the path is set to the default home directory. On UNIX™ and Linux™, the path is set to the *user_name* home directory, and on Windows™ operating systems it is set to .

-jmport ***port_number***

The port used by the HCL Workload Automation for Z controller or the dynamic workload broker to connect to the HCL Workload Automation Agent. The default value is **31114**. The valid range is from 1 to 65535.

-jmportssl ***true|false***

The port used by the HCL Workload Automation for Z controller, or by the dynamic workload broker to connect to the HCL Workload Automation Agent. This number is registered in the `ita.ini` file, located in `ITA\cpa\ita` on Windows™ operating systems and `ITA/cpa/ita` on UNIX™ and Linux™.

For communication using SSL or HTTPS

Set **jmportssl = true**. To communicate with the dynamic workload broker, it is recommended that you set the value to **true**. In this case, the port specified in **jmport** communicates in HTTPS. If you specify **true**, ensure that you also configure the HTTPS communication on the z/OS® master.

For communication without using SSL, or through HTTP

Set **jmportssl = false**. In this case the port specified in **jmport** communicates in HTTP.

The default value is **true**.

To increase the performance of the HCL Workload Automation for Z server, it is recommended that you set this value to **false**.

-lang ***lang_id***

The language in which the **twsinst** messages are displayed. If not specified, the system LANG is used. If the related catalog is missing, the default C language catalog is used.

Note: This is the language in which the installation log is recorded, and not the language of the installed engine instance. twsinst installs all languages as default.

-new

A fresh installation of the agent. Installs an agent and all supported language packs.

-password ***user_password***

Windows™ operating systems only. The password of the user for which you are installing HCL Workload Automation Agent. The password can include alphanumeric, dash (-), and underscore (_) characters, and the following symbols: ()!?=^*/~ [] $`+;:.,@.

-reset_perm

UNIX™ and Linux™ only. Reset the permissions of the libatrc library.

-skip_usercheck

Enable this option if the authentication process within your organization is not standard, thereby disabling the default authentication option. On UNIX™ and Linux™ operating systems if you specify this parameter, the program skips the check of the user in the `/etc/passwd` file or the check you perform using the su command. On Windows™ operating systems if you specify this parameter, the program does not create the user you specified in the -uname *username* parameter. If you specify this parameter you must create the user manually before running the script.

-sslkeysfolder *path*

The name and path of the folder on the agent containing PEM certificates. The installation program automatically generates the keystore and truststore files using the password you specify with the **sslpassword** parameter, which is **required** when using **sslkeysfolder**.

The folder must contain the following files and folders:

ca.crt

The Certificate Authority (CA) public certificate.

tls.key

The private key for the instance to be installed.

tls.crt

The public part of the previous key.

tls.sth

The file storing your encoded password in Base64 encoding.

You can optionally create a subfolder to contain one or more `*.crt` files to be added to the server truststore as trusted CA. This can be used for example to add to the list of trusted CAs the certificate of the LDAP server or DB2 server. Additionally, you can store here any intermediate CA certificate to be added to the truststore. The subfolder must be named **additionalCAs**. If you are connecting a master domain manager using custom certificates to a dynamic agent also using custom certificates, the only required file is `ca.crt`.

Before you start the installation, ensure the required files and folders are available on the agent.

The **sslkeysfolder** and **sslpassword** parameters are **mutually exclusive** with the **wauser**, **wapassword**, **apikey**, and **jwt true** parameters, which are used to download and deploy the certificates or JWT already available on the master domain manager.

For a comprehensive list of supported combinations for this parameter and related ones, see Table 2.

-sslpassword *password*

Specify the password for the certificates in PEM format automatically generated by the installation program. It requires the **sslkeysfolder** parameter.

If you use this parameter, ensure that the **addjruntime** parameter is set to true, because Java™ run time is required for defining custom certificates.

You can optionally encrypt the password by using the secure script. For more information, see Encrypting passwords (optional) on page 219.

-stoponcheckprereq

Stop the installation whenever a problem occurs during the prerequisite check. For more information about the prerequisites, see Product Requirements.

-u

Displays command usage information and exits.

-uname *user_name*

The name of the user for which the HCL Workload Automation Agent is installed. This user name is not to be confused with the user performing the installation logged on as **root** on UNIX™ and Linux™ and as **administrator** on Windows™ operating systems.

On UNIX™ and Linux™, this user account must be created manually before running the installation. Create a user with a home directory. By default, HCL Workload Automation Agent is installed in the home directory of the specified user.

-work_dir *working_directory*

The temporary directory used for theHCL Workload Automation installation process files deployment.

On Windows™ operating systems

If you specify a path that contains blanks, enclose it in double quotes. If you do not manually specify a path, the path is set to `%temp%\TWA\tws95`, where `%temp%` is the temporary directory of the operating system.

On UNIX™ and Linux™ operating systems

The path cannot contain blanks. If you do not manually specify a path, the path is set to `/tmp/TWA/tws95`.

-v

Displays the command version and exits.

-zhostname *zconn_hostname*

The fully qualified host name of the Z connector (this is coincident with the Dynamic Workload Console host name). It is used together with the `-zport zconn_port` parameter. It adds the capability to download the plug-ins from the Z connector.

This value is registered in the ResourceAdvisorAgent property of the `JobManager.ini` file.

-zport *zconn_portnumber*

The HTTP or HTTPS port number of the Z connector (this is coincident with the Dynamic Workload Console port number). It is used together with the `-zhostname zconn_hostname` parameter to add the capability to download the plug-ins from the Z connector. The valid range is from 1 to 65535.

This value is registered in the ResourceAdvisorAgent property in the `JobManager.ini` file.

Example

Examples

This example describes how to install the HCL Workload Automation Agent for the user `HWA_user` and accept the default value to add the runtime environment for Java™. The runtime environment is used to run jobs supplied with the product or implemented through the custom plug-ins, it also enables the capability to remotely run, from the agent, the dynamic workload broker resource command on the server.

On Windows™ operating systems

```
twsinst.vbs -new
-uname HWA_user
-password qaz12qaz
-jmportssl false
-jmport 31114
-inst_dir "install_directory"
-sslkeysfolder certs_path
-sslpassword pwd
```

On UNIX™ and Linux™ operating systems

```
./twsinst -new
-uname HWA_user
-jmportssl false
-jmport 31114
-inst_dir install_directory
-sslkeysfolder certs_path
-sslpassword pwd
```

The twsinst log files

About this task

The twsinst log file name is:

Windows **On Windows operating systems:**

`<TWS_INST_DIR>\logs\twsinst_operating_system_TWS_user^version_number.log`

Where:

TWS_INST_DIR

The HCL Workload Automation installation directory. The default installation directory is `C:\Program Files\HCL\TWA_TWS_user`.

operating_system

The operating system.

TWS_user

The name of the user for which HCL Workload Automation was installed, that you supplied during the installation process.

UNIX **On UNIX operating systems:**

`<`*`TWS_INST_DIR`*`>/TWSDATA/installation/logs/`
`twsinst_`*`operating_system_TWS_user^product_version_number`*`.log`

Where:

TWS_INST_DIR

The HCL Workload Automation installation directory. The default installation directory is `/opt/HCL/TWA_`*`TWS_user`*.

operating_system

The operating system.

TWS_user

The name of the user for which HCL Workload Automation was installed, that you supplied during the installation process.

Enabling dynamic capabilities after installation or upgrade

This section describes the procedure that you must perform to enable dynamic scheduling capabilities after you installed or upgraded the HCL Workload Automation Agent, without enabling them:

1. Update the `JobManager.ini` configuration file located in:

   Windows™ operating systems:

   `tws_home``\TWS\ITA\cpa\config\JobManager.ini`

   UNIX™ and Linux™ operating systems:

   `tws_home``/TWS/ITA/cpa/config/JobManager.ini`

   by assigning to the *tdwb_hostname* and *mdm_httpsport* variables contained in the ResourceAdvisorUrl property, the following values:

   tdwb_hostname

   The fully qualified host name of the workload broker server.

   mdm_httpsport

   The value that the **httpsPort** has on the master domain manager, as shown by the **showHostPorperties** wastool. The default is 31116, which is the dynamic workload broker port number. The port is currently set to zero because at installation time you specified that you would not use the dynamic workload broker.

The **ResourceAdvisorUrl** property has the following syntax:

```
ResourceAdvisorUrl = https://tdwb_hostname:mdm_httpsport
/JobManagerRESTWeb/JobScheduler/resource
```

2. Start the HCL Workload Automation Agent by running the following command from `TWS_home`:

   Windows™ operating systems:

   `StartUpLwa.cmd`

   UNIX™ and Linux™ operating systems:

   `StartUpLwa`

Deploying with containers

Docker is a state-of-the-art technology that creates, deploys, and runs applications by using containers. Packages are provided containing an application with all of the components it requires, such as libraries, specific configurations, and other dependencies, and deploy it in no time on any UNIX, Windows, and Linux on Z workstation, regardless of any different settings between the source and the target workstation.

Docker adoption ensures standardization of your workload scheduling environment and provides an easy method to replicate environments quickly in development, build, test, and production environments, speeding up the time it takes to get from build to production significantly. Install your environment using Docker to improve scalability, portability, and efficiency.

To deploy HCL Workload Automation using a Docker container, proceed as follows:

1. Access and download the Docker image from the entitled registry. For details about the complete procedure, see Deploying containers with Docker on page 304.
2. You can choose to deploy all product containers with a single command, or you can deploy each product component container individually. Start and configure the HCL Workload Automation containers. For details about the complete procedure, see Deploying containers with Docker on page 304.

   More detailed technical information for each component can be found in the sample readme files:
   - HCL Workload Automation Server
   - HCL Workload Automation Console
   - HCL Workload Automation dynamic agent
   - HCL Workload Automation z-centric agent
   - Workload Automation FileProxy
3. Access the container to verify the status and run HCL Workload Automation commands. For details, see Accessing the Docker containers on page 304.

You can also use docker containers to store all the latest integrations available on Automation Hub. For further information, see the Container plug-in.

Deploying z-centric and dynamic agents on Kubernetes

To ensure a fast and responsive experience when using HCL Workload Automation for Z, you can deploy z-centric and dynamic agents on a Kubernetes cluster. Kubernetes is a portable, extensible, open-source platform for managing containerized workloads and applications.

A cloud deployment ensures access anytime anywhere and is a fast and efficient way to get up and running quickly. It also simplifies maintenance, lowers costs, provides rapid upscale and downscale, minimizes IT requirements and physical on-premises data storage.

Orchestrating your Kubernetes workload from a containerized agent optimizes allocations of resources in your cluster and reduces the chance of human errors. Find out more about the advantages of a stable containerized agent in the Orchestrating your workload on cloud from IBM Z Workload Scheduler video, available on the Workload Automation YouTube channel.

Deployment of the z-centric and dynamic agent containers on Kubernetes is supported on the following cloud provider infrastructures. For details and deployment instructions, see the HCL Workload Automation z-Centric Chart readme file.

Amazon Web Services (AWS) Elastic Kubernetes Service (EKS) (Amazon EKS)

The z-centric and dynamic agents are offered on the Amazon Web Services cloud. Within just a few minutes, access the product Helm chart and container images and easily launch an instance to deploy an agent with full on-premises capabilities.

Note: At agent installation completion, the AWS cluster returns a host name 77 characters long. Because in the ROUTOPTS statement the maximum allowed length for host names is 52 characters, ensure that you map the host name returned by AWS to a name no longer than 52 characters.

Azure Kubernetes Service (AKS)

Deploy and manage HCL Workload Automation containerized product components on the Azure AKS, a container orchestration service available on the Microsoft Azure public cloud. You can use Azure AKS to deploy and manage containers in the cluster environment.

Google GKE

Google Kubernetes Engine (GKE) provides a managed environment for deploying, managing, and scaling your containerized applications using Google infrastructure. The Google GKE environment consists of multiple machines grouped together to form a cluster.

Red Hat OpenShift

The HCL Workload Automation product components can be deployed onto a Red Hat OpenShift, V4.*x* environment by using helm charts.

Prerequisites

Prerequisite information when deploying with containers.

When deploying the product using containers, ensure you have fulfilled the following prerequisites:

Check the prerequisites of the command-line installation method in Prerequisites.

If you want to calculate the necessary resources that the agent container needs to run, use the following formula:

Evaluate the volume_size variable:

```
Volume size(MB)=
        120 + [ 30 x jobs_per_day x (average_joblog_size_MB / 3 + 0.008) ]
```

For example, considering "average_joblog_size_MB = 0.001 MB (1KB)", you obtain:

```
1.000
        jobs_per_day: 370 MB --> volume_size = 370Mi
```

```
10.000
        jobs_per_day: 2.6 GB --> volume_size = 2600Mi
```

```
100.000
        jobs_per_day: 25 GB --> volume_size = 25Gi
```

Deploying Docker compose on Linux on Z

Before you deploy HCL Workload Automation components on Linux on Z, ensure that you have deployed Docker compose, as explained in the following procedure.

To deploy the containers, docker-compose is required on the local workstation. Perform the following steps:

1. Browse to `/usr/local/bin` and create a file with name `docker-compose` with the following contents:

```
#
# This script will attempt to mirror the host paths by using volumes for the
# following paths:
#   * $(pwd)
#   * $(dirname $COMPOSE_FILE) if it's set
#   * $HOME if it's set
#
# You can add additional volumes (or any docker run options) using
# the $COMPOSE_OPTIONS environment variable.
#


set -e

VERSION="1.27.4"
IMAGE="ibmcom/dockercompose-s390x:$VERSION"


# Setup options for connecting to docker host
if [ -z "$DOCKER_HOST" ]; then
    DOCKER_HOST='unix:///var/run/docker.sock'
fi
if [ -S "${DOCKER_HOST#unix://}" ]; then
    DOCKER_ADDR="-v ${DOCKER_HOST#unix://}:${DOCKER_HOST#unix://} -e DOCKER_HOST"
```

```
else
    DOCKER_ADDR="-e DOCKER_HOST -e DOCKER_TLS_VERIFY -e DOCKER_CERT_PATH"
fi



# Setup volume mounts for compose config and context
if [ "$(pwd)" != '/' ]; then
    VOLUMES="-v $(pwd):$(pwd)"
fi
if [ -n "$COMPOSE_FILE" ]; then
    COMPOSE_OPTIONS="$COMPOSE_OPTIONS -e COMPOSE_FILE=$COMPOSE_FILE"
    compose_dir="$(dirname "$COMPOSE_FILE")"
    # canonicalize dir, do not use realpath or readlink -f
    # since they are not available in some systems (e.g. macOS).
    compose_dir="$(cd "$compose_dir" && pwd)"
fi
if [ -n "$COMPOSE_PROJECT_NAME" ]; then
    COMPOSE_OPTIONS="-e COMPOSE_PROJECT_NAME $COMPOSE_OPTIONS"
fi
if [ -n "$compose_dir" ]; then
    VOLUMES="$VOLUMES -v $compose_dir:$compose_dir"
fi
if [ -n "$HOME" ]; then
    VOLUMES="$VOLUMES -v $HOME:$HOME -e HOME" # Pass in HOME to share docker.config and allow
 ~/-relative paths to work.
fi
i=$#
while [ $i -gt 0 ]; do
    arg=$1
    i=$((i - 1))
    shift

    case "$arg" in
        -f|--file)
            value=$1
            i=$((i - 1))
            shift
            set -- "$@" "$arg" "$value"

            file_dir=$(realpath "$(dirname "$value")")
            VOLUMES="$VOLUMES -v $file_dir:$file_dir"
        ;;
        *) set -- "$@" "$arg" ;;
    esac
done

# Setup environment variables for compose config and context
ENV_OPTIONS=$(printenv | sed -E "/^PATH=.*/d; s/^/-e /g; s/=.*//g; s/\n/ /g")

# Only allocate tty if we detect one
if [ -t 0 ] && [ -t 1 ]; then
    DOCKER_RUN_OPTIONS="$DOCKER_RUN_OPTIONS -t"
fi

# Always set -i to support piped and terminal input in run/exec
DOCKER_RUN_OPTIONS="$DOCKER_RUN_OPTIONS -i"
```

```
# Handle userns security
if docker info --format '{{json .SecurityOptions}}' 2>/dev/null | grep -q 'name=userns'; then
    DOCKER_RUN_OPTIONS="$DOCKER_RUN_OPTIONS --userns=host"
fi

# shellcheck disable=SC2086
exec docker run --rm $DOCKER_RUN_OPTIONS $DOCKER_ADDR $COMPOSE_OPTIONS $ENV_OPTIONS $VOLUMES -w "$(pwd)"
 $IMAGE "$@"
```

2. Run the following command to make the `docker-compose` file an executable file:

```
sudo chmod +x /usr/local/bin/docker-compose
```

3. More detailed technical information for each component are available in the sample readme files:
    - HCL Workload Automation Server
    - HCL Workload Automation Console
    - HCL Workload Automation dynamic agent
    - HCL Workload Automation z-centric agent
    - Workload Automation FileProxy

Deploying Docker containers on IBM zCX

Before you deploy HCL Workload Automation components on an IBM zCX instance, ensure that you have:

- The IBM z/OS Management Facility (z/OSMF) up and running.
- An IBM zCX instance where you configured the `workflow_variables.properties` file (by default, located in `/usr/lpp/zcx_zos/properties`) with all the input variables listed in Table 35: Properties for Workflow on page 300.

Table 35. Properties for Workflow

Input Variables	Variable Name	IBM z/OSMF field
zCX General Configuration	ZCX_INSTALL_DIR	Install Directory
	ZCX_INSTNAME	zCX Instance Name
	ZCX_REGISTRY_DIR	zCX Instance Registry Directory
	ZCX_CTRACE_NAME	CTRACE Member Name
	ZCX_TEMP_DASD_UNIT_PARM	Temporary DASD Unit Parm
	ZCX_SAVE_PROPERTIES	Directory for saving input properties file
	ZCX_UNIQUE_JOBNAMES	Use unique job names for submitted jobs
	ZCX_UNIQUE_JOBNAME_PREFIX	Job name prefix
zCX CPU and Memory Configuration	ZCX_CPUS	Guest CPUs

Input Variables	Variable Name	IBM z/OSMF field
	ZCX_MEMGB	Guest Memory Size (in GB)
	ZCX_PAGE_FRAME_SIZE	Page Frame Size
zCX Network Configuration	ZCX_HOSTNAME	Hostname
	ZCX_GUESTIPV4 (DVIPa 4)	Guest IPV4 Address
	ZCX_MTU	MTU Size
	ZCX_HOSTDNS1	Primary DNS Server
zCX Root and Config Storage Configuration	HLQ	HLQ
	ZCX_ROOTMB	Root Size (in MB)
	ZCX_ROOTVOLSER	Root Volume Serial
	ZCX_CONFIGMB	Config Size (in MB)
	CX_CONFIGVOLSER	Config Volume Serial
zCX Instance Directory Storage Configuration	ZCX_ZFS_FILESYSTEM_HLQ	zFS Filesystem HLQ
	ZCX_ZFS_ENCRYPT	Encrypt zFS Filesystem
	ZCX_ZFS_VOLUME	zFS Filesystem Volume Serial
	ZCX_ZFS_PRIMARY_MEGABYTES	Primary Megabytes
	ZCX_ZFS_SECONDARY_MEGABYTES	Secondary Megabytes
zCX Swap Data Storage Configuration	ZCX_CREATE_SWAP	Create Swap Data Volume
	ZCX_SWAPMB	Swap Data Size (in MB)
	ZCX_SWAP_COUNT	Swap Data Volume Count
	ZCX_SWAPVOLSER	Swap Data Volume Serial
zCX User Data Storage Configuration	ZCX_DATAMB	User Data Size (in MB)
	ZCX_DATA_COUNT	User Data Volume Count
	ZCX_DATAVOLSER	User Data Volume Serial
zCX Diagnostics Data Storage Configuration	ZCX_DLOGSMB	Dlogs Data Size (in MB)
	ZCX_DLOGS_COUNT	Dlogs Data Volume Count
	ZCX_DLOGSVOLSER	Dlogs Data Volume Serial
zCX Docker Configuration	ZCX_DOCKER_LOGLEVEL	Docker Daemon Logging Level
	ZCX_DOCKER_LOGDRIVER	Docker Logging Driver

Input Variables	Variable Name	IBM z/OSMF field
	ZCX_SECURE_DOCKER_REGISTRY_ENABLE	Secure Docker Registry
zCX Proxy Configuration	ZCX_CONFIGURE_PROXY	Configure Proxy Server
zCX Docker User Management Configuration	ZCX_DOCKER_ADMIN	Docker Admin User ID
	ZCX_DOCKER_ADMIN_SSH_KEY	Docker Admin SSH Key
	ZCX_SSHD_LDAP_ENABLE	Enable LDAP Authentication
	ZCX_SSHD_LDAP_ENABLE_TLS	Enable LDAP Client TLS Authentication
zCX ILMT Configuration	ZCX_ILMT_ENABLE	Enable ILMT services

To deploy the HCL Workload Automation for Z containers to the IBM zCX instance, perform the following procedure:

1. Start the IBM zCX instance as follows:
    a. From z/OSMF, copy the command provided in the last step of the workflow output. The command looks like the following example:

    ```
    S GLZ,JOBNAME=<ZCX_INSTNAME>,CONF='<ZCX_REGISTRY_DIR>/start.json'
    ```

    where `<ZCX_INSTNAME>` and `<ZCX_REGISTRY_DIR>` are the variables that you set in `workflow_variables.properties`.

    b. Log in to your z/OS system, and from the ISPF Primary Option Menu panel issue the `SDSF` command.

    The SDSF menu opens.

    c. Issue the `ST` command. The SDSF STATUS DISPLAY panel opens.

    d. Issue the command `/` to display the SYSTEM COMMAND EXTENSION panel.

    e. From the command line, enter the command that you copied from the z/OSMF workflow output.

    You are returned to the SDSF STATUS DISPLAY, where `<ZCX_INSTNAME>` is added to the list of jobs as in EXECUTION.

    f. When the job starts processing, the following message is stored in the job' SYSPRINT log showing that the zCX instance is up and running on the indicated port and Dynamic Virtual IP address (DVIPA):

    ```
    Please Connect to IBM z/OS Container Extensions Docker CLI via your SSH client using port
     <port_number>.
    The server is listening on: <ZCX_GUESTIPV4>
    ```

2. Log in to the zCX instance:
    a. From the ISPF command shell of your z/OS system, access the UNIX interface by issuing the command: `omvs`

    b. From the command line issue the command:

    ```
    ssh -i <keydir_path>/.ssh/id_rsa -p <port_number> admin@<ZCX_GUESTIPV4>
    ```

    where:

`<keydir_path>`

Path where the `<ZCX_DOCKER_ADMIN_SSH_KEY>` that you set in `workflow_variables.properties` is stored.

`<port_number>`

Port indicated by the message stored in the job' SYSPRINT log.

`<ZCX_GUESTIPV4>`

Value set for the corresponding variable in `workflow_variables.properties`

c. At first logon, a warning message about the authenticity of the host is displayed. This is an expected ssh behavior, enter `Yes`.

The following message is displayed:

```
Welcome to the IBM z/OS Container Extensions (IBM zCX) shell
that provides access to Docker commands.
```

You are now logged in to the zCX instance. You can start importing and deploying containers.

To import the HCL Workload Automation for Z containers into the zCX instance, perform the following steps:

1. From the SDSF STATUS DISPLAY panel, enable the FTP process to the zCX instance by issuing the command:

```
/S FTPSERVE
```

2. To perform an SFTP, from your ssh session issue the command:

```
sftp -i <keydir_path>/.ssh/id_rsa -P <port_number> admin@<ZCX_GUESTIPV4>
```

where:

`<keydir_path>`

Path where the `<ZCX_DOCKER_ADMIN_SSH_KEY>` that you set in `workflow_variables.properties` is stored.

`<port_number>`

Port indicated by the message stored in the job' SYSPRINT log.

`<ZCX_GUESTIPV4>`

Value set for the corresponding variable in `workflow_variables.properties`

The message `Connected to <ZCX_GUESTIPV4>` is displayed.

3. Import the HCL Workload Automation for Z container by issuing the command:

```
put <container_name>.tar /tmp
```

The message `Uploading <container_name>.tar to /tmp/<container_name>.tar` is displayed.

After importing and deploying the containers, to access the container shell and run HCL Workload Automation for Z commands see Accessing the Docker containers on page 304.

Deploying containers with Docker

This topic describes how to deploy the current version of HCL Workload Automation by using Docker containers.

About this task

The following Docker containers are available:

- HCL Workload Automation Console, containing the Dynamic Workload Console image for UNIX, Windows, Linux on Z operating systems, and the IBM z/OS Container Extensions (zCX) feature.
- HCL Workload Automation dynamic agent and the image of the agent with the machine learning engine, containing the Agent image for UNIX, Windows, Linux on Z operating systems. and the IBM z/OS Container Extensions (zCX) feature.
- HCL Workload Automation z-centric agent, containing the Agent image for UNIX, Windows, Linux on Z operating systems. and the IBM z/OS Container Extensions (zCX) feature.

Each container package includes also a `docker-compose.yml` file to configure your installation.

You can choose to deploy all product containers with a single command, or you can deploy each product component container individually.

Deploying all product component containers with a single command

The following readme file contains all the steps required to deploy all product components at the same time: HCL Workload Automation

Deploying each product component container individually

If you want to install server, console and agent containers individually, see the related readme files :

- HCL Workload Automation Server
- HCL Workload Automation Console
- HCL Workload Automation dynamic agent
- HCL Workload Automation z-centric agent
- Workload Automation FileProxy

Note: The database is always external to the Docker engine and is not available as a container

Note: When deploying the server (master domain manager) container, the database schema is automatically created at the container start. If you are planning to install both the HCL Workload Automation server master domain manager and backup master domain manager, ensure that you run the command for one component at a time. To avoid database conflicts, start the second component only when the first component has completed successfully.

Accessing the Docker containers

This topic shows you how to access the container shell and run HCL Workload Automation commands.

To check the container status and run HCL Workload Automation commands, you need to access the containers as described below:

1. Obtain the container ID by running the following command: docker ps

   An output similar to the following one is returned:

   ```
   CONTAINER ID        IMAGE        NAMES               ........        .......

   b02459af2b9c        ......       wa-console          ........        .......
   ```
2. Access the Docker container by running the following command: docker exec -it <container_id> /bin/bash

   Where

   container_id

      Is the ID of the container obtained with the command explained in the first step, for example **b02459af2b9c**.

Creating a Docker image to run HCL Workload Automation Agents

Quickly create a Docker image to run HCL Workload Automation Agents.

You can run HCL Workload Automation Agents in a Docker container that you use to run jobs remotely, for example to call REST APIs or database stored procedures, or to run jobs within the container itself.

To create a Docker container, you are provided with step-by-step instructions and the latest versions of the required samples on GitHub here. Follow the instructions to create a Docker container to run jobs remotely, or use it as base image to add the applications to be run with the agent to other images, or customize the samples to best meet your requirements.

Chapter 13. Upgrading the HCL Workload Automation Agent

Upgrading HCL Workload Automation Agent (also known as z-centric agent) from version 9.4.*x* or 9.5.*x* to 10.2.*x*.

Before beginning the upgrade procedure ensure that you have the required authorization roles, as described in User authorization requirements.

Coexistence with previous versions

The current version of the HCL Workload Automation Agent (z-centric) can be installed on any workstation containing an earlier version, provided that the `TWS_user`, **JobManager** port, and installation path are different from those of the previous versions.

Upgrading notes

Before upgrading the HCL Workload Automation Agent, ensure that there are no jobs running on the agent.

If you are upgrading HCL Workload Automation Agent from an installation where you did not install the dynamic capabilities, you cannot add them during the upgrade process. To add them, perform the procedure described in the following section:

- Enabling dynamic capabilities after installation or upgrade on page 295

When the upgrade procedure is successful, it is not possible to roll back to the previous version. Rollback is possible only for upgrades that fail.

Upgrading using twsinst

Use **twsinst** to perform a direct upgrade of HCL Workload Automation Agent by satisfying the following objectives:

Save time, disk space, and RAM when upgrading the product

It saves disk space and RAM because it is not Java-based.

Use a very simple command

It consists of a single line command.

Manage both UNIX™ and Windows™ operating system workstations

It runs both on UNIX™ and Windows™ agents.

For a list of the supported operating systems and requirements, see HCL Workload Automation Detailed System Requirements.

For detailed upgrade steps, see *HCL Workload Automation: Planning and Installation*.

Examples

To upgrade the agent installed for the user `TWS_user` in the user home directory that does not have the dynamic scheduling capabilities and the Java™ runtime to run job types with advanced options, run the following command:

```
./twsinst -update -uname TWS_user
```

The twsinst log files

About this task

The twsinst log file name is:

Windows **On Windows operating systems:**

`<TWS_INST_DIR>\logs\twsinst_operating_system_TWS_user^version_number.log`

Where:

TWS_INST_DIR

The HCL Workload Automation installation directory. The default installation directory is `C:\Program Files\HCL\TWA_TWS_user`.

operating_system

The operating system.

TWS_user

The name of the user for which HCL Workload Automation was installed, that you supplied during the installation process.

UNIX **On UNIX operating systems:**

`<TWS_INST_DIR>/TWSDATA/installation/logs/twsinst_operating_system_TWS_user^product_version_number.log`

Where:

TWS_INST_DIR

The HCL Workload Automation installation directory. The default installation directory is `/opt/HCL/TWA_TWS_user`.

operating_system

The operating system.

TWS_user

The name of the user for which HCL Workload Automation was installed, that you supplied during the installation process.

Enabling dynamic capabilities after upgrade

To enable dynamic scheduling after you upgraded the HCL Workload Automation Agent without enabling it, see Enabling dynamic capabilities after installation or upgrade on page 295.

Chapter 14. Uninstalling the HCL Workload Automation Agent

Uninstalling the z/centric agent does not remove the files created after the agent was installed, nor files that are open at the time of uninstallation. If you do not need these files, remove them manually. If you intend to reinstall and therefore need the files, make a backup before starting the installation process.

Before beginning the uninstallation procedure, ensure that you have the required authorization roles as described in User authorization requirements.

Uninstalling the HCL Workload Automation Agent using the twsinst script

Depending on your operating system, perform the following steps to uninstall the HCL Workload Automation Agent by using the twsinst script:

- Windows™ operating systems:
  1. Ensure that all HCL Workload Automation processes and services are stopped, and that there are no active or pending jobs.
  2. Log on as administrator on the workstation where you want to uninstall the product.
  3. **twsinst** for Windows™ operating systems is a Visual Basic Script (VBS) that you can run in CScript and WScript mode, from the `installation_dir\TWS` directory, run the twsinst script as follows:

     ```
     cscript twsinst -uninst -uname user_name [-wait minutes]
     [-domain domain_name] [-lang lang_id]
     [-work_dir working_dir]
     ```

  The uninstallation is performed in the language of the locale and not the language set during the installation phase. If you want to uninstall agents in a language other than the locale of the computer, run the **twsinst** script from the /*installation_dir*/TWS (for example, /home/user1/TWS) as follows:

  ```
  cscript twsinst -uninst -uname user_name -lang language
  ```

  where *language* is the language set during the uninstallation.
- UNIX™ and Linux™ operating systems:
  1. Ensure that all processes and services are stopped, and that there are no active or pending jobs. For information about stopping the processes and services, see *Administration Guide*.
  2. Log on as root and change your directory to /*installation_dir*/TWS (for example: /home/user1/TWS where user1 is the name of HCL Workload Automation user.)
  3. From the `TWS` directory, run the twsinst script as follows:

     ```
     twsinst -uninst -uname user_name [-wait minutes]
     [-lang lang_id] [-work_dir working_dir]
     ```

  The uninstallation is performed in the language of the locale and not the language set during the installation phase. If you want to uninstall agents in a language other than the locale of the computer, run the **twsinst** script from the /*installation_dir*/TWS (for example, /home/user1/TWS) as follows:

  ```
  ./twsinst -uninst -uname user_name -lang language
  ```

  where *language* is the language set during the uninstallation.

-uninst

Uninstalls the agent

-uname *user_name*

The name of the user for which the agent is uninstalled. This user name is not to be confused with the user performing the installation logged on as **root** on UNIX™ and Linux™, and as **administrator** on Windows™ operating systems.

-wait *minutes*

The number of minutes that the product waits for jobs that are running to complete before starting the uninstallation. If the jobs do not complete during this interval the uninstallation stops and an error message is displayed. Valid values are integers or **-1** for the product to wait indefinitely. The default is **60** minutes.

-domain *domain_name*

Windows™ operating systems only. The domain name of the HCL Workload Automation user. The default is the name of the workstation on which you are uninstalling the product.

-lang *lang_id*

The language in which the `twsinst` messages are displayed. If not specified, the system LANG is used. If the related catalog is missing, the default C language catalog is used.

Note: The **-lang** option is not to be confused with the HCL Workload Automation supported language packs.

-work_dir *working_dir*

The temporary directory used for the installation process files deployment.

On Windows™ operating systems:

If you specify a path that contains blanks, enclose it in double quotation marks. If you do not manually specify a path, the path is set to `%temp%\TWA\tws93`, where `%temp%` is the temporary directory of the operating system.

On UNIX™ and Linux™ operating systems:

The path cannot contain blanks. If you do not manually specify a path, the path is set to `/tmp/TWA/tws93`.

The following is an example of a twsinst script that uninstalls the HCL Workload Automation agent, originally installed for user named **TWS_user**:

On Windows™ operating systems:

```
twsinst -uninst -uname TWS_user
```

On UNIX™ and Linux™ operating systems:

```
./twsinst -uninst -uname TWS_user
```

The twsinst log files

About this task

The twsinst log file name is:

Windows **On Windows operating systems:**

`<TWS_INST_DIR>\logs\twsinst_operating_system_TWS_user^version_number.log`

Where:

TWS_INST_DIR

The HCL Workload Automation installation directory. The default installation directory is `C:\Program Files\HCL\TWA_TWS_user`.

operating_system

The operating system.

TWS_user

The name of the user for which HCL Workload Automation was installed, that you supplied during the installation process.

UNIX **On UNIX operating systems:**

`<TWS_INST_DIR>/TWSDATA/installation/logs/twsinst_operating_system_TWS_user^product_version_number.log`

Where:

TWS_INST_DIR

The HCL Workload Automation installation directory. The default installation directory is `/opt/HCL/TWA_TWS_user`.

operating_system

The operating system.

TWS_user

The name of the user for which HCL Workload Automation was installed, that you supplied during the installation process.

Chapter 15. Troubleshooting and maintaining installation, upgrade, and uninstallation

Example

Return codes that you can receive when you are installing, upgrading, restoring, or uninstalling agents. To analyze them and take corrective actions, run the following steps:

On Windows operating systems

1. Display the operation completion return code, by using the following command:

```
echo %ERRORLEVEL%
```

2. Analyze the following table to verify how the operation completed:

Table 36. Windows operating system agent return codes

Error Code	Description	User action
0	Success: The operation completed successfully without any warnings or errors.	None.
1	Generic failure	Check the messages that are displayed on the screen by the script. Correct the error and rerun the operation. If the error persists, contact Support.
2	The installation cannot create the HCL Workload Automation user or assign the correct permission to it.	Verify the operating system policies and configuration. Verify the input values. If necessary, create the user manually before you run the installation.
3	The password is not correct or the installation cannot verify it.	Verify the operating system policies and configuration. Verify the input values.
4	The HCL Workload Automation installation directory is not empty. You specified as installation folder a directory that exists.	Empty it or specify a different directory.
5	An error occurred checking the HCL Workload Automation prerequisites on the workstation.	See the System Requirements Document at HCL Workload Automation Detailed System Requirements.
6	The HCL Workload Automation registry is corrupted.	Use the recovInstReg option to recover the registry. Then, rerun the operation.
7	The upgrade or restore operation cannot retrieve the information from the configuration files.	Check that the previous installation and the `localopts`, the `globalopts`, the

Error Code	Description	User action
		`ita.ini`, and the `JobManager.ini` files are not corrupted. Correct the errors and try again the operation.
8	The upgrade, restore, or uninstallation cannot proceed because there are jobs that are running.	Stop the jobs that are running or wait for these jobs to complete. Restart the operation.
9	The upgrade, restore, or uninstallation cannot proceed because there are files that are locked.	Stop all the processes that are running and close all the activities that can block the installation path. Restart the operation.
10	The upgrade, restore, or uninstallation cannot proceed because there are command lines opened.	Close the command lines. Restart the operation.

On UNIX and Linux operating systems:

1. Display the installation completion return code, by using the following command:

```
echo $?
```

2. Analyze the following table to verify how the installation completed:

Table 37. UNIX or Linux operating system agent return codes

Error Code	Description	User action
0	Success: The installation completed successfully without any warnings or errors.	None.
1	Generic failure.	Check the messages that are displayed on the video by the script. Correct the error and rerun the operation. If the error persists, contact Support.
2	The installation did not find the HCL Workload Automation user or its home directory. The HCL Workload Automation user that you specified either does not exist or does not have an associated home directory.	Verify the operating system definition of the HCL Workload Automation user.
3	Not applicable	

Error Code	Description	User action
4	The HCL Workload Automation installation directory is not empty. You specified as installation folder a directory that exists.	Empty it or specify a different directory.
5	An error occurred checking the HCL Workload Automation prerequisites on the workstation.	See the System Requirements Document at HCL Workload Automation Detailed System Requirements.
6	The HCL Workload Automation registry is corrupted.	Use the recovInstReg option to recover the registry. Then, rerun the operation.
7	The upgrade or restore operation cannot retrieve the information from the configuration files.	Check that the previous installation and the `localopts`, the `globalopts`, the `ita.ini`, and the `JobManager.ini` files are not corrupted. Correct the errors and try again the operation.
8	The upgrade, restore, or uninstallation cannot proceed because there are jobs that are running.	Stop the jobs that are running or wait for these jobs to complete. Restart the operation.
9	The upgrade, restore, or uninstallation cannot proceed because there are files that are locked.	Stop all the processes that are running and close all the activities that can block the installation path. Restart the operation.
10	The upgrade, restore, or uninstallation cannot proceed because there are command lines opened.	Close the command lines. Restart the operation.

Analyzing return codes for agent installation, upgrade, restore, and uninstallation

Check how your operation completed by analyzing the return codes that are issued by twsinst.

Return codes that you can receive when you are installing, upgrading, restoring, or uninstalling agents. To analyze them and take corrective actions, run the following steps:

On Windows operating systems

1. Display the operation completion return code, by using the following command:

```
echo %ERRORLEVEL%
```

2. Analyze the following table to verify how the operation completed:

Table 38. Windows operating system agent return codes

Error Code	Description	User action
0	Success: The operation completed successfully without any warnings or errors.	None.
1	Generic failure	Check the messages that are displayed on the screen by the script. Correct the error and rerun the operation. If the error persists, contact Support.
2	The installation cannot create the HCL Workload Automation user or assign the correct permission to it.	Verify the operating system policies and configuration. Verify the input values. If necessary, create the user manually before you run the installation.
3	The password is not correct or the installation cannot verify it.	Verify the operating system policies and configuration. Verify the input values.
4	The HCL Workload Automation installation directory is not empty. You specified as installation folder a directory that exists.	Empty it or specify a different directory.
5	An error occurred checking the HCL Workload Automation prerequisites on the workstation.	See the System Requirements Document at HCL Workload Automation Detailed System Requirements.
6	The HCL Workload Automation registry is corrupted.	Use the recovInstReg option to recover the registry. Then, rerun the operation.
7	The upgrade or restore operation cannot retrieve the information from the configuration files.	Check that the previous installation and the `localopts`, the `globalopts`, the `ita.ini`, and the `JobManager.ini` files are not corrupted. Correct the errors and try again the operation.
8	The upgrade, restore, or uninstallation cannot proceed because there are jobs that are running.	Stop the jobs that are running or wait for these jobs to complete. Restart the operation.
9	The upgrade, restore, or uninstallation cannot proceed because there are files that are locked.	Stop all the processes that are running and close all the activities that can

Error Code	Description	User action
		block the installation path. Restart the operation.
10	The upgrade, restore, or uninstallation cannot proceed because there are command lines opened.	Close the command lines. Restart the operation.

On UNIX and Linux operating systems:

1. Display the installation completion return code, by using the following command:

```
echo $?
```

2. Analyze the following table to verify how the installation completed:

Table 39. UNIX or Linux operating system agent return codes

Error Code	Description	User action
0	Success: The installation completed successfully without any warnings or errors.	None.
1	Generic failure.	Check the messages that are displayed on the video by the script. Correct the error and rerun the operation. If the error persists, contact Support.
2	The installation did not find the HCL Workload Automation user or its home directory. The HCL Workload Automation user that you specified either does not exist or does not have an associated home directory.	Verify the operating system definition of the HCL Workload Automation user.
3	Not applicable	
4	The HCL Workload Automation installation directory is not empty. You specified as installation folder a directory that exists.	Empty it or specify a different directory.
5	An error occurred checking the HCL Workload Automation prerequisites on the workstation.	See the System Requirements Document at HCL Workload Automation Detailed System Requirements.
6	The HCL Workload Automation registry is corrupted.	Use the recovInstReg option to recover the registry. Then, rerun the operation.

Error Code	Description	User action
7	The upgrade or restore operation cannot retrieve the information from the configuration files.	Check that the previous installation and the `localopts`, the `globalopts`, the `ita.ini`, and the `JobManager.ini` files are not corrupted. Correct the errors and try again the operation.
8	The upgrade, restore, or uninstallation cannot proceed because there are jobs that are running.	Stop the jobs that are running or wait for these jobs to complete. Restart the operation.
9	The upgrade, restore, or uninstallation cannot proceed because there are files that are locked.	Stop all the processes that are running and close all the activities that can block the installation path. Restart the operation.
10	The upgrade, restore, or uninstallation cannot proceed because there are command lines opened.	Close the command lines. Restart the operation.

Part VII. Installing HCL Workload Automation Agent on IBM i systems

To install and use the z/centric agent on IBM i systems, ensure that you have a supported version of the IBM i operating system. For a detailed list of the supported operating systems, see the HCL Workload Automation Detailed System Requirements.

Chapter 16. Installing agents on IBM i systems

Learn how to install agents on IBM i systems.

About this task

To install dynamic agents and z-centric agents on IBM i systems, use the `twsinst` installation script.

You can either use the default **QSECOFR** user or create a new user with **ALLOBJ authority**. If you plan to use a user different from **QSECOFR**, create the user before the installation and assign it the ALLOBJ authority.

For dynamic agents, you can also use a user **different from QSECOFR** with no specific authority. Note that only the user who performs the installation can use the agent.

Verify that the user profile used as **TWSUser** is not a member of a group profile. Set the group profile associated with the **TWSUser** to **NONE*. If the **TWSUser** is a member of a group, the installation might fail.

To install the agents, perform the following steps:

1. Sign on as the user of your choice, either **QSECOFR** or an **existing user with ALLOBJ authority**. If you use a user different from **QSECOFR**, specify the **allObjAuth** parameter to indicate that the specified user has the ALLOBJ authority. Ensure the user is existing and has ALLOBJ authority because the product does not verify that the correct authority is assigned. For more information about the **allObjAuth** parameter, see Agent installation parameters on IBM i systems on page 320. Only for dynamic agents, you can also use a user different from **QSECOFR** with no specific authority.
2. Create an IBM i user profile for which the HCL Workload Automation agent is installed.

   Note: The user profile is not the same as for the user that is performing the installation, unless you use a user different from **QSECOFR** with no specific authority (dynamic agents only). Instead the user profile is for the user that you specify in the **-uname** *username* parameter when running the twsinst script. For descriptions of the syntax parameters, see Agent installation parameters on IBM i systems on page 320. You cannot use an existing IBM i system user profile, an application supplied user profile, or any of the following reserved IBM i user profiles:

   - QDBSHR
   - QDFTOWN
   - QDOC
   - QLPAUTO
   - QLPINSTALL
   - QRJE
   - QSECOFR
   - QSPL

- QSYS
- QTSTRQS

Attention: Consider that:

- If the user profile is a member of a group, the installation fails. Set the group profile that is associated with the user profile to **NONE*.
- If the *username* is longer than 8 characters, after the installation the agent (and the JobManager component) runs under the **QSECOFR** user instead of under the authority of the installation user. To prevent this problem, set the `PASE_USRGRP_LIMITED` environment variable to N.

3. On the IBM i system, verify that no library exists with the same name as the user profile supplied for the agent user.
4. Download the installation images from Flexnet or from HCL Software.
5. To untar or unzip the agent eImage, you can use the *PASE* shell or the *AIXterm* command.

   Using *PASE* shell:

   a. Open the *PASE* shell.
   b. Run the command "CALL QP2TERM".
   c. Locate the folder where you downloaded the agent eImage and run the command:

      HCL Workload Automation Agent

      ```
      "tar xvf TWSversion_number>_IBM_I.tar"
      ```

      Dynamic Agent

      ```
      "unzip TWSversion_number>_IBM_I.zip"
      ```

   d. Exit from the *PASE* shell.

   Using *AIXterm* command:

   a. Start the *Xserver* on your desktop.
   b. On the iSeries machine, open a *QSH shell* and export the display.
   c. In *QSH shell*, go to the directory */QopenSys* and run the command "aixterm -sb".
   d. A pop-up window is displayed on your desktop. By Using this pop-up window, unzip the *TWSversion_number>_IBM_I.zip* file, or untar the *TWSversion_number>_IBM_I.tar* file.

6. If your machine's primary language is other than English, carry out these steps:
   a. Add English as secondary language.
   b. Ensure that when connecting to the environment the Host Code-Page is set to 037
   c. Before starting the installation, verify that the *Qshell session* is configured correctly and type the following command in the <yourfilename> :

      ```
      echo " key key2 " | sed 's/ *$//g' | sed 's/^ *//g'
      ```

   d. Run the <yourfilename>
   e. The environment is configured in the correct way if the output is: "key key2".

7. Open a *QSH shell* and run the twsinst script. During the installation process, the product creates an IBM i library and a job description with the same name as the user profile created in Step 2 on page 318.

   The installation procedure adds this library to the user profile library list of the dynamic agent user profile and sets this job description as the job description of the dynamic agent user profile. By default, the software is installed in the user's home directory.

If the installation fails to understand the cause of the error, see Analyzing return codes for agent installation, upgrade, restore, and uninstallation on page 313.

After a successful installation, perform the following configuration task:

- Configuring a dynamic agent, as described in *HCL Workload Automation: Planning and Installation*.

Command usage and version

Show command usage and version

```
twsinst -u | -v
```

Install a new instance

```
twsinst -new -uname username
   -acceptlicense yes|no
   [-addjruntime true|false]
   [-agent dynamic]
   [-allObjAuth]
   [-company company_name]
   [-displayname agentname]
   [-gateway local|remote|none]
   [-gweifport gateway_eif_port]
   [-gwid gateway_id]
   [-hostname hostname]
   [-inst_dir install_dir]
   [-jmport port_number]
   [-jmportssl true|false]
   [-lang lang_id]
   [-tdwbport tdwbport_number]
   [-tdwbhostname host_name]
   [-work_dir working_dir]
```

For a description of the installation parameters and options that are related to agent on this operating system, see Agent installation parameters on IBM i systems on page 320 in *HCL Workload Automation: Planning and Installation*.

Agent installation parameters on IBM i systems

About this task

The parameters set when using the **twsinst** script to install dynamic and z-centric agents on IBM i systems.

-acceptlicense *yes|no*

Specifies whether to accept the License Agreement.

-addjruntime *true|false*

Adds the Java™ run time to run job types with advanced options, both those types that are supplied with the product and the additional types that are implemented through the custom plug-ins. Valid values are **true** and **false**. The default for a fresh installation is **true**. Set this parameter to `true` if you use the **sslkeysfolder** and **sslpassword** parameters to define custom certificates in PEM format.

If you decided not to install Java™ run time at installation time, you can still add this feature later as it is described in Adding a feature.

-allObjAuth

If you are installing, upgrading, or uninstalling with a user different from the default **QSECOFR** user, this parameter specifies that the user has the required ALLOBJ authority. Ensure the user is existing and has ALLOBJ authority because the product does not verify that the correct authority is assigned. The same user must be specified when installing, upgrading or uninstalling the agent. If you are using the **QSECOFR** user, this parameter does not apply.

-apikey

Specifies the API key for authentication with the master domain manager. This key enables downloading certificates or JWT for communication between dynamic agent and dynamic domain manager. A random password in base64 encoding is automatically created for generating stash files. The password stored in the `tls.sth` file. If needed, you can decrypt this password using any base64 decoder.

Obtain the string to be provided with this parameter from the Dynamic Workload Console before running the command. For more information, see the section about authenticating the command line client using API Keys in *Dynamic Workload Console User's Guide*.

This parameter is **mutually exclusive** with:

- -wauser wauser_name
- -wapassword wauser_password
- -sslkeysfolder path
- -sslpassword password

and it is **required** with:

- -tdwbhostname host_name
- -tdwbport tdwbport_number

For a comprehensive list of supported combinations for this parameter and related ones, see Table 2.

-company *company_name*

The name of the company. The company name cannot contain blank characters. The name is shown in program headers and reports. If not specified, the default name is COMPANY.

-displayname *display_name*

The name to assign to the agent. The name cannot start with a number. The default is based on the host name of this computer.

If the host name starts with a number, the **-displayname** parameter must be specified.

-gateway *local|remote|none*

Specifies whether to configure a gateway to communicate with the dynamic workload broker or not, and how it is configured. Specify *`local`* if the gateway is local to the dynamic agent workstation. Specify *`remote`* if the dynamic agent communicates through a gateway that is installed on a different dynamic agent workstation from the dynamic agent being installed. The default value is *`none`*, which means no gateway is configured. For information about installing with a local and remote gateway, see Example installation commands.

-gweifport *gateway_eif_port*

Specifies the Job Manager Event Integration Facility (EIF) port number. The default value is **31132**. The valid range is 1 to 65535.

-gwid *gateway_id*

The unique identifier for the gateway. This parameter is required when you specify **-gateway** *`local`* and must be unique across all agents. The default gateway identifier that is assigned is **GW1**. The gateway identifier must start with either an alphabetic character or an underscore character (_), and it can contain only the following types of characters: alphabetic, numeric, underscores (_), hyphens (-), and periods (.).

Gateways can also work in parallel to mutually take over in routing communications to the agents connected to them. To enable gateways to work in parallel, all gateways must have the same *gateway_id* assigned. This information is stored in the `JobManagerGW.ini` file, by setting the **JobManagerGWURIs** property.

-hostname *host_name*

The fully qualified hostname or IP address on which the agent is contacted by the dynamic workload broker. The default is the hostname of this computer. If the hostname is a localhost, the hostname parameter must be specified.

-inst_dir *installation_dir*

The directory of the HCL Workload Automation installation. Specify an absolute path. The path cannot contain blanks. If you do not manually specify a path, the path is set to the default home directory, that is, the *home/username* directory, where *username* is the value specified in the -uname option.

-jmport *port_number*

The JobManager port number used by the dynamic workload broker to connect to the dynamic agent. The default value is **31114**. The valid range is from 1 to 65535.

-jmportssl *true|false*

The JobManager port used by the dynamic workload broker to connect to the HCL Workload Automation dynamic agent. The port value is the value of the ssl_port parameter in the `ita.ini` file if **-jmportssl** is set to `true`. If set to `false`, it corresponds to the value of the **tcp_port** parameter in the `ita.ini` file. The `ita.ini` file is located in `ITA\cpa\ita` on Windows™ systems and `ITA/cpa/ita` on UNIX™, Linux™, and IBM i systems.

Set the value to "true" if **- gateway** is set to `local`.

For communication using SSL or HTTPS

Set **jmportssl =** ***true***. To communicate with the dynamic workload broker, it is recommended that you set the value to *true*. In this case, the port specified in **jmport** communicates in HTTPS.

For communication without using SSL or through HTTP

Set **jmportssl =** ***false***. In this case the port specified in **jmport** communicates in HTTP.

-lang *lang_id*

The language in which the twsinst messages are displayed. If not specified, the system LANG is used. If the related catalog is missing, the default C language catalog is used. If neither **-lang** nor LANG are used, the default codepage is set to SBCS. For a list of valid values for these variables, see the following table:

Table 40. Valid values for -lang and LANG parameter

Language	Value
Brazilian portuguese	pt_BR
Chinese (traditional and simplified)	zh_CN, zh_TW
English	en
French	fr
German	de
Italian	it
Japanese	ja
Korean	ko
Russian	ru
Spanish	es

Note: This is the language in which the installation log is recorded and not the language of the installed engine instance. twsinst installs all languages as default.

-new

A fresh installation of the agent. Installs an agent and all supported language packs.

-skip_usercheck

Enable this option if the authentication process within your organization is not standard, thereby disabling the default authentication option. If you specify this parameter, you must create the user manually before running the script.

-skipcheckprereq

If you specify this parameter, HCL Workload Automation does not scan system prerequisites before installing the agent.

For a detailed list of supported operating systems and product prerequisites, see HCL Workload Automation Detailed System Requirements.

-sslkeysfolder

The name and path of the folder containing the certificates in PEM format. The installation program generates the keystore and truststore files using the password you specify with the **--sslpassword** parameter. If you use this parameter, ensure that the **addjruntime** parameter is set to true, because Java™ run time is required for defining custom certificates. This parameter is not supported on HCL Workload Automation Agent (also known as the agent with z-centric capabilities).

-sslpassword

Specify the password for the certificates automatically generated by the installation program. If you use this parameter, ensure that the **addjruntime** parameter is set to true, because Java™ run time is required for defining custom certificates. This parameter is not supported on HCL Workload Automation Agent (also known as the agent with z-centric capabilities).

-tdwbhostname *host_name*

The fully qualified host name of the dynamic workload broker. It is used together with the **-agent** *dynamic* and the **-tdwbport** *tdwbport_number* parameters. This value is registered in the **ResourceAdvisorUrl** property in the `JobManager.ini` file. This parameter is required if you use the **wauser** and **wapassword** or the **apikey** parameters.

If you set the **-gateway** parameter to `remote`, this is the host name of the dynamic agent hosting the gateway and to which the agent you are installing will connect. This information is stored in the `JobManager.ini` file. For information about installing with a local and remote gateway, see Example installation commands.

-tdwbport *tdwbport_number*

The dynamic workload broker HTTP or HTTPS transport port number. It is used together with the **-agent** dynamic and the **-tdwbhostname** *host_name* parameters. The valid range is from 0 to 65535. If you specify **0**, you cannot run workload dynamically. Do not specify **0** if the *-agent* value is **dynamic**. This number is registered in the **ResourceAdvisorUrl** property in the `JobManager.ini` file. This parameter is required if you use the **wauser** and **wapassword** or the **apikey** parameters.

If you set the **-gateway** parameter to `remote`, this is the HTTP or HTTPS port number of the dynamic agent hosting the gateway and to which the agent you are installing will connect. You have specified this port with the

jmport parameter when installing the agent hosting the gateway. For information about installing with a local and remote gateway, see Example installation commands.

***-thiscpu** workstation*

The name of the HCL Workload Automation workstation of this installation. The name cannot exceed 16 characters, cannot start with a number, cannot contain spaces, and cannot be the same as the workstation name of the master domain manager. This name is registered in the `localopts` file. If not specified, the default value is the host name of the workstation.

If the host name starts with a number, **-thiscpu** parameter must be specified.

-u

Displays command usage information and exits.

-uname ***username***

The name of the user for which HCL Workload Automation is installed.

If you are using the **QSECOFR** user or a user with **ALLOBJ authority**, this user name is not the same as the user performing the installation. If you are using a user **different from QSECOFR**, the user performing the installation and the user for which the agent is installed are the same.

If *username* is longer than 8 characters, after installation the agent (and the JobManager component) erroneously run under the **QSECOFR** user, instead of under the authority of the installation user. To prevent this, set the `PASE_USRGRP_LIMITED` environment variable to N.

-wauser ***wauser_name***

One of the following users, defined on the master domain manager:

- The user for which you have installed the master domain manager the agent is connecting to.
- The user with the DISPLAY permission on the FILE named AGENT_CERTIFICATE. This permission allows the user to download certificates or JWT. For more information about this scenario, see Downloading certificates or JWT using a different user.

Always specify the user defined on the master domain manager, also if you are installing a dynamic agent and want it to register to a dynamic domain manager. This is because the dynamic domain manager simply forwards data to and from the master domain manager.

By providing the **wauser** and **wapassword** parameters or the **apikey** parameter, you enable HCL Workload Automation to download and install either the certificates or the JWT already available on the master domain manager:

- To download certificates, set the **jwt** parameter to `false`
- To download JWT, set the **jwt** parameter to `true`. For more information, see -jwt true | false.

Key details about this parameter:

- It is **mutually exclusive** with the -apikey parameter, which provides authentication using an API Key and the -sslkeysfolder path and -sslpassword password parameters.
- It **always requires** the tdwbport and -tdwbhostname host_name parameters.
- It is **not supported** on the HCL Workload Automation Agent (also known as the agent with z-centric capabilities). To generate certificates for the HCL Workload Automation Agent, use the **sslkeysfolder** and **sslpassword** parameters.

For further information about how to automatically download and deploy certificates in PEM format from the master domain manager to dynamic agents and fault-tolerant agents, see Certificates download to dynamic agents and fault-tolerant agents - AgentCertificateDownloader script.

For a comprehensive list of supported combinations for this parameter and related ones, see Table 2.

-wapassword *wauser_password*

One of the following passwords, defined on the master domain manager:

- The password of the user for which you have installed the master domain manager the agent is connecting to.
- The password of the user with the DISPLAY permission on the FILE named AGENT_CERTIFICATE. This permission allows the user to download certificates or JWT. For more information about this scenario, see Downloading certificates or JWT using a different user.

Always specify the user defined on the master domain manager, also if you are installing a dynamic agent and want it to register to a dynamic domain manager. This is because the dynamic domain manager simply forwards data to and from the master domain manager.

By providing the **wauser** and **wapassword** parameters or the **apikey** parameter, you enable HCL Workload Automation to download and install either the certificates or the JWT already available on the master domain manager:

See also -jwt true | false.

Key details about this parameter:

- It is **mutually exclusive** with the -apikey parameter, which provides authentication using an API Key and the -sslkeysfolder path and -sslpassword password parameters.
- It **always requires** the tdwbport and -tdwbhostname host_name parameters.
- It is **not supported** on the HCL Workload Automation Agent (also known as the agent with z-centric capabilities). To generate certificates for the HCL Workload Automation Agent, use the **sslkeysfolder** and **sslpassword** parameters.

You can optionally encrypt the password using the secure script. For more information, see Optional password encryption - secure script on page 284.

For further information about how to automatically download and deploy certificates in PEM format from the master domain manager to dynamic agents and fault-tolerant agents, see Certificates download to dynamic agents and fault-tolerant agents - AgentCertificateDownloader script.

This parameter always requires the tdwbport and -tdwbhostname host_name parameters.

For a comprehensive list of supported combinations for this parameter and related ones, see Table 2.

-work_dir *working_dir*

The temporary directory used for the HCL Workload Automation installation process files deployment. The path cannot contain blanks. If you do not manually specify a path, the path is set to `/tmp/TWA/tws`*`version_number>`*.

-v

Displays the command version and exits.

Example installation of an agent on IBM i systems

About this task

The following example shows the syntax used when using the **twsinst** script to install a new instance of the agent on an IBM i system.

```
./twsinst -new
-uname TWS_user
-acceptlicense yes
-hostname thishostname.mycompany.com
-jmport 31114
-tdwbport 41114
-tdwbhostname mainbroker.mycompany.com
-work_dir "/tmp/TWA/tws93"
```

Chapter 17. Upgrading agents on IBM i systems

How to upgrade agents on IBM i systems.

About this task

You can upgrade the agent on an IBM i system by using the `twsinst` installation script.

To upgrade an HCL Workload Automation agent, perform the following steps:

1. Sign on as the user who performed the installation, either **QSECOFR** or an existing user with ALLOBJ authority. If you installed with a user different from **QSECOFR**, use the same user who performed the installation and specify the **allObjAuth** parameter to indicate that the user has the ALLOBJ authority. For more information about this parameter, see Agent installation parameters on IBM i systems on page 320. You can find the name of the profile used to perform the installation in the `instUser` located in the *`agent_data_dir`*`/installation/instInfo`.
2. Download the installation images from Flexnet or from HCL Software.
3. If you downloaded the eImages, to extract the package, use the *PASE* shell or the *AIXterm* command.

   Using *PASE* shell:

   a. Open the *PASE* shell.
   b. Run the command "CALL QP2TERM".
   c. Locate the folder where you downloaded the eImages and run the command:

      ```
      "tar xvf TWS1025_IBM_I.tar"
      ```

   d. Exit from the *PASE* shell.

   Using *AIXterm* command:

   a. Start the *Xserver* on your desktop.
   b. On the iSeries machine, open a *QSH shell* and export the display.
   c. In *QSH shell*, go to the directory */QopenSys* and run the command "aixterm -sb".
   d. A pop-up window is displayed on your desktop. By Using this pop-up window, extract the file *TWS1025_IBM_I.tar*.

4. Open a *QSH shell* and run the **twsinst** script.

   The installation procedure replaces the library to the user profile library list of the dynamic agent user profile and sets this job description as the job description of the dynamic agent user profile. The upgrade process replaces the new version of the agent in the directory where the old agent is installed.

If the operation fails to understand the cause of the error, see Analyzing return codes for agent installation, upgrade, restore, and uninstallation on page 313.

Command usage and version

Show command usage and version

```
twsinst -u | -v
```

Upgrade an instance

```
./twsinst -update -uname user_name
 -acceptlicense yes|no
 [-addjruntime true]
 [-allObjAuth]
 [-create_link]
 [-hostname host_name]
 [-inst_dir install_dir]
 [-jmport port_number]
 [-jmportssl boolean]
 [-lang lang-id]
 [-reset_perm]
 [-recovInstReg true]
 [-skip_usercheck]
 [-tdwbhostname host_name]
 [-tdwbport port_number]
 [-wait minutes]
 [-work_dir working_dir]
```

For a description of the installation parameters and options that are related to agent on this operating system, see Agent upgrade parameters on IBM i systems on page 329.

Agent upgrade parameters on IBM i systems

About this task

The parameters set when using the **twsinst** script to upgrade a dynamic agent on IBM i systems.

-acceptlicense *yes|no*

Specifies whether to accept the License Agreement.

-addjruntime *true*

Adds the Java™ run time to run job types with advanced options to the agent. The run time environment is used to run application job plug-ins on the agent and to enable the capability to run remotely, from the agent, the dynamic workload broker resource command on the server.

By default, if the Java run time was already installed on the agent, it will be upgraded to the new version.

If the Java run time was not installed on the agent, it will not be installed during the upgrade, unless you specify `-addjruntime true`.

If you decided not to install Java™ run time when you upgrade, you can still add this feature later. For details about how to add a feature, see *HCL Workload Automation for Z: Planning and installation*.

-allObjAuth

If you are installing, upgrading, or uninstalling with a user different from the default **QSECOFR** user, this parameter specifies that the user has the required ALLOBJ authority. Ensure the user is existing and has ALLOBJ authority because the product does not verify that the correct authority is assigned. The same user must be specified when installing, upgrading or uninstalling the agent. If you are using the **QSECOFR** user, this parameter does not apply.

-create_link

Create the **symlink** between `/usr/bin/at` and `<`*`install_dir`*`>/TWS/bin/at`. See Table 1 for more information.

-displayname

The name to assign to the agent. The default is the host name of this computer.

-inst_dir *installation_dir*

The directory of the HCL Workload Automation installation.

Note: The path cannot contain blanks. If you do not manually specify a path, the path is set to the default home directory, that is, the *user_ home\user_name* directory.

-jmport *port_number*

The JobManager port number used by the dynamic workload broker to connect to the HCL Workload Automation dynamic agent. The default value is **31114**. The valid range is from 1 to 65535.

-jmportssl *true|false*

The JobManager port used by the dynamic workload broker to connect to the HCL Workload Automation dynamic agent. This number is registered in the `ita.ini` file located in the `ITA/cpa/ita` directory.

For communication using SSL or HTTPS

Set **jmportssl = true**. To communicate with the dynamic workload broker, it is recommended that you set the value to **true**. If the value is set to *true*, the port specified in **jmport** communicates in HTTPS.

For communication without using SSL, or through HTTP

Set **jmportssl = false**. If the value is set to *false*, the port specified in **jmport** communicates in HTTP.

-lang *lang_id*

The language in which the `twsinst` messages are displayed. If not specified, the system LANG is used. If the related catalog is missing, the default C language catalog is used.

Note: This is the language in which the installation log is recorded, and not the language of the installed engine instance. The twsinst script installs all languages by default.

-recovInstReg *true*

To re-create the registry files. Specify it if you have tried to upgrade a stand-alone agent and you received an error message that states that an instance of HCL Workload Automation cannot be found, this can be caused by a corrupt registry file. See Upgrading when there are corrupt registry files.

-skip_usercheck

Enable this option if the authentication process within your organization is not standard, thereby disabling the default authentication option. If you specify this parameter, you must create the user manually before running the script.

-skipcheckprereq

If you specify this parameter, HCL Workload Automation does not scan system prerequisites before upgrading the agent.

For a detailed list of supported operating systems and product prerequisites, see HCL Workload Automation Detailed System Requirements.

-tdwbhostname *host_name*

The dynamic workload broker fully qualified host name. It is used together with the **-tdwbport** *tdwbport_number* parameter. It adds and starts the capabilities to run workload dynamically to HCL Workload Automation. This value is registered in the **ResourceAdvisorUrl** property in the `JobManager.ini` file.

-tdwbport *tdwbport_number*

The dynamic workload broker HTTP or HTTPS port number used to add dynamic scheduling capabilities to your distributed or end-to-end environment. It is used together with the **-tdwbhostname** *host_name* parameter. This number is registered in the ResourceAdvisorUrl property in the `JobManager.ini` file. Specify a nonzero value to add dynamic capability. The valid range is 0 to 65535.

-uname *user_name*

The name of the user for which HCL Workload Automation is being updated. The software is updated in this user's home directory. This user name is not to be confused with the user performing the upgrade.

Note: This user name is not the same as the user performing the installation logged on as **QSECOFR**.

-update

Upgrades an existing agent that was installed using **twsinst**.

-wait *minutes*

The number of minutes that the product waits for jobs that are running to complete before starting the upgrade. If the jobs do not complete during this interval the upgrade does not proceed and an error message is displayed. Valid values are integers or **-1** for the product to wait indefinitely. The default is **60** minutes.

-work_dir *working_dir*

The temporary directory used for the HCL Workload Automation installation process files deployment. The path cannot contain blanks. If you do not manually specify a path, the path is set to `/tmp/TWA/tws1025`.

Example upgrade of an agent on IBM i systems

About this task

The following example shows the syntax used when using the **twsinst** script to upgrade an instance of the agent on IBM i system.

```
./twsinst -update
-uname TWS_user
-allObjAuth
-acceptlicense yes
-nobackup
-work_dir "/tmp/TWA/tws1025"
```

Chapter 18. Uninstalling agents on IBM i systems

How to uninstall dynamic and z-centric agents on IBM i systems.

To uninstall the agents on an IBM i system by using the twsinst script, perform the following steps:

1. Ensure that all HCL Workload Automation processes and services are stopped, and that there are no active or pending jobs. For information about stopping the processes and services, see the section about starting and stopping Application server in *Administration Guide*.
2. Sign on as the user who performed the installation, either **QSECOFR** or an existing user with ALLOBJ authority. If you installed with a user different from **QSECOFR**, use the same user who performed the installation and specify the **allObjAuth** parameter to indicate that the user has the ALLOBJ authority. For more information about this parameter, see Agent installation parameters on IBM i systems on page 320. You can find the name of the profile used to perform the installation in the `instUser` located in the *`agent_data_dir`*`/installation/instInfo`.
3. **Note:** Only for dynamic agents, you have the option of installing using a user different from **QSECOFR** and with no specific authorizations. In this case, specify the same user who performed the installation.
4. Change your directory to /*installation_dir*/TWS. For example: `/home/user1/TWS` where user1 is the name of HCL Workload Automation user.
5. From the `Installation directory\TWS` directory, run the twsinst script as follows:

```
twsinst -uninst -allObjAuth -uname username
[-wait minutes][-lang lang_id] [-work_dir working_dir]
```

-uninst

Uninstalls HCL Workload Automation.

uname *username*

The name of the user for which HCL Workload Automation is uninstalled. This user name is not the same as the user performing the installation.

-allObjAuth

If you are installing, upgrading, or uninstalling with a user different from the default **QSECOFR** user, this parameter specifies that the user has the required ALLOBJ authority. Ensure the user is existing and has ALLOBJ authority because the product does not verify that the correct authority is assigned. The same user must be specified when installing, upgrading or uninstalling the agent. If you are using the **QSECOFR** user, this parameter does not apply.

-uname *username*

The name of the user for which HCL Workload Automation is uninstalled.

If you are using the **QSECOFR** user or a user with **ALLOBJ authority**, this user name is not the same as the user performing the installation. If you are using a user **different from QSECOFR**, the user performing the installation and the user for which the agent is installed are the same.

-wait *minutes*

The number of minutes that the product waits for jobs that are running to complete before starting the uninstallation. If the jobs do not complete during this intervals the uninstallation stops and an error message is displayed. Valid values are integers or **-1** for the product to wait indefinitely. The default is **60** minutes.

-lang *lang_id*

The language in which the `twsinst` messages are displayed. If not specified, the system LANG is used. If the related catalog is missing, the default C language catalog is used.

-work_dir *working_dir*

The temporary directory used for the HCL Workload Automation installation process files deployment. If you do not manually specify a path, the path is set to `/tmp/TWA/tws`*`version_number`*`>`.

The following example shows a twsinst script that uninstalls the HCL Workload Automation agent, originally installed for **twsuser** user:

On IBM i systems:

```
./twsinst -uninst -uname TWS_user -allObjAuth
```

Chapter 19. The twsinst script log files on IBM i systems

About this task

The twsinst log file name is:

Where: `<TWS_INST_DIR>/twsinst_IBM_i_TWS_user^product_version.log`

TWS_INST_DIR

The HCL Workload Automation installation directory. The default installation directory is `/home/TWS_user`.

TWS_user

The name of the user for which HCL Workload Automation was installed, that you supplied during the installation process.

product_version

Represents the product version. For example, for version 10.2.5 of the product, the value is 10.2.5.00

Chapter 20. Analyzing return codes for agent installation, upgrade, restore, and uninstallation

Check how your operation completed by analyzing the return codes that are issued by twsinst.

Return codes that you can receive when you are installing, upgrading, restoring, or uninstalling agents. To analyze them and take corrective actions, run the following steps:

On Windows operating systems

1. Display the operation completion return code, by using the following command:

```
echo %ERRORLEVEL%
```

2. Analyze the following table to verify how the operation completed:

Table 41. Windows operating system agent return codes

Error Code	Description	User action
0	Success: The operation completed successfully without any warnings or errors.	None.
1	Generic failure	Check the messages that are displayed on the screen by the script. Correct the error and rerun the operation. If the error persists, contact Support.
2	The installation cannot create the HCL Workload Automation user or assign the correct permission to it.	Verify the operating system policies and configuration. Verify the input values. If necessary, create the user manually before you run the installation.
3	The password is not correct or the installation cannot verify it.	Verify the operating system policies and configuration. Verify the input values.
4	The HCL Workload Automation installation directory is not empty. You specified as installation folder a directory that exists.	Empty it or specify a different directory.
5	An error occurred checking the HCL Workload Automation prerequisites on the workstation.	See the System Requirements Document at HCL Workload Automation Detailed System Requirements.
6	The HCL Workload Automation registry is corrupted.	Use the recovInstReg option to recover the registry. Then, rerun the operation.
7	The upgrade or restore operation cannot retrieve the information from the configuration files.	Check that the previous installation and the `localopts`, the `globalopts`, the

Error Code	Description	User action
		`ita.ini`, and the `JobManager.ini` files are not corrupted. Correct the errors and try again the operation.
8	The upgrade, restore, or uninstallation cannot proceed because there are jobs that are running.	Stop the jobs that are running or wait for these jobs to complete. Restart the operation.
9	The upgrade, restore, or uninstallation cannot proceed because there are files that are locked.	Stop all the processes that are running and close all the activities that can block the installation path. Restart the operation.
10	The upgrade, restore, or uninstallation cannot proceed because there are command lines opened.	Close the command lines. Restart the operation.

On UNIX and Linux operating systems:

1. Display the installation completion return code, by using the following command:

```
echo $?
```

2. Analyze the following table to verify how the installation completed:

Table 42. UNIX or Linux operating system agent return codes

Error Code	Description	User action
0	Success: The installation completed successfully without any warnings or errors.	None.
1	Generic failure.	Check the messages that are displayed on the video by the script. Correct the error and rerun the operation. If the error persists, contact Support.
2	The installation did not find the HCL Workload Automation user or its home directory. The HCL Workload Automation user that you specified either does not exist or does not have an associated home directory.	Verify the operating system definition of the HCL Workload Automation user.
3	Not applicable	

Error Code	Description	User action
4	The HCL Workload Automation installation directory is not empty. You specified as installation folder a directory that exists.	Empty it or specify a different directory.
5	An error occurred checking the HCL Workload Automation prerequisites on the workstation.	See the System Requirements Document at HCL Workload Automation Detailed System Requirements.
6	The HCL Workload Automation registry is corrupted.	Use the recovInstReg option to recover the registry. Then, rerun the operation.
7	The upgrade or restore operation cannot retrieve the information from the configuration files.	Check that the previous installation and the `localopts`, the `globalopts`, the `ita.ini`, and the `JobManager.ini` files are not corrupted. Correct the errors and try again the operation.
8	The upgrade, restore, or uninstallation cannot proceed because there are jobs that are running.	Stop the jobs that are running or wait for these jobs to complete. Restart the operation.
9	The upgrade, restore, or uninstallation cannot proceed because there are files that are locked.	Stop all the processes that are running and close all the activities that can block the installation path. Restart the operation.
10	The upgrade, restore, or uninstallation cannot proceed because there are command lines opened.	Close the command lines. Restart the operation.

Appendix A. Integrating Workload Automation with Zowe™

Zowe™ is an open source project that enables you to interact with z/OS through modern interfaces. You can issue Workload Automation commands through the Zowe command-line interface (CLI), and access Workload Automation REST APIs through the Zowe API Mediation Layer (ML).

The following sections provide detailed information about:

- Using Zowe CLI to issue Workload Automation commands on page 339
- Using Zowe API Mediation Layer to access the Workload Automation REST APIs on page 340
- Supporting authentication through Zowe JWT token on page 342

Using Zowe CLI to issue Workload Automation commands

The Workload Automation plug-in for Zowe CLI lets you issue Workload Automation commands to remotely control your workload. With Workload Automation commands, you can monitor and modify jobs, jostreams and resources, as well as issue WAPL commands.

How the Workload Automation (WA) plug-in works

The WA plug-in:

- Defines a Workload Automation profile to manage the connection information, which is required to access the WA APIs.
- Provides you with a CLI interface with the relevant APIs on the Z connector server.

For information about Workload Automation commands and syntax, see the online help that is provided within the WA plug-in.

Software requirements

Before installing and using the WA plug-in, you must:

- Install Zowe CLI on your workstation. For details about this task, see Installing Zowe CLI.
- Ensure that the Z connector V9.5 Fix Pack 2 is installed and running in your environment.
- If you are using the Z controller V9.5, ensure that you have installed the PTFs UI68881, UI68882, UI69085, and UI69193.
- Access the Workload Automation APIs through the API Mediation Layer (API ML), or connect the WA plug-in directly to the WA API. For more information, see Using Zowe API Mediation Layer to access the Workload Automation REST APIs on page 340

Installing the WA plug-in

To install the WA plug-in, download the `zowe-cli-hcl-wa-plugin.zip` file that is provided on Flexera and install the WA plug-in by performing the following steps:

1. Unzip the zip file locally.
2. From the directory where you unzipped the file, install the WA plug-in by issuing the following command:

   ```
   zowe plugins install .
   ```

Uninstalling the WA plug-in

To uninstall the WA plug-in, issue the following command:

```
zowe plugins uninstall @zowe/zowe-cli-hcl-wa-plugin
```

After the uninstallation process completes successfully, the product no longer contains the WA plug-in.

For more details

For more detailed information about the WA plug-in, see the readme file that you find in the Info tab of Automation Hub. A video about using the Zowe CLI is available at this link.

Using Zowe API Mediation Layer to access the Workload Automation REST APIs

How to add the Workload Automation REST APIs to the Zowe API Mediation Layer, which is a component that provides a gateway acting as a reverse proxy for z/OS services, and a catalog of REST APIs.

Before you begin

Before adding the Workload Automation REST APIs to the Zowe API Mediation Layer, ensure that:

1. Zowe is installed and configured on your Z/OS system. For details about how to install and configure Zowe, see the Zowe documentation. .
2. You have installed Dynamic Workload Console V9.5 Fix Pack 2.
3. You have downloaded the `zowe-cli-hcl-wa-plugin.zip` file that is provided on Flexera and installed the plug-in as described in the `README_wa.html` file that is provided in the zip file.

About this task

To add the Workload Automation REST API to the Zowe API mediation layer, perform the following steps:

1. Edit the `wa.yml` file that is provided in `zowe-cli-hcl-wa-plugin.zip` as follows:

   ```
   services:
         - serviceId: <serviceID>   # unique lowercase ID of the service
           catalogUiTileId: static
           title: HCL Workload Automation for Z
           description: HCL Workload Automation for Z Service
           instanceBaseUrls:  # list of base URLs for each instance
           - https://<zconn_hostname>:<zconn_port>  # scheme:https//zconn_hostname:zconn_port
         homePageRelativeUrl: /
         statusPageRelativeUrl: /twsz
         healthCheckRelativeUrl: /twsz/v1
         routes:
           - gatewayUrl: api/v1
             serviceRelativeUrl: /twsz/v1 # relativePath that is added to baseUrl of an instance
   ```

```
    # List of APIs provided by the service:
    apiInfo:
      - gatewayUrl: api/v1
        documentationUrl: https://<zconn_hostname:<zconn_port>/twsz/IWS_API3_all_zos.json
    # List of tiles that can be used by services defined in the YAML file:
    catalogUiTiles:
       static:
          title: HCL Workload Automation for Z API Services
          description: HCL Workload Automation for Z REST Services
```

where:

zconn_hostname

Host name of the Z connector where the Workload Automation REST APIs are located.

zconn_port

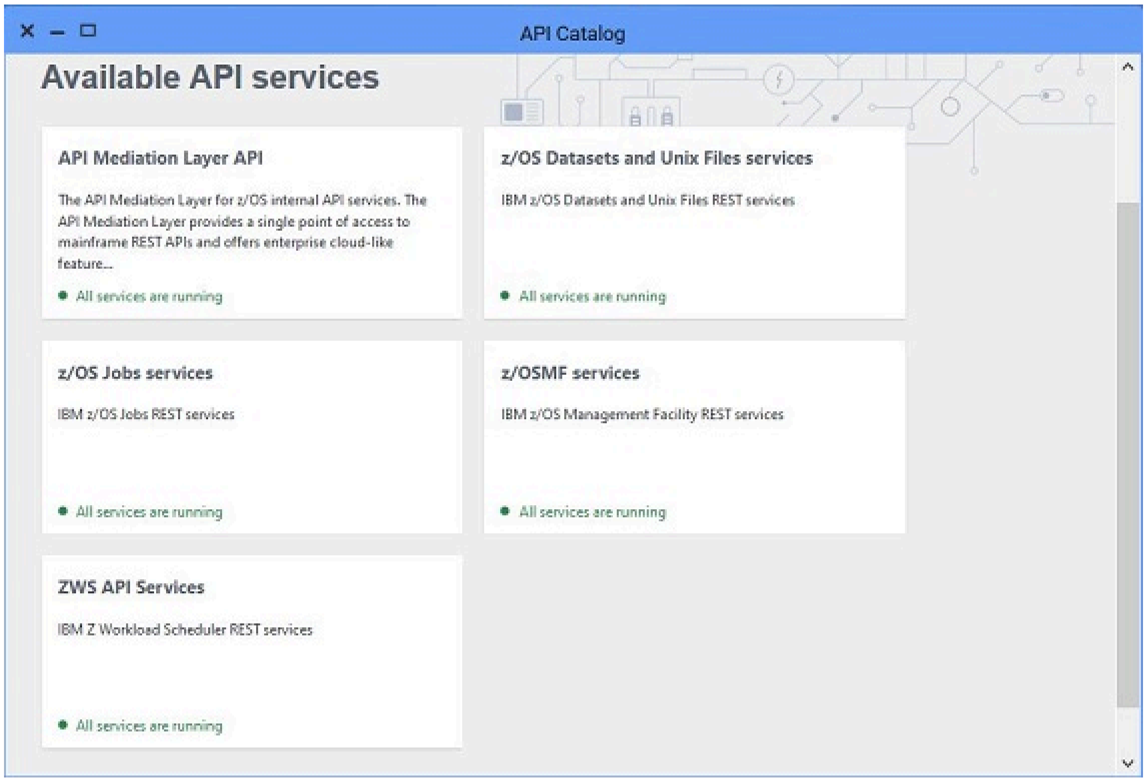
Port used by the Z connector where the Workload Automation REST APIs are located.

The result of the YAML file is that each call to the Zowe API mediation layer at https://zowehost:zoweport/api/v1/waservice is redirected to the Z connector REST API at `https://`*`zconn_hostname`*`:`*`zconn_port`*`/twsz/v1`.

2. Ensure that the certificate used by the Workload Automation REST API is trusted by the API mediation layer.

   To meet this requirement, follow the instructions provided in the Zowe documentation.

3. Verify that the Workload Automation REST API was successfully added to the API mediation layer by opening the Zowe Virtual Desktop and accessing the API catalog. The catalog will show the Workload Automation REST API, as in the following example:



Available API services

API Mediation Layer API

The API Mediation Layer for z/OS internal API services. The API Mediation Layer provides a single point of access to mainframe REST APIs and offers enterprise cloud-like feature...

● All services are running

z/OS Datasets and Unix Files services

IBM z/OS Datasets and Unix Files REST services

● All services are running

z/OS Jobs services

IBM z/OS Jobs REST services

● All services are running

z/OSMF services

IBM z/OS Management Facility REST services

● All services are running

ZWS API Services

IBM Z Workload Scheduler REST services

● All services are running

Results

You have successfully added the Workload Automation REST API to the Zowe API Mediation Layer. You can issue the WA commands from the Zowe command-line.

Supporting authentication through Zowe JWT token

The JWT secret that signs the JWT token is a private key that is generated during Zowe keystore configuration. To support authentication through JWT, perform the following steps.

About this task

1. Copy the JWT secret from the API ML installation. For more information, see the Zowe documentation about authenticating with JSON Web tokens.
2. From the workstation where you installed the Dynamic Workload Console, import the secret into the trust store by issuing the following command from `<DWC_DIR>/usr/servers/dwcServer/resources/security`:

   ```
   keytool -import -alias <my_secret> -keystore TWSServerTrustFile.jks
   -file <complete_path>/localhost.keystore.jwtsecret.pem
   ```

3. Edit the `<DWC_DIR>/usr/servers/dwcServer/server.xml` file as follows:
   a. Add the row `<feature>mpJwt-1.1</feature>` as shown in the following example:

      ```
      <featureManager>
           <feature>javaee-7.0</feature>
        <feature>passwordUtilities-1.0</feature>
        <feature>localConnector-1.0</feature>
        <feature>mpJwt-1.1</feature>
          </featureManager>
      ```

   b. Add the following information:

      ```
      <!-- MPJWT configuration -->
       <mpJwt id="<my_mpJwt>" keyName="<my_secret>"
                 userNameAttribute="sub" ignoreApplicationAuthMethod="false"/>
      ```

   where:

   `<my_mpJwt>`

      A unique identifier that you define for the MicroProfile JWT (mpJwt).

   `<my_secret>`

      The secret that you imported into the trust store at step 2.

4. Save the server.xml file.

Appendix B. Sample library (SEQQSAMP)

The SEQQSAMP library contains samples to help you install and customize HCL Workload Automation for Z.

In most cases, you need only add installation-specific JCL to adapt a member in SEQQSAMP to your requirements. Sample library (SEQQSAMP) on page 343 lists all members in the SEQQSAMP library and provides a brief description of each member. The following pages describe the samples relating to installing HCL Workload Automation for Z in more detail. Descriptions of other sample-library members are included in the book that describes the function demonstrated by the sample. For example, program-interface samples are described in *Developer's Guide: Driving HCL Workload Automation for Z*.

Some of the samples provided address a specific function and you might be able to use the sample unchanged in your environment. If you need to change a sample member, it is advisable to copy the source to a separate library. The original sample member is then available for reference. It is also recommended that you create an SMP/E *usermod* for each sample member you run in the production environment. Changes to the sample source code will then be flagged for your attention, and subsequent updates can be reflected in the production code as soon as possible.

Table 43. SEQQSAMP library members

Member	Brief description
EQQ9RF01	Sample RACF® router table entry to enable security environment.
EQQ9RFDE	Sample RACF® class descriptor entry to enable security environment.
EQQ9SM01	JCL to install RACF® router table update.
EQQ9SMDE	JCL to install RACF® class descriptor update.
EQQACPT*x*	Sample SMP/E ACCEPT JCL for the Z controller software, where the value of *x* depends on the language.
EQQACTR1	Sample SMF exit IEFACTRT, written in assembler, to enable job-tracking.
EQQAIXST	Parameters used by the EQQX9AIX and EQQAIXTR samples.
EQQAIXTR	Sample tracker running on AIX®, used with EQQX9AIX.
EQQALLOC	JCL to allocate the HCL Workload Automation for Z distribution and target libraries.
EQQALSMP	Sample JCL to allocate and initialize the SMP/E environment needed to install HCL Workload Automation for Z
EQQAPISM	ASCII file containing a sample API application.
EQQAPPL*x*	Sample SMP/E APPLY JCL for the Z controller software, where the value of *x* depends on the language.
EQQAUDIB	Sample to invoke EQQAUDIT in batch mode outside of the dialog, processing EQQTROUT or EQQDROUT data set.

Table 43. SEQQSAMP library members (continued)

Member	Brief description
	Note: EQQAUDIB can be used successfully only if the **EQQTROUT dsname** and the **EQQAUDIT output dsn** fields in the EQQJOBSA panel are typed out.
EQQCLEAN	Sample procedure invoking EQQCLEAN program.
EQQCONOP	Sample parameters used by EQQCONO.
EQQCONO	Sample started task procedure for controller only.
EQQCONP	Sample initial parameters for a controller and tracker in the same address space.
EQQCON	Sample started task procedure for a controller and tracker in the same address space.
EQQCVM2	Sample to enable submission and tracking on VM systems using EQQUX009.
EQQCVM	Sample to enable job-tracking facilities on VM systems.
EQQDBENC	Contains the JCL to encrypt the password in the DBOPT statement.
EQQDBOPT	Sample DBOPT statement.
EQQDBREP	Contains the SQL statements to create the DB2 reporting objects.
EQQDDDEF	Sample job to allocate DDDEFs in SMP/E.
EQQDELDI	JCL and usage notes for the data set deletion function.
EQQDLFX	Assembler installation sample of DLF connect/disconnect exit.
EQQDPCOP	JCL and usage notes for copy VSAM function.
EQQDPX01	DP batch sample user exit to update the scheduling environment.
EQQDSCL	Batch Clean Up sample.
EQQDSCLP	Batch Clean up sample parameters.
EQQDSECT	Assembler version of PIF data areas.
EQQDSEX	Batch Export sample.
EQQDSEXP	Batch Export sample parameters.
EQQDSIM	Batch Import sample.
EQQDSIMP	Batch Import sample parameters.
EQQDSRG	Batch sample reorg.
EQQDSRI	Batch Recovery index.
EQQDSRIP	Batch Recovery index parameters.

Table 43. SEQQSAMP library members (continued)

Member	Brief description
EQQDST	Sample procedure to start Data Store.
EQQDSTP	Parameters for sample procedure to start Data Store.
EQQFLEX	Sample job used by the Z controller at daily planning, to communicate the number of jobs that were submitted on z-centric and dynamic agents to the license server.
EQQICNVS	Sample job to migrate VSAM files.
EQQINIRE	Sample JCL to create the DB2 reporting objects.
EQQISMKD	Sample job to run EQQMKDIR exec for directories.
EQQJCCTB	JCL to assemble a JCC message table macro definition.
EQQJCLIN	Sample JCL to start program EQQPDLF.
EQQJER2U	Sample to restore the EXIT7 as a JES2 usermod.
EQQJER2V	Sample to restore the EXIT5 as a JES2 usermod.
EQQJER3U	Sample to restore the EQQUX191 and EQQUX291 as JES3 usermods.
EQQJES21	JCL to assemble and link-edit the JES2 EXIT51.
EQQJES2	JCL to assemble and link-edit a JES2 exit.
EQQJES2U	JCL to install the JES2 EXIT7 as an SMP/E usermod.
EQQJES2V	JCL to install the JES2 EXIT51 as an SMP/E usermod.
EQQJES3	JCL to assemble and link-edit a JES3 exit.
EQQJES3U	JCL to install a JES3 exit as an SMP/E usermod.
EQQJVXIT	Sample assembler JCL-variable-substitution exit. Also used for variable substitution in System Automation commands.
EQQLSJCL	Sample JCL to invoke the EQQLSENT macro.
EQQMIGRE	Sample JCL to migrate DB2 reporting objects from the previous version.
EQQMKDIR	Sample exec to create directories.
EQQNCFCT	Sample parameters for an SNA connection between Z controller and tracker.
EQQNETW1	REXX™ EXEC that receives HCL Workload Automation for Z WTO messages and issues z/OS 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.

Table 43. SEQQSAMP library members (continued)

Member	Brief description
EQQOCWTO	Sample job to assemble and linkedit the IPOWTO routine used by the PIF REX sample.
EQQPCS01	Allocates data sets that need to be unique within the SYSPLEX.
EQQPCS02	Allocates data sets that need to be unique to each ZOS image in the SYSPLEX.
EQQPCS03	Generates a job that allocates VSAM copy data sets.
EQQPCS04	Defines Data Store VSAM files and initializes them.
EQQPCS07	Allocates VSAM data sets for Restart and Cleanup.
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 Z controller.
EQQPCS11	Allocates data sets (EQQOUCEV on page 136 and EQQOUCKP on page 136) 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.
EQQPIFAD	Program-interface PL/I sample that creates a two-operation application in the AD database.
EQQPIFAP	Program-interface PL/I sample that resolves JCL variables.
EQQPIFCB	Program-interface assembler samples for various current plan or long-term plan actions.
EQQPIFCL	Program-interface assembler sample that uses the DAYSTAT command to return work or free status for a particular date.
EQQPIFDJ	Program-interface assembler sample, deletes JCL for completed occurrences from JS data set.
EQQPIFJC	Program-interface COBOL sample to manipulate JCL variable tables.
EQQPIFJD	Program-interface PL/I sample that can either list or delete records in the JCL repository data set (JS).
EQQPIFJV	Program-interface PL/I sample to manipulate JCL variable tables.
EQQPIFJX	Sample to maintain the JCL repository.
EQQPIFOP	Program-interface REXX™ sample to modify an operation in the current plan.
EQQPIFPR	Program-interface REXX™ sample to list all cyclic periods.
EQQPIFWI	Program-interface PL/I sample to modify capacity values in an open interval of a current plan workstation.
EQQPROC	Sample procedure, started by HCL Workload Automation for Z, to initiate purge of DLF objects.

Table 43. SEQQSAMP library members (continued)

Member	Brief description
EQQRECVE	Sample SMP/E RECEIVE JCL for the Z controller software
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.
EQQRETWT	Sample program to simulate abends, return codes and waits.
EQQRMDS	Usage notes for the job-log-retrieval exit object code to interface to RMDS.
EQQRXSTG	An assembler routine to get and free storage for the REXX™ program-interface samples.
EQQSAMPI	JCL to load sample data for application descriptions, operator instructions, and workstation descriptions to the databases.
EQQSERP	Sample initial parameters for a Server.
EQQSER	Sample started task procedure for a Server.
EQQSMF	JCL to assemble and install the SMF exits.
EQQSMLOG	Sample procedure that creates the GDG data set where the outgoing MLOG file is archived when the MLOG switching function takes effect. Uses the EQQREPRO input parameter.
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.
EQQU831	Sample SMF exit IEFU83 to enable job tracking and optionally include data set triggering support.
EQQUJI1	Sample SMF exit IEFUJI to enable job-tracking.
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 backup an HCL Workload Automation for Z 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.
EQQUX004	Sample event-filtering exit.
EQQUX011	Sample job-tracking log write exit.

Table 43. SEQQSAMP library members (continued)

Member	Brief description
EQQUX013	Sample job-tailoring prevention exit.
EQQUX0N	Sample PL/I start/stop exit, EQQUX000.
EQQUX191	Sample JES3 exit IATUX19 to enable job tracking.
EQQUX291	Sample JES3 exit IATUX29 to enable job tracking.
EQQUX9N	Sample PL/I operation-initiation exit, communicating with VM (EQQUX009).
EQQUXCAT	Sample restart and clean up exit for the EQQCLEAN program.
EQQUXPIF	Sample user exit to validate application descriptions.
EQQUXSAZ	Sample assembler system command exit, communicating with System Automation invoked in place of EQQUX007 for automation workstations.
EQQVTAMN	Sample VTAM® definition for SNA connection between tracker and controller.
EQQVTAMS	Sample VTAM® definition for server SNA connection.
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®.
EQQXCFCT	Sample definitions for XCF connection between tracker and controller.
EQQXIT51	Sample JES2 EXIT51 to enable job tracking for JES2.
EQQXIT74	Sample JES2 EXIT7 to enable job tracking for JES2.
EQQXML01	Sample XML file for data set triggering event rule definitions.

Using the Visual Age compiler

With the z/OS® operating system, the Visual Age PL/I compiler replaces all the previous PL/I compilers. Therefore, if you use this compiler, you need to customize the samples in PL/I as follows:

1. Replace the PL/I compiler invocation statement:

   ```
   EXEC PGM=IEL0AA
   ```

   with:

```
EXEC PGM=IBMZPLI
```

2. Link into a PDS/E data set for SYSLMOD or include a pre-link edit step in the JCL.

As an example, here is the JCL for the EQQPIFJV sample using the Visual Age PL/I compiler:

```
//EQQPIFJV JOB MSGCLASS=N, ...........
//PLI1     EXEC PGM=IBMZPLI,REGION=1024K,
//         PARM='OBJECT,OPTIONS'
//STEPLIB DD DSN=IBMZ.V2R2M1.SIBMZCMP,DISP=SHR
//SYSPRINT DD  SYSOUT=*
//SYSLIN   DD  UNIT=SYSDA,SPACE=(CYL,(2,1)),DISP=(,PASS),
//    DSN=&&OBJ1
//SYSUT1   DD  UNIT=SYSDA,SPACE=(CYL,(3,3))
//SYSIN    DD  *
/*
//*
//PLI2     EXEC PGM=IBMZPLI,REGION=1024K,
//         COND=(4,LT,PLI1),PARM='OBJECT,OPTIONS'

//SYSPRINT DD  SYSOUT=*
//STEPLIB DD DSN=IBMZ.V2R2M1.SIBMZCMP,DISP=SHR
//SYSLIN   DD  UNIT=SYSDA,SPACE=(CYL,(2,1)),
//    DISP=(,PASS),DSN=&&OBJ2
//SYSUT1   DD  UNIT=SYSDA,SPACE=(CYL,(3,3))
//SYSIN    DD  *
 ........................
/*
//*
//******************************************************
//* PRE-LINK-EDIT STEP                                 *
//******************************************************
//PLKED    EXEC PGM=EDCPRLK,COND=(8,LT,PLI1),
//     REGION=2048K
//SYSDEFSD DD DSN=&&DEF1,LRECL=80,BLKSIZE=3200,
//     DISP=(,PASS)
//STEPLIB  DD  DSN=CEE.SCEERUN,DISP=SHR
//SYSMSGS  DD  DSN=CEE.SCEEMSGP(EDCPMSGE),DISP=SHR
//SYSLIB   DD  DUMMY
//SYSMOD   DD  DSN=&&PLNK,DISP=(,PASS),
//             UNIT=SYSALLDA,SPACE=(CYL,(1,1)),
//             DCB=(RECFM=FB,LRECL=80,BLKSIZE=3200)
//SYSIN DD DSN=&&OBJ1,DISP=(OLD,DELETE)
//      DD DSN=&&OBJ2,DISP=(OLD,DELETE)
//SYSPRINT DD  SYSOUT=*
//SYSOUT   DD  SYSOUT=*
//******************************************************
//* SCEELKED ADDED TO SYSLIB ON LINK STEP              *
//******************************************************
//LKED     EXEC PGM=IEWL,PARM='XREF',
//    COND=(4,LT,PLI2),REGION=4M
//SYSPRINT DD  SYSOUT=*
//SYSLIB   DD  DISP=SHR,DSN=CEE.SCEELKED
//         DD  DISP=SHR,DSN=USER.OPC23.LINKLI
//SYSLMOD  DD  DISP=SHR,DSN=SVIOLA.SEQQLMD0
//OPCLIB   DD  DISP=SHR,DSN=USER.OPC23.LINKLI
//SYSUT1   DD  UNIT=SYSDA,SPACE=(CYL,(3,3))
//SEQOBJ1  DD  DISP=(OLD,DELETE),DSN=&&PLNK
```

```
//SYSLIN   DD  *
  INCLUDE SEQOBJ1
  INCLUDE OPCLIB(EQQYCOM)
  SETCODE AC(1)
  ENTRY   CEESTART
  NAME    EQQPIFT(R)
/*
//*
//EQQPIFT EXEC PGM=EQQPIFT,PARM='NOSTAE,NOSPIE',
//    COND=(4,LT,LKED), REGION=4096K
//STEPLIB  DD DISP=SHR,DSN=SVIOLA.SEQQLMD0
//         DD DISP=SHR,DSN=USER.OPC23.LINKLIB
//EQQMLIB  DD DSN=EQQ.V2R3M0.SEQQMSG0,DISP=SHR
//EQQYPARM DD DISP=SHR,DSN=XXXX.YYYY.ZZZZ(YPARM)
//EQQMLOG  DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//EQQDUMP  DD SYSOUT=*
//EQQDMSG  DD SYSOUT=*
//CARDIN   DD *
.......................
/*
//*
```

SMP/E samples

The following SEQQSAMP members relate to SMP/E processes.

Environment setup

You can use the sample library members EQQALSMP, EQQDDDEF, and EQQALLOC to create and initialize the SMP/E environment and the HCL Workload Automation for Z product libraries that are needed to support the installation and continuing maintenance of HCL Workload Automation for Z.

The EQQALSMP job performs the following functions:

- Initializes an SMP/E CSI, adding a global zone, and the HCL Workload Automation for Z FMID.

The EQQDDDEF job sets up DDDEFs for all HCL Workload Automation for Z data sets to provide for basic JCL requirements for RECEIVE, APPLY, and ACCEPT processing.

The EQQALLOC job allocates all Tivoli target and distribution libraries. The JCL also contains a number of steps, which are currently commented out. You can use those steps to delete the HCL Workload Automation for Z libraries if you need to reinstall the product.

RECEIVE processing

The sample library members EQQRECVE, EQQRECVS, EQQRECVJ, EQQRECVD, and EQQRECVK contain JCL that you can use to run SMP/E RECEIVE processing for HCL Workload Automation for Z data sets. These library members enable you to receive the following HCL Workload Automation for Z features:

- Tracker
- Controller
- Java™ enabler

The EQQRECVE job receives all the scheduler base and tracker components plus the English feature for the controller.

You might need to change the distribution library and zone name to reflect those defined in the HCL Workload Automation for Z CSI.

For further details, see the HCL Workload Automation for Z *HCL Workload Scheduler for Z: Program Directory*.

Note: The *Program Directory* refers to trackers as *agents*, and to the controller as the *engine*.

APPLY processing

The sample library members EQQAPPLE, EQQAPPLS, EQQAPPLJ, EQQAPPLD, and EQQAPPLK contain JCL that you can use to run SMP/E APPLY processing for HCL Workload Automation for Z. These members enable you to apply the following features:

- Tracker
- Controller
- Java™ enabler

The EQQAPPLE job applies all the scheduler base and tracker components plus the English feature for the controller.

You might need to change the distribution library and zone name to reflect those defined in the HCL Workload Automation for Z CSI.

For further details, see the *HCL Workload Scheduler for Z: Program Directory*.

Note: The *Program Directory* refers to trackers as *agents*, and to the controller as the *engine*.

ACCEPT processing

The sample library members EQQACPTE, EQQACPTS, EQQACPTJ, EQQACPTD, and EQQACPTK contain JCL that you can use to run SMP/E ACCEPT processing for HCL Workload Automation for Z. These members enable you to accept the following features:

- Tracker
- Controller
- Java™ enabler

The EQQACPTE job accepts all the scheduler base and tracker components plus the English feature for the controller.

You might need to change the distribution library and zone name to reflect those defined in the HCL Workload Automation for Z CSI.

For further details, see the *HCL Workload Scheduler for Z: Program Directory*.

Note: The *Program Directory* refers to trackers as *agents*, and to the controller as the *engine*.

SMF exits

The following text provides details of the SEQQSAMP members relating to SMF exits.

Note: If version ASMA90 of the compiler reports errors, and the RMODE=ANY statement is defined, remove the RMODE=ANY statement from the sample exit.

Exit installation

The sample library member EQQSMF contains the JCL needed to assemble the SMF exits required for HCL Workload Automation for Z. The job also defines an SMP/E usermod to connect the SMF exits to your target zone.

A single usermod is used to define the three SMF exits. You can, if you prefer, define usermods for each exit.

To restore the JES exits as SMP/E usermods, use the samples EQQJER2U, EQQJER2V, and EQQJER3U.

Job step termination exit

The sample library member EQQACTR1 contains the assembler source code of an SMF job/step termination exit, IEFACTRT. The sample contains two subroutines:

- OPCASUB provides the necessary HCL Workload Automation for Z code to track job- and step-end events.
- LOCALSUB generates WTO messages for step- and job-end.

If you use the HCL Workload Automation for Z Restart and Cleanup functionality or other functions, such as the one related to the NOERROR table (for details, see *Customization and Tuning*), you are required to install this exit. Use the sample provided with the product to install this exit.

From the introduction of the usability enhancement on, the IEFACTRT exit creates two different tables in the joblog, the Steptable ad the Not_Executed_Step_Table.

If you use the HCL Workload Automation for Z Restart and Cleanup functionality or other functions, such as the one related to the NOERROR table, and you want to replace this subroutine with your own, you need to comply with the following restrictions:

- The fields JOBNAME, STEPNAME, PROCSTEP and STEPNO must continue to be filled on the basis of the following logic:
    - JOBNAME must contain the name of the job.
    - STEPNAME is the label of the EXEC PROC=... Card and must be filled only if a PROC is used.
    - PROCSTEP is the label of the EXEC PGM=... Card and must be filled also if a PROC is not used.
    - STEPNO must contain the sequence number of the steps inside the job.
- The JOBNAME, STEPNAME, PROCSTEP identifiers in the tables header must match the values specified in the *HDRJOBNAME*, *HDRSTEPNAME*, and *HDRPROCNAME* parameters of the DSTOPTS DATASTORE statement.
- The layout of STEPTABLE and NOT_EXECUTED_STEP_TABLE must be in compliance with the following rules:
    - JOBNAME must be preceded by a hyphen sign (-), some characters can be inserted between a hyphen sign and the JOBNAME.
    - JOBNAME must be followed by a blank.
    - STEPNAME must be preceded and followed by a blank.
    - PROCSTEP must be preceded and followed by a blank.
    - STEPNO must be preceded by a blank.
    - STEPNO must follow the PROCSTEP in the NOT_EXECUTED_STEP_TABLE.
- NOT_EXECUTED_STEP_TABLE must be aligned to STEPTABLE as far as it concerns JOBNAME, STEPNAME and PROCSTEP information.
- The string "JOBXXXXX ENDED. NAME-" must be aligned so that the JOBNAME JOBXXXXX is under the JOBNAME header.
- JOBNAME, STEPNAME, PROCSTEP position in the STEPTABLE and in the NOT_EXECUTED_STEP_TABLE must match the values specified in the *HDRJOBLENGTH*, *HDRSTEPLENGTH*, *HDRPROCLENGTH* parameters of the DSTOPTS DATASTORE statement. STEPNO position in the STEPTABLE must match the value specified in *HDRSTEPNOLENGTH*.
- User-customized records issued in the STEPTABLE and in the NOT_EXECUTED_STEP_TABLE must be avoided.

Initialization exit

The sample library member EQQUJI1 contains the assembler source code of an SMF initialization exit, IEFUJI. HCL Workload Automation for Z uses events generated from the exit to track job start information.

If your installation is already using an IEFUJI, incorporate the code into your existing exit and reassemble.

Record write exits

The sample library member EQQU831 contains the assembler source code of a record write exit, IEFU83. HCL Workload Automation for Z uses events generated from the exit to track print group and purge information.

If your installation is already using an IEFU83, incorporate the code into your existing exit and reassemble.

You can optionally include support for both the data set triggering and job-tracking functions using the EQQU831 sample. This provides you with a method to automatically generate a special resource availability depending on specific actions affecting data sets. The event can be used by HCL Workload Automation for Z to change the status of a special resource to make it available for operations and/or to trigger an application to be added to the current plan. You specify the data sets you want special resource availability events for using a specific macro, as described in Invoking the EQQLSENT macro on page 361. For more information about data set triggering, see Implementing support for data set triggering on page 100. Use the EQQSMF sample to install EQQU831.

If you do not track print operations through HCL Workload Automation for Z, and you do not want to include data set triggering support, you need not change IEFU83.

JES exits

The following text provides details of the SEQQSAMP members relating to JES exits.

Note: If version ASMA90 of the compiler reports errors, and the RMODE=ANY statement is defined, remove the RMODE=ANY statement from the sample exit.

Exit installation

The sample library contains a number of members to assemble and link-edit JES exits. EQQJES2, EQQJES21, and EQQJES3 provide sample JCL to assemble and link-edit of JES2 and JES3 exits respectively. However, it is recommended that you use members EQQJES2U, EQQJES2V, and EQQJES3U. These samples provide the JCL to install the JES exits as SMP/E usermods. The usermods are defined so that both the JES and HCL Workload Automation for Z target zones are informed of the dependencies. This ensures that future maintenance, to either the JES component or the HCL Workload Automation for Z component, will be handled correctly.

JES2 QMOD phase change exit

The sample library member EQQXIT51 contains the assembler source code of the JES2 QMOD Phase Change exit, JES EXIT51. HCL Workload Automation for Z uses JES2 EXIT51 to detect job errors occurring during the JES2 input phase, and to trigger the creation of IJ2 events for started task.

JES2 JCT I/O exit

The sample library member EQQXIT74 contains the assembler source code of a JES2 JCT I/O exit, JESEXIT7. EQQXIT74 is used for JES2. HCL Workload Automation for Z uses JESEXIT7 to detect new jobs on the internal reader and also to detect output group purge.

If you are already using a JESEXIT7, and want to keep the HCL Workload Automation for Z job-tracking support in a separate load module, you can specify that JES use multiple EXIT7 modules in your JES2 parameters.

JES3 OSE modification exit

The sample library member EQQUX191 contains the assembler source code of a JES3 OSE modification exit, IATUX19. HCL Workload Automation for Z uses events generated from the exit to detect output group purge.

If you are already using an IATUX19, you should include the code in your existing exit and reassemble.

Note: If you are using JES3 Exit IATUX72 then this exit must return with R15 = 8 to call IATUX19.

JES3 input service final-user exit

The sample library member EQQUX291 contains the assembler source code of a JES3 input service final-user exit, IATUX29. HCL Workload Automation for Z uses events generated from the exit to detect new jobs on the internal reader.

If you are already using an IATUX29, then you should incorporate the code into your existing exit and reassemble.

RACF® samples

The following text provides details of the SEQQSAMP members relating to RACF® changes, which are required for HCL Workload Automation for Z security.

Class descriptor table

The sample library member EQQ9RFDE provides the class descriptor entry required to define the HCL Workload Automation for Z security environment to RACF®, or a functionally equivalent product.

Each class descriptor contains control information needed by RACF® to validate class names and is a CSECT in the load module ICHRRCDE.

You can use member EQQ9SMDE to install ICHRRCDE as an SMP/E usermod on the RACF® target zone.

Router table

The sample library member EQQ9RF01 provides the router table entry required to define the HCL Workload Automation for Z security environment to RACF®, or a functionally equivalent product.

This is a sample RACF® router table that provides action codes to determine if RACF® is invoked on behalf of the RACROUTE macro.

You can use member EQQ9SM01 to install ICHRFR01 as an SMP/E usermod on the RACF® target zone.

EQQYCBAG sample

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 batch loader control statement format.

The group applications, as well as other applications, can be modified via the batch loader control statements. From then on, you can use the sequential file as input for the batch loader 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 batch loader.

EQQBENCR sample

The EQQBENCR member of the EQQSAMP library provides a sample that you can use to encrypt the passwords written in plain text in the USRREC statement of the USRINFO configuration member.

Appendix C. Invoking the EQQEXIT macro

The sample event-tracking exits shipped with HCL Workload Automation for Z are written in assembler language. The event-tracking code in these exits is generated by an assembler macro called EQQEXIT. The following sections describe how you invoke the EQQEXIT macro. This appendix contains General-use Programming Interface and Associated Guidance Information.

Invoking EQQEXIT in SMF exits

EQQEXIT establishes its own addressability in SMF exits. It saves and restores all used registers. To do this, it expects Register 13 to point to a standard z/OS save area.

There are two ways to invoke the EQQEXIT macro in an SMF exit:

- Invoke EQQEXIT with all registers unchanged since the exit was called (except Register 15).
- Save all registers on entry to the exit and then invoke EQQEXIT by specifying the address of the initial save area.

In both cases, the EQQEXIT macro must be invoked in Supervisor state, PSW key 0.

Invoking EQQEXIT in JES exits

In JES exits, EQQEXIT must be invoked in Supervisor state, PSW key 1. EQQEXIT expects code addressability to be already established. It also expects registers to be set up as follows:

- **EXIT7**

  R0

  JCT read/write indicator (JES2 SP™ Version 3, or earlier); address of a parameter list mapped by the JES2 $XPL macro (JES2 SP™ Version 4, or later).

  R1

  Address of the JCT being read or written.

  R13

  Address of the current PCE.
- **EXIT51**

  R1

  Address of a parameter list mapped by the JES2 $XPL macro.
- **IATUX19**

  R8

  Address of the current JDS entry.

R9

Address of the current RESQUEUE entry.

R11

Address of the current FCT entry.

R12

Address of the TVTABLE entry.

- **IATUX29**

  R11

  Address of the current FCT entry.

  R13

  Address of the input-service data area for the current function.

Note that these register conventions are already set up when the exit is called. You must invoke EQQEXIT while these registers are unchanged.

If a shipped JES exit example (or the EQQEXIT macro) has been user–modified, make sure that it does not prevent or filter the tracking of HCL Workload Automation for Z itself.

See the NOTES section of the EQQEXIT prolog for information about the register contents that are destroyed by EQQEXIT in JES exits.

Macro invocation syntax for EQQEXIT

Purpose

EQQEXIT produces HCL Workload Automation for Z event-tracking exit code by generating assembler code to perform in an SMF or JES exit.

Syntax

EXIT=*exit name*
REG13=*address of save area*
MAPMAC={<u>YES</u>|NO}
SETUID={YES|<u>NO</u>}
SRREAD={YES|<u>NO</u>|NONE}
SNA={<u>YES</u>|NO}

Parameters

EXIT=*exit name*

A required keyword defining the name of the exit in which the macro is used. The following names can be specified: IEFACTRT, IEFUJI, IEFU83, EXIT7, IATUX19, and IATUX29. Except for the EXIT7 exit, a warning message is issued if the name of the current CSECT differs from the name specified by the EXIT keyword.

REG13=*address of save area*

An optional keyword defining the address of the current-register save area when the SMF or JES exit was called. The default for this keyword depends on the name specified by the EXIT keyword. If the current exit is EXIT7, the default is PCELPSV. If the current exit is IATUX19 or IATUX29, the default is FCTSAVCH. In all other cases, the default is the second fullword in the current save area (if the current save area is properly chained, and the previous save area contains the registers at entry to the exit).

If the default does not apply, the REG13 keyword must be specified. Its value must be a fullword pointing to the save area that was used to store all the registers when the exit was entered.

MAPMAC={YES|NO}

An optional keyword specifying whether the macro should generate the required assembler mapping macros. The default is to generate these mapping macros. The following mapping macros are required by EQQEXIT code: CVT, IEFJESCT, IEFJSSOB, and IEFJSSIB. The IEFACTRT exit also requires the IEFJMR macro.

If you specify NO, the IEFU83 exit requires mapping of the SMF records IFASMFR 14 and IFASMFR 64. You must label them SMF14REC and SMF64REC, respectively. For example:

```
SMF14REC DSECT           * SMF RECORD 14 MAPPING
         IFASMFR 14      * DATA SET ACTIVITY RECORD
```

SETUID={YES|NO}

An optional keyword specifying whether the macro should generate code to place the current user ID in the JMRUSEID field when the IEFUJI exit is taken. Specify YES to generate this code. If you specify NO, which is the default, the JMRUSEID field is not updated. You are recommended to specify YES if you use the current user ID to filter data set close events. You need these mapping macros when you specify YES: IHAPSA, IHAASCB, IHAASXB, and IHAACEE.

SRREAD={YES|NO|NONE}

An optional keyword defining whether a resource availability event should be generated when a data set is closed after being opened for read processing.

When YES is specified, an SR event is generated each time a data set is closed after being opened for either read or output processing.

When NO is specified or defaulted, the SR event is generated only when a data set has been opened for output processing. The event is not generated if the data set has been opened for read processing.

When you specify NONE, no data set triggering is performed.

For more information about the data set triggering function, see Implementing support for data set triggering on page 100.

SNA={YES|NO}

An optional keyword specifying whether JES3 SNA NJE is supported.

Messages

The following messages can be generated at assembly time:

- WARNING: SNA KEYWORD IS ONLY USED FOR EXIT = IATUX19
- WARNING: SNA VALUE *SNA* IS NOT RECOGNIZED
- WARNING: EXIT NAME DIFFERS FROM CURRENT CSECT NAME
- WARNING: MAPMAC VALUE *MAPMAC* IS NOT RECOGNIZED
- WARNING: SRREAD KEYWORD IS ONLY USED FOR EXIT=IEFU83
- WARNING: SRREAD VALUE NOT RECOGNIZED, YES OR NO ARE THE ONLY VALID VALUES
- EXIT NAME *EXIT* IS NOT SUPPORTED

Return codes

The following return codes can be generated at assembly time:

4

Input invalid, check for warning messages.

12

Unsupported exit specified for the EXIT keyword.

Appendix D. Invoking the EQQLSENT macro

The following procedure is supported only for compatibility with earlier versions. To use the current support for data set triggering, see the procedure for running event-driven workload automation described in *Managing the Workload*.

When the data set triggering function is used, you specify the data sets for which you want events generated by building the data set selection table EQQDSLST. The EQQDSLST is created by invoking the EQQLSENT macro. The following sections describe how you invoke the EQQLSENT macro. This appendix contains General-use Programming Interface and Associated Guidance Information.

Note: The current support for data set triggering is based on the EQQEVLST configuration file. If EQQJCLIB contains both EQQEVLST and EQQDSLST, the resulting triggering selection table is the union of EQQEVLST and EQQDSLST. In this case, EQQEVLST data is processed first. If EQQJCLIB contains only EQQDSLST, the tracker loads it as triggering selection table.

Invoking EQQLSENT to create EQQDSLST

The EQQLSENT macro is used to create entries in the data set triggering selection table. The selection table is loaded into ECSA when the HCL Workload Automation for Z event writer is started.

The sample EQQLSJCL in the SEQQSAMP library can be used to invoke the EQQLSENT macro.

Macro invocation syntax for EQQLSENT

Purpose

EQQLSENT produces an entry in the data set triggering selection table, EQQDSLST. EQQDSLST is used in SMF exit IEFU83 by the data set triggering function to decide which SMF records to process. When an SMF 14, 15, or 64 record matches a condition in EQQDSLST, a special resource availability event is created and broadcast to all HCL Workload Automation for Z subsystems defined on the system where the SMF record was created.

Format

STRING= *string*|**LASTENTRY**
POS= *numeric position*
USERID= *user ID filter criteria*
JOBNAME= *jobname filter criteria*
AINDIC={Y|N}
LIFACT={Y|N|R}
LIFTIM=*interval*

Parameters

STRING=*string*|LASTENTRY

Required keyword specifying the character string to be searched for. The string can be 1 to 44 characters long. To identify the fully qualified last level of a data set name, add a space as the last character and enclose the string in single quotation marks. Consider this example. You have two data sets:

```
         DSN.NAME.AB
         DSN.NAME.ABC
```

Specify `STRING=DSN.NAME.AB,POS=1` if you want SR availability events created for both data sets. Specify `STRING='DSN.NAME.AB ',POS=1` if you want events created only for the first data set.

When EQQLSENT is invoked with STRING=LASTENTRY it generates an end of table indicator. After having invoked EQQLSENT with keyword parameters STRING and POS a number of times, EQQLSENT must be invoked one last time with STRING=LASTENTRY in order to complete the table.

To create an empty EQQDSLST, just invoke EQQLSENT once, with STRING=LASTENTRY. When an empty list is used by IEFU83, no SR events are created.

POS=*numeric position*

A required keyword specifying the numeric position where the string begins.

USERID=*string*

Optional keyword specifying a generic character string to be compared with the SMF*xx*UID field, which contains the user identification associated with the job, started task, or TSO user that requested the activity against the data set that resulted in the data set close. The string can be 1 to 8 characters long. For this parameter the following wildcards are allowed:

- `*`: to match any sequence of characters.
- `%`: to match any single character. For example, if you specify `AB%`, `ABC` is a match, `AB` or `ABCD` are not a match.

Note: The SMF user ID field may contain a blank value. See *z/OS® System Management Facilities* for more information about the SMF*xx*UID or SMF*xx*UIF field.

If you need to control SR availability events based on the user ID and the SMF value is blank in your installation, consider using the IEFUJI exit to insert the user ID. You are recommended to specify SETUID=YES on the EQQEXIT macro when you generate the IEFUJI exit: this sets the JMRUSEID field, which SMF then copies to the SMF user ID field.

If you want to update the JMRUSEID field yourself, the user ID is most easily taken from the ACEEUSRI field in the ACEE, pointed to from the ASXB, pointed to from the ASCB. This can be located as follows: `PSAAOLD ===> ASCB ACSBASXB ===> ASXB ASXBSENV ===> ACEE ACEEUSRI ===> userid`

The DSECTs needed are mapped by these macros:

Area	Macro	Library
PSA	IHAPSA	SYS1.MACLIB
ASCB	IHAASCB	SYS1.MACLIB
ASXB	IHAASXB	SYS1.MODGEN
ACEE	IHAACEE	SYS1.MACLIB

The JMR, mapped by IEFJMR, is already available in the EQQEXIT expansion in IEFUJI.

JOBNAME=*string*

Optional keyword specifying a generic character string to be compared with the SMF14JBN, SMF15JBN, or SMF64JMN field, which contains the name of the job, started task or TSO user that requested the activity against the data set that resulted in the data set close. The string can be 1 to 8 characters long. For this parameter the following wildcards are allowed:

- `*`: to match any sequence of characters.
- `%`: to match any single character. For example, if you specify `AB%`, `ABC` is a match, `AB` or `ABCD` are not a match.

If the data set is to be processed by FTP, JOBNAME corresponds to the ** USERID ** under which the data set is received. That is, the USERID supplied when the remote host opened the FTP session to PUT the data set, or when a local user (or batch job) opened the FTP session to GET the data set.

AINDIC={Y|N}

Optional keyword specifying that the special resource is available (Y) or unavailable (N). The default is that the resource available.

LIFACT={Y|N|R}

Optional keyword specifying the value to which the global availability of the special resource is reset, after the interval of time specified by LIFTIM has expired. Allowed values are:

Y

Sets the global availability to Yes

N

Sets the global availability to No

R

Sets the global availability to blank

This keyword is valid only if LIFTIM is specified. The default is R.

LIFTIM=*interval*

Optional keyword specifying the interval of time, in minutes, after which the global availability of the special resource is reset to the value specified by LIFACT. The allowed range is from 1 to 999999.

Note:

1. The output from assembling the EQQLSENT macro must be placed in the EQQDSLST member in the data set referenced by the ddname EQQJCLIB.
2. Generation Data Group data sets are specified by the group name. For example, when a GDG data set with the name 'DSN.OPCSUBS.GDG.G0001V00' is closed the special resource event contains resource name 'DSN.OPCSUBS.GDG'.
3. For a partitioned data set, the member name is not part of the resource name in the SR event.
4. For VSAM data sets the resource name in the SR event is the cluster name (without the DATA or INDEX suffix).

Example

Examples

```
EQQLSENT STRING=SYS1.MAN,POS=1
EQQLSENT STRING='TEST.DSCLOSE ',POS=1,USERID=SYSOP
EQQLSENT STRING=CP2,POS=12
EQQLSENT STRING=EQQDATA.EXCL,POS=5
EQQLSENT STRING='DSN.OPCSUBS.GDG ',POS=1
EQQLSENT STRING=LASTENTRY
END
```

In this example, SMF records with:

- A data set name beginning with SYS1.MAN, or
- Data set name TEST.DSCLOSE and user ID SYSOP
- Records with CP2 in position 12, such as DSN.OPCSUB.CP2, or
- Records that have EQQDATA.EXCL starting in position 5
- The root of a GDG data set name

will cause SR availability events to be generated.

Messages

The following messages can be generated at assembly time:

- KEYWORD STRING IS REQUIRED
- KEYWORD POS IS REQUIRED
- POSITION MUST BE BETWEEN 1 AND 43
- NULL NAME NOT VALID

- NAME (STRING) GREATER THAN 44 CHARACTERS
- POSITION INVALID FOR NAME (STRING)
- USERID STRING NOT VALID
- JOBNAME STRING NOT VALID
- AINDIC MUST BE EITHER Y OR N
- POSITION NOT VALID FOR NAME (STRING)
- LIFACT MUST BE Y, N, OR R
- LIFTIM LENGTH NOT VALID
- LIFTIM VALUE NOT VALID
- LIFTIM VALUE 0 NOT ALLOWED

Return codes

The following return code can be generated at assembly time:

12

Input invalid, check error messages.

Notices

This document provides information about copyright, trademarks, terms and conditions for product documentation.

© Copyright IBM Corporation 1993, 2016 / © Copyright HCL Technologies Limited 2016, 2025

This information was developed for products and services offered in the US. This material might be available from HCL in other languages. However, you may be required to own a copy of the product or product version in that language in order to access it.

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

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

HCL
330 Potrero Ave.
Sunnyvale, CA 94085
USA
Attention: Office of the General Counsel

For license inquiries regarding double-byte character set (DBCS) information, contact the HCL Intellectual Property Department in your country or send inquiries, in writing, to:

HCL
330 Potrero Ave.
Sunnyvale, CA 94085
USA
Attention: Office of the General Counsel

HCL TECHNOLOGIES LTD. PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some jurisdictions do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

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

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

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

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

HCL
330 Potrero Ave.
Sunnyvale, CA 94085
USA
Attention: Office of the General Counsel

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by HCL under terms of the HCL Customer Agreement, HCL International Program License Agreement or any equivalent agreement between us.

The performance data discussed herein is presented as derived under specific operating conditions. Actual results may vary.

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

This information is for planning purposes only. The information herein is subject to change before the products described become available.

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

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to HCL, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. HCL, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. HCL shall not be liable for any damages arising out of your use of the sample programs.

© (HCL Technologies Limited) (2025).
Portions of this code are derived from Sample Programs.
© Copyright 2016

Trademarks

HCL®, and other HCL graphics, logos, and service names including "hcltech.com" are trademarks of HCL. Except as specifically permitted herein, these Trademarks may not be used without the prior written permission from HCL. All other trademarks not owned by HCL that appear on this website are the property of their respective owners, who may or may not be affiliated with, connected to, or sponsored by HCL.

Adobe™, the Adobe™ logo, PostScript™, and the PostScript™ logo are either registered trademarks or trademarks of Adobe™ Systems Incorporated in the United States, and/or other countries.

IT Infrastructure Library™ is a Registered Trade Mark of AXELOS Limited.

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

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

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

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

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

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

ITIL™ is a Registered Trade Mark of AXELOS Limited.

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

Terms and conditions for product documentation

Permissions for the use of these publications are granted subject to the following terms and conditions.

Applicability

These terms and conditions are in addition to any terms of use for the HCL website.

Personal use

You may reproduce these publications for your personal, noncommercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative work of these publications, or any portion thereof, without the express consent of HCL.

Commercial use

You may reproduce, distribute and display these publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these publications, or reproduce, distribute or display these publications or any portion thereof outside your enterprise, without the express consent of HCL.

Rights

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the publications or any information, data, software or other intellectual property contained therein.

HCL reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the publications is detrimental to its interest or, as determined by HCL, the above instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

HCL MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.

Index