IBM Rational Developer for System z
Version 8.5

Configuration Guide

SC23-7658-07

IBM Rational Developer for System z
Version 8.5

Configuration Guide

SC23-7658-07

Note

Before using this document, read the general information under “Documentation notices for IBM Rational Developer for System z” on page 197.

Eighth edition (June 2012)

This edition applies to IBM Rational Developer for System z Version 8.5 (program number 5724-T07) and to all subsequent releases and modifications until otherwise indicated in new editions.

Order publications by phone or fax. IBM Software Manufacturing Solutions takes publication orders between 8:30 a.m. and 7:00 p.m. eastern standard time (EST). The phone number is (800) 879-2755. The fax number is (800) 445-9269. Faxes should be sent Attn: Publications, 3rd floor.

You can also order publications through your IBM representative or the IBM branch office serving your locality. Publications are not stocked at the address below.

IBM welcomes your comments. You can send your comments by mail to the following address:

IBM Corporation
Attn: Information Development Department 53NA
Building 501 P.O. Box 12195
Research Triangle Park NC 27709-2195
USA

You can fax your comments to: 1-800-227-5088 (US and Canada)

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

Contents

Figures

Tables

About this document

This document discusses the configuration of the IBM Rational Developer for System z functions. It includes instructions on how to configure IBM Rational Developer for System z Version 8.5 on your z/OS® host system.

From here on, the following names are used in this manual:

- *IBM Rational Developer for System z* is called *Developer for System z*.
- *Common Access Repository Manager* is abbreviated to *CARMA*.
- *Software Configuration and Library Manager Developer Toolkit* is called *SCLM Developer Toolkit*, abbreviated to *SCLMDT*.
- IBM z/OS Automated Unit Testing Framework is called zUnit.
- *z/OS UNIX System Services* is called *z/OS UNIX*.
- *Customer Information Control System Transaction Server* is called *CICSTS*, abbreviated to *CICS®*.

For earlier releases, including IBM WebSphere Developer for System z, IBM WebSphere Developer for zSeries and IBM® WebSphere Studio Enterprise Developer, use the configuration information found in the Host Configuration Guide and Program Directories for those releases.

This document is part of a set of documents that describe Developer for System z host configuration. Each of these documents has a specific target audience. You are not required to read all documents to complete the Developer for System z configuration.

- *Rational® Developer for System z® Host Configuration Guide* (SC23-7658) describes in detail all planning tasks, configuration tasks, and options (including optional ones) and provides alternative scenarios.
- *Rational Developer for System z Host Configuration Reference* (SC14-7290) describes Developer for System z design and gives background information for various configuration tasks of Developer for System z, z/OS components, and other products (such as WLM and CICS) related to Developer for System z.
- *Rational Developer for System z Host Configuration Quick Start Guide* (GI11-9201) describes a minimal setup of Developer for System z.
- *Rational Developer for System z Host Configuration Utility* (SC14-7282) describes the Host Configuration Utility, an ISPF panel application that guides you through basic and common optional customization steps for Developer for System z.

The information in this document applies to all IBM Rational Developer for System z Version 8.5 packages including IBM Rational Developer for zEnterprise®.

Who should use this document

This document is intended for system programmers installing and configuring IBM Rational Developer for System z Version 8.5.

This document lists in detail the different steps needed to do a full setup of the product, including some non-default scenarios. Background information that can help you plan and execute the configuration can be found in the *IBM Rational*

Developer for System z Host Configuration Reference (SC14-7290). To use this document, you need to be familiar with the z/OS UNIX System Services and MVS™ host systems.

Summary of changes

This section summarizes the changes for *IBM Rational Developer for System z Version 8.5 Host Configuration Guide*, SC23-7658-07 (updated June 2012).

Technical changes or additions to the text and illustrations are indicated by a vertical line to the left of the change.

This document contains information previously presented in *Rational Developer for System z Version 8.0.3 Host Configuration Guide*, SC23-7658-06.

New information:

- New optional directives in FEJJCNFG. See "FEJJCNFG, JES Job Monitor configuration file" on page 25.
- New optional directives in rsed.envvars. See "rsed.envvars, RSE configuration file" on page 30.
- New optional configuration file. See "(Optional) include.conf, Forced includes for C/C++ content assist" on page 99.
- New optional component. See "(Optional) Include preprocessor support" on page 104.
- New optional component. See "(Optional) xUnit support for Enterprise COBOL and PL/I" on page 105.
- New optional component. See "(Optional) DB2 and IMS debug support" on page 108.
- New and enhanced operator commands. See Chapter 10, "Operator commands," on page 169.
- Version 8.5 migration information. See "Migrate from version 8.0.1 to version 8.5" on page 145.

Removed information:

- Support for File Manager Integration has changed, resulting in the removal of most information in section "(Optional) File Manager support" on page 109.

This document contains information previously presented in Rational Developer for System z Version 8.0.1 Host Configuration Guide, SC23-7658-05.

New information:

- New directives in `rsed.envvars`. See "rsed.envvars, RSE configuration file" on page 30.
- Added support for CA Endevor® SCM background actions. See "CA Endevor® SCM RAM batch actions" on page 70.
- Added support for CA Endevor® SCM packages. See "CARMA Repository Access Managers (RAMs)" on page 67.
- New directives in `pushtoclient.properties`. See "(Optional) pushtoclient.properties, Host-based client control" on page 93.
- File Manager Integration is deprecated. See "(Optional) File Manager support" on page 109.

- New and enhanced operator commands. See Chapter 10, “Operator commands,” on page 169.
- Version 8.0.3 migration information. See “Version 8.0.x migration notes” on page 151.
- Version 8.0.1 sample migration scenarios. See “Sample migration scenarios” on page 159.
- New publication, *IBM Rational Developer for System z Messages and Codes* (SC14-7497).

This document contains information previously presented in *Rational Developer for System z Version 7.6.1 Host Configuration Guide*, SC23-7658-04.

New information:

- New directives in FEJJCNFG. See “FEJJCNFG, JES Job Monitor configuration file” on page 25.
- New directives in rsed.envvars. See “rsed.envvars, RSE configuration file” on page 30.
- The CARMA chapter has been reorganized, and some new information has been added. See Chapter 3, “(Optional) Common Access Repository Manager (CARMA),” on page 49.
- New configuration file pushtoclient.properties. See “(Optional) pushtoclient.properties, Host-based client control” on page 93.
- Version 8.0.1 migration information. See “Migrate from version 7.6 to version 8.0.1” on page 153.

Removed information:

- The information previously presented in *Rational Developer for System z Version 7.6.1 Host Configuration Guide* (SC23-7658-04) is now split into 2 documents: *Rational Developer for System z Host Configuration Guide* (SC23-7658) and *Rational Developer for System z Host Configuration Reference* (SC14-7290).
- Information regarding APPC setup has moved to white paper *Using APPC to provide TSO command services* (SC14-7291).
- Information regarding CARMA using ISPF Client Gateway has moved to white paper *Using ISPF Client Gateway to provide CARMA services* (SC14-7292).
- “(Optional) Host based property groups” section in “(Optional) Other customization tasks” (described propertiescfg.properties).
- “(Optional) Host based projects” section in “(Optional) Other customization tasks” (described projectcfg.properties).
- “(Optional) Uneditable characters” section in “(Optional) Other customization tasks” (described uchars.settings).
- “Version 7.6.1 migration notes” section in “Migration guide”.

This document contains information previously presented in *Rational Developer for System z Version 7.6. Host Configuration Guide*, SC23-7658-03.

New information:

- Corrections and additional information presented in *Rational Developer for System z v7.6 Host Configuration Release Notes®* are incorporated.
- Version 7.6.1 specific migration notes.
- Added document overview. See “Description of document content” on page xiv.

- Support for 64-bit Java. See *Rational Developer for System z Prerequisites* (SC23-7659).
- Product configuration through ISPF panels. See “Pre-configuration considerations” on page 6.
- New directives in `rsed.envvars`. See “rsed.envvars, RSE configuration file” on page 30.
- New proclib members. See “ELAXF* remote build procedures” on page 23.
- New CARMA VSAM layout. See “CARMA VSAM migration notes” on page 66.
- Supporting multiple CARMA RAMs. See “(Optional) Supporting multiple RAMs” on page 73.
- New Application Deployment Manager options. See "Administrative utility" in the *Host Configuration Reference* (SC14-7290).
- New Application Deployment Manager VSAM layout. See "Administrative utility migration notes" in the *Host Configuration Reference* (SC14-7290).
- New operator commands. See Chapter 10, “Operator commands,” on page 169.
- New console messages. See “Console messages” on page 179.
- Workload Management information. See "WLM considerations" in the *Host Configuration Reference* (SC14-7290).

Description of document content

This section summarizes the information presented in this document.

Planning

Use the information in this chapter to plan the installation and deployment of Developer for System z.

Basic customization

The following customization steps are for a basic Developer for System z setup:

- “Customization setup” on page 13
- “PARMLIB changes” on page 14
- “PROCLIB changes” on page 20
- “Security definitions” on page 25
- “FEJJCNFG, JES Job Monitor configuration file” on page 25
- “rsed.envvars, RSE configuration file” on page 30
- “ISPF.conf, ISPF's TSO/ISPF Client Gateway configuration file” on page 46

(Optional) Common Access Repository Manager (CARMA)

Common Access Repository Manager (CARMA) is a server platform for Repository Access Managers (RAMs). A RAM is an Application Programming Interface (API) for a z/OS based Software Configuration Manager (SCM). By wrapping the SCM functionality in a RAM, a single API is available for a client to access any supported SCM.

Developer for System z provides multiple pre-built RAMs, as well as source code examples for creating your own RAM.

The IBM® Rational® Developer for System z Interface for CA Endevor® Software Configuration Manager gives Developer for System z clients direct access to CA Endevor® SCM.

(Optional) SCLM Developer Toolkit

SCLM Developer Toolkit provides the tools needed to extend the capabilities of SCLM to the client. SCLM itself is a host-based source code manager that is shipped as part of ISPF.

The SCLM Developer Toolkit has an Eclipse-based plug-in that interfaces to SCLM and provides for access to all SCLM processes for legacy code development as well as support for full Java and J2EE development on the workstation with synchronization to SCLM on the mainframe including building, assembling, and deployment of the J2EE code from the mainframe.

(Optional) Application Deployment Manager

Developer for System z uses certain functions of Application Deployment Manager as a common deployment approach for various components. Optional customization enables more features of Application Deployment Manager and can add the following services to Developer for System z:

- IBM CICS Explorer® provides an Eclipse-based infrastructure to view and manage CICS resources and enables greater integration between CICS tools.
- CICS Resource Definition (CRD) client and server provide the following functions:
  - CICS Resource Definition editor
  - Allow application developers to define CICS resources in a limited, controlled, and secure fashion.
  - Prevent CICS development access to unauthorized or incorrect VSAM data sets by providing the CICS administrator control over the physical data set name attribute in File definitions.
  - Miscellaneous CICS development aids
  - Miscellaneous CICS Web Service development aids

(Optional) Other customization tasks

This section combines a variety of optional customization tasks. Follow the instructions in the appropriate section to configure the desired service.

Customizations to Developer for System z configuration files:

- pushtoclient.properties, Host-based client control
- ssl.properties, RSE SSL encryption
- rsecomm.properties, RSE tracing
- include.conf, Forced includes for C/C++ content assist

Developer for System z related customizations to (or for) other products:

- DB2® stored procedure
- z/OS UNIX subprojects
- Include preprocessor support
- xUnit support for Enterprise COBOL and PL/I
- Enterprise Service Tools support
- CICS bidirectional language support
- Diagnostic IRZ messages for generated code
- Problem Determination Tools support
- DB2 and IMS™ debug support

- File Manager support
- WORKAREA and /tmp cleanup

Installation verification

After completing the product customization, you can use the Installation Verification Programs (IVPs) described in this chapter to verify the successful setup of key product components.

Security definitions

This section describes the required and optional security definitions with sample RACF® commands.

Migration guide

This section highlights installation and configuration changes compared to previous releases of the product. It also gives some general guidelines to migrate to this release.

Operator commands

This section provides an overview of the available operator (or console) commands for Developer for System z.

Host configuration reference

This section summarizes the information in *IBM Rational Developer for System z Host Configuration Reference* (SC14-7290).

Chapter 1. Planning

Use the information in this chapter, *IBM Rational Developer for System z Prerequisites* (SC23-7659), to plan the installation and deployment of Developer for System z. The following subjects are described:

- “Migration considerations”
- “Planning considerations”
- “Preinstallation considerations” on page 2
- “Pre-configuration considerations” on page 6
- “Predeployment considerations” on page 9
- “Client checklist” on page 10

Migration considerations

Chapter 9, “Migration guide,” on page 143 describes installation and configuration changes compared to previous releases of the product. Use this information to plan your migration to the current release of Developer for System z.

Note:

- If you are a previous user of IBM Rational Developer for System z, IBM WebSphere Developer for System z, IBM WebSphere Developer for zSeries or IBM WebSphere Studio Enterprise Developer, it is recommended that you save the related customized files BEFORE installing IBM Rational Developer for System z Version 8.5. Refer to Chapter 9, “Migration guide,” on page 143 for an overview of files that required customization.
- Refer to "Running multiple instances" in the *Host Configuration Reference* (SC14-7290), if you plan on running multiple instances of Developer for System z.

Planning considerations

Product overview

Developer for System z consists of a client, installed on the user's personal computer, and a server, installed on one or more hosts. This documentation will focus on the host being a z/OS system. However, other operating systems, such as AIX® and Linux on System z, are also supported.

The client provides developers an Eclipse-based development environment that facilitates a uniform graphical interface to the host, and that, among other things, can offload work from the host to the client, saving resources on the host.

The host portion consists of several permanently active tasks and tasks that are started ad-hoc. These tasks allow the client to work with the various components of your z/OS host, such as MVS data sets, TSO commands, z/OS UNIX files and commands, job submit, and job output.

Developer for System z can also interact with subsystems and other application software on the host, such as CICS, Debug Tool, and Software Configuration Managers (SCMs), if Developer for system z is configured to do so, and if these (corequisite) products are available.

Refer to "Understanding Developer for System z" in the *Host Configuration Reference* (SC14-7290) to get a basic understanding of the Developer for System z design.

Refer to the Developer for System z web site, `http://www-01.ibm.com/software/rational/products/developer/systemz/`, or your local IBM representative to learn more about the functionality offered by Developer for System z.

Skill requirements

Minimal SMP/E skills are needed for a Developer for System z host installation.

The configuration of Developer for System z requires more than the typical system programming permissions and expertise, so assistance from others might be needed. Table 3 on page 4 and Table 4 on page 5 list the administrators needed for the required and optional customization tasks.

Time requirements

The amount of time required to install and configure the Developer for System z host components depends on various factors, such as:

- the current z/OS UNIX and TCP/IP configuration
- the availability of prerequisite software and maintenance
- whether or not OMVS segments are defined for Developer for System z users
- the availability of a user, who has successfully installed the client, to test the installation and report any problems that might occur

Experience has shown that the installation and configuration process of the Developer for System z host requires from one to four days to complete. This time requirement is for a clean installation performed by an experienced system programmer. If problems are encountered, or if the required skills are not available, then the setup will take longer.

Preinstallation considerations

Refer to *Program Directory for IBM Rational Developer for System z* (GI11-8298) for detailed instructions on the SMP/E installation of the product.

Refer to "Running multiple instances" in the *Host Configuration Reference* (SC14-7290) if you plan on running multiple instances of Developer for System z.

The file system (HFS or zFS) in which Developer for System z is installed must be mounted with the `SETUID` permission bit on (this is the system default). Mounting the file system with the `NOSETUID` parameter will prevent Developer for System z from creating the user's security environment, and will fail the connection request of the client. The same is true for the file systems hosting Java and z/OS UNIX binaries.

Requisite products

IBM Rational Developer for System z Prerequisites (SC23-7659) has a list of prerequisite software that must be installed and operational before Developer for System z will work. There is also a list of corequisite software to support specific features of Developer for System z. These requisites must be installed and operational at runtime for the corresponding feature to work as designed.

Plan ahead to have these requisite products available, as it might take some time, depending on the policies at your site. The key requisites for a basic setup are the following:

- z/OS 1.8 or higher
- | ISPF APAR OA38740 (TSO/ISPF Client Gateway)
- | Java 6.0 or higher

Required resources

Developer for System z requires the allocation of the systems resources listed in Table 1. The resources listed in Table 2 are required for optional services. Plan ahead to have these resources available, as it might take some time to get them, depending on the policies at your site.

Table 1. Required resources

Resource	Default value	Information
APF authorized data set	FEK.SFEKAUTH	"APF authorizations in PROGxx" on page 16
started task	JMON, RSED, and LOCKD	"PROCLIB changes" on page 20
port for host-confined use (JMON)	6715	"FEJJCNFG, JES Job Monitor configuration file" on page 25
port for host-confined use (LOCKD)	4036	"rsed.envvars, RSE configuration file" on page 30
port for client-host communication (RSED)	4035	"rsed.envvars, RSE configuration file" on page 30
port range for client-host communication (RSED)	any available port is used	"Defining the PORTRANGE available for RSE server" on page 39
z/OS UNIX server security definition	UPDATE permission to BPX.SERVER for RSED started task	"Define RSE as a secure z/OS UNIX server" on page 137
PassTicket security definitions	no default	"Define PassTicket support for RSE" on page 139
MVS build procedures	ELAXF*	"PROCLIB changes" on page 20

Table 2. Optional resources

Resource	Default value	Information
LINKLIST data set	FEK.SFEKAUTH and FEK.SFEKLOAD	Chapter 4, "(Optional) SCLM Developer Toolkit," on page 77
LPA data set	FEK.SFEKLPA	Chapter 3, "(Optional) Common Access Repository Manager (CARMA)," on page 49

Table 2. Optional resources (continued)

Resource	**Default value**	**Information**
port range for host-confined use	5227-5326 (100 ports)	Chapter 3, "(Optional) Common Access Repository Manager (CARMA)," on page 49
port for client-host communication	5129 for Web Service or 5130 for RESTful	Chapter 5, "(Optional) Application Deployment Manager," on page 85
CICS CSD update	multiple values	Chapter 5, "(Optional) Application Deployment Manager," on page 85
CICS JCL update	FEK.SFEKLOAD	• Chapter 5, "(Optional) Application Deployment Manager," on page 85 • "(Optional) CICS bidirectional language support" on page 106
security definitions	FEK.PTC.* profiles for push-to-client	"Push-to-client considerations" in the *Host Configuration Reference* (SC14-7290)
LDAP definitions	FEK.PTC.* groups for push-to-client	"Push-to-client considerations" in the *Host Configuration Reference* (SC14-7290)

The configuration of Developer for System z requires more than the typical system programming permissions and expertise, so minimal assistance from others might be needed. Table 3 and Table 4 on page 5 list the administrators needed for the required and optional customization tasks.

Table 3. Administrators needed for required tasks

Administrator	**Task**	**Information**
System	Typical system programmer actions are required for all customization tasks	N/A
Security	• Define OMVS segment for Developer for System z users • Define data set profiles • Define started tasks • Define operator command security • Define z/OS UNIX server profiles • Define application security • Define PassTicket support • Define program controlled data sets • Define program controlled z/OS UNIX files	"Security considerations" in *Host Configuration Reference* (SC14-7290)

Table 3. Administrators needed for required tasks (continued)

Administrator	Task	Information
TCP/IP	Define new TCP/IP ports	"TCP/IP considerations" in *Host Configuration Reference* (SC14-7290)
WLM	Assign started task goals to the servers and their child processes	"WLM considerations" in the *Host Configuration Reference* (SC14-7290).

Table 4. Administrators needed for optional tasks

Administrator	Task	Information
System	Typical system programmer actions are required for all customization tasks	N/A
Security	• Define data set profiles • Define program controlled data sets • Define permission to submit xxx* jobs • Define CICS transaction security • Add certificate for SSL • Define X.509 client certificate support • Define groups and profiles for push-to-client	• "Security considerations" in *Host Configuration Reference* (SC14-7290) • "CICSTS considerations" in the *Host Configuration Reference* (SC14-7290) • "Setting up SSL and X.509 authentication" in the *Host Configuration Reference* (SC14-7290) • "Push-to-client considerations" in the *Host Configuration Reference* (SC14-7290)
TCP/IP	Define new TCP/IP ports	"TCP/IP ports" in *Host Configuration Reference* (SC14-7290)
SCLM	• Define SCLM language translators for JAVA/J2EE support • Define SCLM types for JAVA/J2EE support	Chapter 4, "(Optional) SCLM Developer Toolkit," on page 77
CICS TS	• Update CICS region JCL • Update CICS region CSD • Define CICS group • Define CICS transaction names • Define a program to CICS	• Chapter 5, "(Optional) Application Deployment Manager," on page 85 • "(Optional) CICS bidirectional language support" on page 106
DB2	Define a DB2 stored procedure	"(Optional) DB2 stored procedure" on page 101
WLM	• Assign goals to a DB2 stored procedure • Assign goals to Developer for System z tasks	• "(Optional) DB2 stored procedure" on page 101 • "WLM considerations" in the *Host Configuration Reference* (SC14-7290)

Table 4. Administrators needed for optional tasks (continued)

Administrator	Task	Information
LDAP	Define groups for push-to-client	“Push-to-client considerations” in the *Host Configuration Reference* (SC14-7290)

Pre-configuration considerations

Refer to *Host Configuration Reference* (SC14-7290) for information about Developer for System z itself and how it interacts with your system and with the prerequisite and corequisite products. This information can assist you in creating a setup that supports your current needs and future growth.

Workload management

Unlike traditional z/OS applications, Developer for System z is not a monolithic application that can be identified easily to Workload Manager (WLM). Developer for System z consists of several components that interact to give the client access to the host services and data. Refer to "WLM considerations" in the *Host Configuration Reference* (SC14-7290) to plan your WLM configuration accordingly.

Note: Developer for System z consists of multiple tasks that communicate with each other and the client. These tasks use various timers to detect communication loss with their partner or partners. This implies that timeout issues can arise (due to lack of CPU time during the timeout window) on systems with a heavy CPU load or incorrect Workload Management (WLM) settings for Developer for system z.

Resource usage and system limits

When in use, Developer for System z will use a variable number of system resources like address spaces and z/OS UNIX processes and threads. The availability of these resources is limited by various system definitions. Refer to "Tuning considerations" in the *Host Configuration Reference* (SC14-7290) to estimate the usage of key resources, so you can plan your system configuration accordingly.

Required configuration of requisite products

Consult your MVS system programmer, security administrator and TCP/IP administrator to check if the requisite products and software are installed, tested, and working. Some requisite customization tasks that are easily overlooked are listed here:

- All Developer for System z users must have READ and EXECUTE access to the Java directories.
- Remote (host-based) actions for z/OS UNIX subprojects require that z/OS UNIX version of REXEC or SSH is active on the host.

User ID considerations

The user ID of a Developer for System z user must have (at least) the following attributes:

- TSO access (with a normal region size).

Note: A large region size is required for the user ID that executes the Installation Verification Programs (IVPs), because functions requiring a lot of memory (such as Java) will be executed. You should set the region size to 131072 kilobytes (128 megabytes) or higher.

- An OMVS segment defined to the security system (for example, RACF), both for the user ID and its default group.
  - The HOME field must refer to a home directory allocated for the user (with READ, WRITE and EXECUTE access).
  - The `PROGRAM` field in the OMVS segment should be `/bin/sh` or other valid z/OS UNIX shell, such as `/bin/tcsh`.
  - The `ASSIZEMAX` field should not be set, so that system defaults will be used.
  - The user ID does not require UID 0.

    Example (command **LISTUSER userid NORACF OMVS**):

    ```
    USER=userid

    OMVS INFORMATION
    ----------------
    UID= 0000003200
    HOME= /u/userid
    PROGRAM= /bin/sh
    CPUTIMEMAX= NONE
    ASSIZEMAX= NONE
    FILEPROCMAX= NONE
    PROCUSERMAX= NONE
    THREADSMAX= NONE
    MMAPAREAMAX= NONE
    ```
  - The user ID's default group requires a GID.

    Example (command **LISTGRP group NORACF OMVS**):

    ```
    GROUP group

    OMVS INFORMATION
    ----------------
    GID= 0000003243
    ```
- READ and EXECUTE access to the Developer for System z installation and configuration directories and files, default `/usr/lpp/rdz/*`, `/etc/rdz/*`, and `/var/rdz/*`.
- READ, WRITE, and EXECUTE access to the Developer for System z WORKAREA directory, default `/var/rdz/WORKAREA`, and user log directory, default `/var/rdz/logs`.
- READ access to the Developer for System z installation data sets, default `FEK.SFEK*`.
- READ, WRITE, and EXECUTE access to the `/tmp` directory (or a directory referenced in the `TMPDIR` environment variable).

Server considerations

Developer for System z consists of three permanently active servers, which can be started tasks or user jobs. These servers provide the requested services themselves, or start other servers (as z/OS UNIX threads or user jobs) to provide the service.

- JES Job Monitor (JMON) provides all JES-related services.
- Lock Daemon (LOCKD) provides tracking services for data set locks.
- Remote Systems Explorer (RSE) provides core services such as connecting the client to the host and starting other servers for specific services. RSE consists of two logical entities:

  - RSE daemon (RSED), which manages connection setup and which is responsible for running in single server mode.
  - RSE server, which handles individual client request.

JES Job Monitor (JMON) provides all JES related services.

- The security mechanisms used by JES Job Monitor rely on the data sets it resides in being secure. This implies that only trusted system administrators should be able to update the libraries and configuration files.

Remote Systems Explorer (RSE) is the Developer for System z component that provides core services such as connecting the client to the host.

- Since version 7.5, RSE daemon is no longer an INETD managed process but a started task.
- Since version 7.5, RSE server uses a single server model whereas with previous versions, each client-host connection had a private RSE server.
- Different levels of communication security are supported by RSE:
  - External (client-host) communication can be limited to specified ports. This feature is disabled by default.
  - External (client-host) communication can be encrypted using SSL. This feature is disabled by default.
  - Port Of Entry (POE) checking can be used to allow access only to trusted TCP/IP addresses. This feature is disabled by default.
- RSE also supports multiple client authentication methods:
  - User ID and password
  - User ID and one-time password
  - X.509 certificate
- The security mechanisms used by RSE rely on the file system it resides in being secure. This implies that only trusted system administrators should be able to update the libraries and configuration files.
- The user IDs assigned to the RSED and LOCKD started tasks must have READ, WRITE, and EXECUTE access to their home directory if `/tmp` is not available for WRITE access.

As documented in "TCP/IP ports" in *Host Configuration Reference* (SC14-7290), certain host services, and thus their ports, must be available for the client to connect to, and must be defined to your firewall protecting the host. All other ports used by Developer for System z have host-only traffic. Listed below are the ports needed for external communication in a basic Developer for System z setup.

- RSE daemon for client-host communication setup (using the tcp protocol), default port 4035.
- RSE server for client-host communication (using the tcp protocol). By default, any available port is used, but this can be limited to a specified range.

Configuration method

Developer for System z provides alternative methods to configure the host side of the product. This gives you a choice of the following methods:

- Using the Host Configuration Utility. This ISPF panel application guides you through the required customization steps and selected optional customization steps. For more information, refer to the *Host Configuration Utility* (SC14-7282).
- Using the *Host Configuration Quick Start Guide*. This guides you through the required customization steps. The scope of this guide is limited to a basic setup.

- Using the *Host Configuration Guide*. This guides you through the required customization steps and all optional customization steps. All configurable options are covered in this guide, including some non-default scenarios

Predeployment considerations

Developer for System z supports cloning an installation to a different system, avoiding the need for a SMP/E installation on each system.

The following data sets, directories, and files are mandatory for deployment to other systems. If you copied a file to a different location, then this file must replace its counterpart in the following lists.

Note: The following list does not cover the deployment needs of the pre- and corequisite software.

- FEK.SFEKAUTH(*)
- FEK.SFEKLMOD(*)
- FEK.SFEKLOAD(*)
- FEK.SFEKPROC(*)
- FEK.#CUST.PARMLIB(*)
- FEK.#CUST.PROCLIB(*)
- /usr/lpp/rdz/*
- /etc/rdz/*
- /var/rdz/* (directory structure only)
- optional parts:
  - FEK.SFEKLPA(*)
  - FEK.#CUST.CNTL(*)
  - definitions, data sets, files, and directories resulting from customization jobs in FEK.#CUST.JCL

Note:

- `FEK` and `/usr/lpp/rdz` are the high-level qualifier and path used during the installation of the product. `FEK.#CUST`, `/etc/rdz` and `/var/rdz` are the default locations used during the customization of the product (see "Customization setup" on page 13 for more information).
- You should install Developer for System z in a private file system (HFS or zFS) to easy deploying the z/OS UNIX parts of the product.
- If you cannot use a private file system, you should use an archiving tool such as the z/OS UNIX `tar` command to transport the z/OS UNIX directories from system to system. This to preserve the attributes (such as program control) for the Developer for System z files and directories.

  Refer to *UNIX System Services Command Reference* (SA22-7802) for more information about the following sample commands to archive and restore the Developer for System z installation directory.
  - Archive: `cd /SYS1/usr/lpp/rdz; tar -cSf /u/userid/rdz.tar`
  - Restore: `cd /SYS2/usr/lpp/rdz; tar -xSf /u/userid/rdz.tar`

Client checklist

Users of the Developer for System z client must know the result of certain host customizations, such as TCP/IP port numbers, for the client to work properly. Use these checklists to gather the information needed.

The checklist in Table 5 lists the required results of mandatory customization steps. Table 6 lists the required results of optional customization steps.

Table 5. Client checklist - mandatory parts

Customization	Value
RSE daemon TCP/IP port number (default 4035): See "RSE daemon" on page 21.	

Table 6. Client checklist - optional parts

Customization	Value
Location of the ELAXF* procedures if they are not in a system procedure library (default FEK.#CUST.PROCLIB): See note on JCLLIB in "ELAXF* remote build procedures" on page 23.	
Procedure or step names of the ELAXF* procedures if they were changed: See note on changing them in "ELAXF* remote build procedures" on page 23.	
DB2 stored procedure name (default ELAXMSAM): See information about DB2 stored procedures in "Running multiple instances" in the *Host Configuration Reference* (SC14-7290).	
Location of the DB2 stored procedure if it is not in a system procedure library: See "(Optional) DB2 stored procedure" on page 101.	
Location of the FEKRNPLI Include Preprocessor exec (default FEK.#CUST.CNTL): See "(Optional) Include preprocessor support" on page 104.	
Location of the unit test load modules if not in LINKLIST or STEPLIB of rsed.envvars (default FEK.SFEKLOAD): See "(Optional) xUnit support for Enterprise COBOL and PL/I" on page 105.	
Location of the AZUZUNIT procedure if it is not in a system procedure library (default FEK.#CUST.PROCLIB): See note on JCLLIB in "(Optional) xUnit support for Enterprise COBOL and PL/I" on page 105.	
(corequisite) TN3270 port number for Host Connect Emulator (default 23): See "TCP/IP ports" in *Host Configuration Reference* (SC14-7290).	
(corequisite) REXEC or SSH port number (default 512 or 22, respectively): See "(Optional) z/OS UNIX subprojects" on page 103.	

Table 6. Client checklist - optional parts (continued)

Customization	**Value**
(corequisite) Debug Tool server port number (no default): See “(Optional) DB2 and IMS debug support” on page 108.	
Application Deployment Manager port number (default 5129 for WebService or 5130 for RESTful): See "TCP/IP ports" in *Host Configuration Reference* (SC14-7290).	
Location of the CRA#ASLM JCL for CARMA SCLM RAM data set allocations (default FEK.#CUST.JCL): See note on CRA#ASLM in “SCLM RAM” on page 69.	

Chapter 2. Basic customization

The following customization steps are for a basic Developer for System z setup. Refer to the chapters about the optional components for their customization requirements.

Requirements and checklist

You will need the assistance of a security administrator and a TCP/IP administrator to complete this customization task, which requires the following resources and special customization tasks:

- APF authorized data set
- Various PARMLIB updates
- Various security software updates
- Various TCP/IP ports for internal and client-host communication

In order to verify the installation and to start using Developer for System z at your site, you must perform the following tasks. Unless otherwise indicated, all tasks are mandatory.

1. Create customizable copies of samples and create the work environment for Developer for system z. For details, see "Customization setup."
2. Update z/OS UNIX system limits, start started tasks, define APF authorized and LINKLIST data sets and optionally LPA data sets. For details, see "PARMLIB changes" on page 14.
3. Create started task procedures and compile/link procedures. For details, see "PROCLIB changes" on page 20.
4. Update security definitions. For details, see "Security definitions" on page 25. You must also be aware of and understand how PassTickets are used to establish thread security. See "Using PassTickets" in *Host Configuration Reference* (SC14-7290) for details.
5. Customize Developer for System z configuration files. For details, see:
   - "FEJJCNFG, JES Job Monitor configuration file" on page 25
   - "rsed.envvars, RSE configuration file" on page 30
   - "ISPF.conf, ISPF's TSO/ISPF Client Gateway configuration file" on page 46

Customization setup

Developer for System z comes with several sample configuration files and sample JCL. To avoid overwriting your customizations when applying maintenance, you should copy all these members and z/OS UNIX files to a different location and customize the copy.

Some functions of Developer for System z also require the existence of certain directories in z/OS UNIX, which must be created during the customization of the product. To ease the installation effort, a sample job, FEKSETUP, is provided to create the copies and the required directories.

Note: The *Rational Developer for System z Host Configuration Utility Guide* (SC14-7282) describes the host configuration using the Host Configuration Utility. The FEKSETUP job and the utility do some of the same tasks, with no way of checking to see if those tasks have already been performed. Therefore it is possible to undo changes that have already been made. For this reason, you should not use both methods for a single installation.

Customize and submit sample member FEKSETUP in data set FEK.SFEKSAMP to create customizable copies of configuration files and configuration JCL, and to create required z/OS UNIX directories. The required customization steps are described within the member.

This job performs the following tasks:

- Create FEK.#CUST.PARMLIB and populate it with sample configuration files.
- Create FEK.#CUST.PROCLIB and populate it with sample SYS1.PROCLIB members.
- Create FEK.#CUST.JCL and populate it with sample configuration JCL.
- Create FEK.#CUST.CNTL and populate it with sample server startup scripts.
- Create FEK.#CUST.ASM and populate it with sample assembler source code.
- Create FEK.#CUST.COBOL and populate it with sample COBOL source code.
- Create /etc/rdz/* and populate it with sample configuration files.
- Create /var/rdz/* as work directories for various Developer for System z functions, and populate it with sample files.

Note:

- The configuration steps in this publication use the member/file locations created by the FEKSETUP job, unless noted otherwise. The original samples, which should not be updated, can be found in FEK.SFEKSAMP and /usr/lpp/rdz/samples/.
- Refer to comments in FEK.SFEKSAMP(FEKSETUP) for more details on which sample members are copied to which data set, and for more details on which directories are created, their permission bitmask, and where the various sample files are copied to.
- The comments in FEK.SFEKSAMP(FEKSETUP) also document changes between different versions of Developer for System z to aid in migrating an existing setup.
- If you want to keep all Developer for System z z/OS UNIX files in the same file system (HFS or zFS), but also want the configuration files placed in /etc/rdz, you can use symbolic links to solve this problem. The following sample z/OS UNIX commands create a new directory in the existing file system (/usr/lpp/rdz/cust) and define a symbolic link (/etc/rdz) to it:

```
mkdir /usr/lpp/rdz/cust
ln -s /usr/lpp/rdz/cust /etc/rdz
```

PARMLIB changes

Refer to *MVS Initialization and Tuning Reference* (SA22-7592) for more information about the PARMLIB definitions listed in the next sections. Refer to *MVS System Commands* (SA22-7627) for more information about the sample console commands.

Set z/OS UNIX limits in BPXPRMxx

Remote Systems Explorer (RSE), which provides core services such as connecting the client to the host, is a z/OS UNIX based process. Therefore it is important to set correct values for the z/OS UNIX system limits in BPXPRMxx, based upon the number of concurrently active Developer for System z users and their average workload.

Refer to "Tuning considerations" in the *Host Configuration Reference* (SC14-7290) for more information about different BPXPRMxx defined limits and their impact on Developer for System z.

MAXASSIZE specifies the maximum address space (process) region size. Set MAXASSIZE in SYS1.PARMLIB(BPXPRMxx) to 2G. This is the maximum value allowed. This is a system-wide limit, and thus active for all z/OS UNIX address spaces. If this is not what you want, then you can set the limit also just for Developer for System z in your security software, as described in “Define the Developer for System z started tasks” on page 134.

MAXTHREADS specifies the maximum number of active threads for a single process. Set MAXTHREADS in SYS1.PARMLIB(BPXPRMxx) to 1500 or higher. This is a system-wide limit, and thus active for all z/OS UNIX address spaces. If this is not what you want, then you can set the limit also just for Developer for System z in your security software, as described in “Define the Developer for System z started tasks” on page 134.

MAXTHREADTASKS specifies the maximum number of active MVS tasks for a single process. Set MAXTHREADTASKS in SYS1.PARMLIB(BPXPRMxx) to 1500 or higher. This is a system-wide limit, and thus active for all z/OS UNIX address spaces. If this is not what you want, then you can set the limit also just for Developer for System z in your security software, as described in “Define the Developer for System z started tasks” on page 134.

MAXPROCUSER specifies the maximum number of processes that a single z/OS UNIX user ID can have concurrently active. Set MAXPROCUSER in SYS1.PARMLIB(BPXPRMxx) to 50 or higher. This setting is intended to be a system-wide limit, as it should be active for each client using Developer for System z.

These values can be checked and set dynamically (until the next IPL) with the following console commands:

- DISPLAY OMVS,O
- SETOMVS MAXASSIZE=2G
- SETOMVS MAXTHREADS=1500
- SETOMVS MAXTHREADTASKS=1500
- SETOMVS MAXPROCUSER=50

Note:

- Refer to "Address space size" in the *Host Configuration Reference* (SC14-7290) for more information about other locations where address space sizes can be set or limited.
- The MAXPROCUSER value used above is based upon users having a unique z/OS UNIX user ID (UID). Increase this value if your users share the same UID.
- Ensure that other BPXPRMxx values, such as those for MAXPROCSYS and MAXUIDS, are sufficient to handle the expected amount of concurrently active Developer for System z users. Refer to "Tuning considerations" in the *Host Configuration Reference* (SC14-7290) for more details.

Add started tasks to COMMNDxx

Add start commands for the Developer for System z RSED, LOCKD, and JMON servers to SYS1.PARMLIB(COMMANDxx) to start them automatically at next system IPL.

After the servers are defined and configured, they can be started dynamically (until the next IPL) with the following console commands:

- `S RSED`
- `S LOCKD`
- `S JMON`

Note: The lock daemon should be started before Developer for System z users log on to the RSE daemon. This is so the lock daemon can track the data set lock requests by these users. Therefore you should start the lock daemon at system startup.

LPA definitions in LPALSTxx

The (optional) Common Access Repository Manager (CARMA) service supports different server startup methods for the CARMA server. The CRASTART startup method requires that module `CRASTART` in the `FEK.SFEKLPA` load library is in the Link Pack Area (LPA).

LPA data sets are defined in `SYS1.PARMLIB(LPALSTxx)`.

LPA definitions can be set dynamically (until the next IPL) with the following console command:

- `SETPROG LPA,ADD,DSN=FEK.SFEKLPA`

Note:

- All libraries loaded into LPA are automatically considered to be APF authorized and program controlled. Ensure you have proper security controls in place for these libraries.
- If you choose to not place a library designed for LPA placement in LPA and use LINKLIST or STEPLIB instead, ensure that you define the APF authorization and program control status.

APF authorizations in PROGxx

In order for JES Job Monitor to access JES spool files, module `FEJJMON` in the `FEK.SFEKAUTH` load library and the Language Environment® (LE) runtime libraries (`CEE.SCEERUN*`) must be APF authorized.

In order for the (optional) SCLM Developer Toolkit service to work, module `BWBTSOW` in the `FEK.SFEKAUTH` load library and the REXX runtime library (`REXX.*.SEAGLPA`) must be APF authorized.

In order for ISPF to create the TSO/ISPF Client Gateway, module `ISPZTSO` in `SYS1.LINKLIB` must be APF authorized. The TSO/ISPF Client Gateway is used by Developer for System z's TSO Commands service and SCLM Developer Toolkit.

APF authorizations are defined in `SYS1.PARMLIB(PROGxx)`, if your site followed IBM recommendations.

APF authorizations can be set dynamically (until the next IPL) with the following console commands, where `volser` is the volume on which the data set resides if it is not SMS managed:

- `SETPROG APF,ADD,DSN=FEK.SFEKAUTH,SMS`
- `SETPROG APF,ADD,DSN=CEE.SCEERUN,VOL=volser`
- `SETPROG APF,ADD,DSN=CEE.SCEERUN2,VOL=volser`

- `SETPROG APF,ADD,DSN=REXX.V1R4M0.SEAGLPA,VOL=volser`
- `SETPROG APF,ADD,DSN=SYS1.LINKLIB,VOL=volser`

Note:

- When you use the Alternate Library for REXX product package, the default REXX runtime library name is `REXX.*.SEAGALT`, instead of `REXX.*.SEAGLPA` as used in the preceding sample.
- LPA libraries, such as `REXX.*.SEAGLPA`, are automatically APF authorized when located in LPA, and thus do not require explicit definitions.
- Some of the corequisite products, such as IBM Debug Tool, also require APF authorization. Refer to the related product customization guides for more information about this.

LINKLIST definitions in PROGxx

LINKLIST definitions for Developer for System z can be grouped in three categories:

- Developer for System z load libraries needed for Developer for System z functions. These definitions are described in this section.
- Requisite load libraries needed for Developer for System z functions. These definitions are described in “Requisite LINKLIST and LPA definitions” on page 18.
- Developer for System z load libraries needed by other products. These definitions are described in “LINKLIST definitions for other products” on page 20.

Table 7. Match load modules to functions

Load library	Load modules	Usage	STEPLIB
FEK.SFEKAUTH	BWB*	Chapter 4, “(Optional) SCLM Developer Toolkit,” on page 77	rsed.envvars
	FEJ*	“PROCLIB changes” on page 20 (JES Job Monitor started task)	Started task procedure
FEK.SFEKLMOD	IRZ* and IIRZ*	“(Optional) Diagnostic IRZ messages for generated code” on page 107	CICS, IMS, or MVS batch
FEK.SFEKLOAD	AND*	Chapter 5, “(Optional) Application Deployment Manager,” on page 85	CICS
	AZU* and IAZU*	“(Optional) xUnit support for Enterprise COBOL and PL/I” on page 105	rsed.envvars or MVS batch
	BWB*	Chapter 4, “(Optional) SCLM Developer Toolkit,” on page 77	rsed.envvars
	CRA*	Chapter 3, “(Optional) Common Access Repository Manager (CARMA),” on page 49	CRASUB* or crastart*.conf

Table 7. Match load modules to functions (continued)

Load library	Load modules	Usage	STEPLIB
	ELAX*	"ELAXF* remote build procedures" on page 23 (error feedback and include preprocessor)	ELAXF* procedures
	FEJB*	"(Optional) CICS bidirectional language support" on page 106	CICS
FEK.SFEKLPA	CRA*	Chapter 3, "(Optional) Common Access Repository Manager (CARMA)," on page 49	CRASRV.properties

In order for the listed Developer for System z services to work, all modules documented in Table 7 on page 17 that are related to the service must be made available either through STEPLIB or LINKLST (or LPA). Note that the `SFEKLMOD` library is not used by Developer for System z itself, but by code generated by Developer for System z. See the STEPLIB column in Table 7 on page 17 if you choose to use STEPLIB to learn where the STEPLIB (or DFHRPL for CICS) definition must be made. However, you should be aware of the following things:

- Using STEPLIB in z/OS UNIX has a negative performance impact.
- If one STEPLIB library is APF authorized, then all must be authorized. Libraries lose their APF authorization when they are mixed with non-authorized libraries in STEPLIB.
- Libraries added to the STEPLIB DD in a JCL are not propagated to the z/OS UNIX processes started by the JCL.

LINKLIST data sets are defined in `SYS1.PARMLIB(PROGxx)`, if your site followed IBM recommendations.

The required definitions will look like the following, where `listname` is the name of the LINKLIST set that will be activated, and `volser` is the volume on which the data set resides if it is not cataloged in the master catalog:

- `LNKLST ADD NAME(listname) DSNAME(FEK.SFEKAUTH) VOLUME(volser)`
- `LNKLST ADD NAME(listname) DSNAME(FEK.SFEKLOAD)`

LINKLIST definitions can be created dynamically (until the next IPL) with the following group of console commands, where `listname` is the name of the current LINKLIST set, and `volser` is the volume on which the data set resides if it is not cataloged in the master catalog:

1. `LNKLST DEFINE,NAME=LLTMP,COPYFROM=CURRENT`
2. `LNKLST ADD NAME=LLTMP,DSN=FEK.SFEKAUTH,VOL=volser`
3. `LNKLST ADD NAME=LLTMP,DSN=FEK.SFEKLOAD`
4. `LNKLST ACTIVATE,NAME=LLTMP`

Requisite LINKLIST and LPA definitions

Remote Systems Explorer (RSE) is a z/OS UNIX process that requires access to MVS load libraries. The following (prerequisite) libraries must be made available, either through STEPLIB or LINKLIST/LPALIB:

- System load library

  - SYS1.LINKLIB
- Language Environment runtime
  - CEE.SCEERUN
  - CEE.SCEERUN2
- C++'s DLL class library
  - CBC.SCLBDLL
- ISPF's TSO/ISPF Client Gateway
  - ISP.SISPLOAD
  - ISP.SISPLPA

The following additional libraries must be made available, either through STEPLIB or LINKLIST/LPALIB, to support the use of optional services. This list does not include data sets that are specific to a product that Developer for System z interacts with, such as IBM Debug Tool for z/OS:

- REXX runtime library (for SCLM Developer Toolkit)
  - REXX.*.SEAGLPA
- System load library (for SSL encryption)
  - SYS1.SIEALNKE
- System load library (for Enterprise COBOL and PL/I unit test)
  - SYS1.CSSLIB
  - SYS1.SIXMLOD1

Note:

- When you use the Alternate Library for REXX product package, the default REXX runtime library name is REXX.*.SEAGALT, instead of REXX.*.SEAGLPA as used in the preceding sample.
- All libraries loaded into LPA are automatically considered to be APF authorized and program controlled. Ensure you have proper security controls in place for these libraries.
- Libraries that are designed for LPA placement, such as REXX.*.SEAGLPA, might require additional program control or APF authorizations if they are accessed through LINKLIST or STEPLIB.
- Some of the corequisite products, such as IBM Debug Tool, also require STEPLIB or LINKLIST/LPALIB definitions. Refer to the related product customization guides for more information about this.

LINKLIST data sets are defined in SYS1.PARMLIB(PROGxx), if your site followed IBM recommendations. LPA data sets are defined in SYS1.PARMLIB(LPALSTxx).

If you opt to use STEPLIB, you must define the libraries not available through LINKLIST/LPALIB in the STEPLIB directive of rsed.envvars, the RSE configuration file. Be aware, however, that:

- Using STEPLIB in z/OS UNIX has a negative performance impact.
- If one STEPLIB library is APF authorized, then all must be authorized. Libraries lose their APF authorization when they are mixed with non-authorized libraries in STEPLIB.
- Libraries added to the STEPLIB DD in a JCL are not propagated to the z/OS UNIX processes started by the JCL.

LINKLIST definitions for other products

The Developer for System z client has a code generation component called Enterprise Service Tools. In order for the generated code to issue diagnostic error messages, all IRZM* and IIRZ* modules in the FEK.SFEKLMOD load library must be made available either through STEPLIB or LINKLIST.

LINKLIST data sets are defined in SYS1.PARMLIB(PROGxx), if your site followed IBM recommendations.

If you opt to use STEPLIB, you must define the libraries not available through LINKLIST in the STEPLIB directive of the task that executes the code (IMS or batch job). However, be aware of the following:

- If one STEPLIB library is APF authorized, then all must be authorized. Libraries loose their APF authorization when they are mixed with non-authorized libraries in STEPLIB.

PROCLIB changes

The started task and remote build procedures listed in the following sections must reside in a system procedure library defined to your JES subsystem. In the instructions found in the following sections,, the IBM default procedure library, SYS1.PROCLIB, is used.

JES Job Monitor

Customize the sample started task member FEK.#CUST.PROCLIB(JMON), as described within the member, and copy it to SYS1.PROCLIB. As shown in the following code sample, you have to provide this information:

- The high-level qualifier of the (authorized) load library, default FEK
- The JES Job Monitor configuration file, default FEK.#CUST.PARMLIB(FEJJCNFG)

```
//*
//* JES JOB MONITOR
//*
//JMON     PROC PRM=,            * PRM='-TV' TO START TRACING
//            LEPRM='RPTOPTS(ON)',
//            HLQ=FEK,
//            CFG=FEK.#CUST.PARMLIB(FEJJCNFG)
//*
//JMON     EXEC PGM=FEJJMON,REGION=0M,TIME=NOLIMIT,
//            PARM=('&LEPRM,ENVAR("_CEE_ENVFILE_S=DD:ENVIRON")/&PRM')
//STEPLIB  DD DISP=SHR,DSN=&HLQ..SFEKAUTH
//ENVIRON  DD DISP=SHR,DSN=&CFG
//SYSPRINT DD SYSOUT=*
//SYSOUT   DD SYSOUT=*
//         PEND
//*
```

Figure 1. JMON - JES Job Monitor started task

Note:

- Refer to Chapter 10, “Operator commands,” on page 169 for more information about the startup parameters.
- The sample JCL is initially shipped as FEK.SFEKSAMP(FEJJJCL) and is renamed to FEK.#CUST.PROCLIB(JMON) in “Customization setup” on page 13.
- Tracing can also be controlled by console commands, as described in Chapter 10, “Operator commands,” on page 169.

- See "WLM considerations" in the *Host Configuration Reference* (SC14-7290) for the advised Workload Manager (WLM) goals for this task.

RSE daemon

Customize the sample started task member FEK.#CUST.PROCLIB(RSED), as described within the member, and copy it to SYS1.PROCLIB. As shown in the following code sample, you have to provide this information:

- The home directory where Developer for System z is installed, default /usr/lpp/rdz.
- The location of the configuration files, default /etc/rdz

```
//*
//* RSE DAEMON
//*
//RSED     PROC IVP=,                   * 'IVP' to do an IVP test
//            PORT=,
//            CNFG='/etc/rdz',
//            HOME='/usr/lpp/rdz'
//*
//RSED     EXEC PGM=BPXBATSL,REGION=0M,TIME=NOLIMIT,
// PARM='PGM &HOME./bin/rsed.sh &IVP -C&CNFG -P&PORT'
//STDOUT   DD SYSOUT=*
//STDERR   DD SYSOUT=*
//         PEND
//*
```

Figure 2. RSED - RSE daemon started task

Note:

- Refer to Chapter 10, “Operator commands,” on page 169 for more information about the startup parameters.
- The sample JCL is initially shipped as FEK.SFEKSAMP(FEKRSED) and is renamed to FEK.#CUST.PROCLIB(RSED) in “Customization setup” on page 13.
- Limit the length of the job name to 7 characters or less. The **modify** and **stop** operator commands will fail with message "IEE342I MODIFY   REJECTED-TASK BUSY" if an 8 character name is used. This behavior is caused by the z/OS UNIX design for child processes.
- See "WLM considerations" in the *Host Configuration Reference* (SC14-7290) for the advised Workload Manager (WLM) goals for this task and the child processes it creates. The child processes have the same name as the parent task, RSED, appended with a random 1-digit number, for example RSED8.

Lock daemon

Customize the sample started task member FEK.#CUST.PROCLIB(LOCKD), as described within the member, and copy it to SYS1.PROCLIB. As shown in the following code sample, you have to provide this information:

- The home directory where Developer for System z is installed, default /usr/lpp/rdz.
- The location of the configuration files, default /etc/rdz.

```
//*
//* LOCK DAEMON
//*
//LOCKD    PROC LOG=,
//            CNFG='/etc/rdz',
//            HOME='/usr/lpp/rdz'
//*
//LOCKD    EXEC PGM=BPXBATSL,REGION=0M,TIME=NOLIMIT,
//            PARM='PGM &HOME./bin/lockd.sh -C&CNFG -L&LOG'
//STDOUT   DD SYSOUT=*
//STDERR   DD SYSOUT=*
//         PEND
//*
```

Figure 3. LOCKD - Lock daemon started task

Note:

- Refer to Chapter 10, “Operator commands,” on page 169 for more information about the startup parameters.
- The sample JCL is initially shipped as `FEK.SFEKSAMP(FEKLOCKD)` and is renamed to `FEK.#CUST.PROCLIB(LOCKD)` in “Customization setup” on page 13.
- See "WLM considerations" in the *Host Configuration Reference* (SC14-7290) for the advised Workload Manager (WLM) goals for this task.

JCL limitations for the PARM variable

The maximum length for the PARM variable is 100 characters, which might cause problems if you use custom directory names. To bypass this problem, you can either:

- Use default values

  The `rsed.sh` and `lockd.sh` startup scripts can be invoked without arguments, in which case the default argument values will be used.
- Use symbolic links

  Symbolic links can be used as shorthand for a long directory name. The following sample z/OS UNIX command defines a symbolic link (`/usr/lpp/rdz`) to another directory (`/long/directory/name/usr/lpp/rdz`).

  ```
  ln -s /long/directory/name/usr/lpp/rdz /usr/lpp/rdz
  ```
- Use STDIN

  When the PARM field is empty, **BPXBATCH** will start a z/OS UNIX shell and execute the shell script that is provided by STDIN. Note that STDIN must be a z/OS UNIX file (allocated as ORDONLY) and that using STDIN disables the usage of PROC variables like `TMPDIR`. Also note that the shell will execute the shell logon scripts `/etc/profile` and `$HOME/.profile`.

  To use this method, you must first update the startup JCL to match something like the following sample:

```
//*
//* RSE DAEMON - USING STDIN
//*
//RSED     PROC CNFG='/etc/rdz'
//*
//RSE      EXEC PGM=BPXBATCH,REGION=0M,TIME=NOLIMIT
//STDOUT   DD SYSOUT=*
//STDERR   DD SYSOUT=*
//STDIN    DD PATHOPTS=(ORDONLY),PATH='&CNFG./rsed.stdin.sh'
//         PEND
//*
```

Figure 4. RSED - alternate RSE daemon startup

Second, you must create the shell script (`/etc/rdz/rsed.stdin.sh` in this example) that will start the RSE daemon. The content of this script will look like the following sample:

```
CNFG=/etc/rdz
PORT=
IVP=
/long/directory/name/usr/lpp/rdz/bin/rsed.sh $IVP -C$CNFG –P$PORT –T$TMPDIR
```

Figure 5. rsed.stdin.sh - alternate RSE daemon startup

Note: When using this method, the RSE daemon itself will not be active in the RSED address space. The RSE daemon will be active in a RSEDx address space. This because z/OS UNIX runs child processes (such as starting a shell) in separate address spaces. Adding a STDENV DD with a _BPX_SHAREAS=YES directive does not change this, as it is interpreted too late. This side effect severely complicates using Developer for System z operator commands.

ELAXF* remote build procedures

Developer for System z provides sample JCL procedures that can be used for the JCL generation, remote project builds, and remote syntax check features of CICS BMS maps, IMS MFS screens, COBOL, PL/I, Assembler, and C/C++ programs. These procedures allow installations to apply their own standards, and ensure that developers use the same procedures with the same compiler options and compiler levels.

The sample procedures and their function are listed in Table 8.

Table 8. Sample ELAXF procedures*

Member	**Purpose**
ELAXFADT	Sample procedure for assembling and debugging High Level assembler programs.
ELAXFASM	Sample procedure for assembling High Level assembler programs.
ELAXFBMS	Sample procedure for creating CICS BMS object and corresponding copy, dsect, or include member.
ELAXFCOC	Sample procedure for doing COBOL Compiles, Integrated CICS translate and integrated DB2 translate.
ELAXFCOP	Sample procedure for doing DB2 preprocess of EXEC SQL statements embedded in COBOL programs.
ELAXFCOT	Sample procedure for doing CICS translation for EXEC CICS statements embedded in COBOL programs.
ELAXFCPC	Sample procedure for doing C compiles.

Table 8. Sample ELAXF procedures (continued)*

Member	Purpose
ELAXFCPP	Sample procedure for doing C++ compiles.
ELAXFCP1	Sample procedure for COBOL compiles with SCM preprocessor statements (-INC and ++INCLUDE).
ELAXFDCL	Sample procedure for running a program in TSO mode.
ELAXFGO	Sample procedure for the GO step.
ELAXFLNK	Sample procedure for linking C/C++, COBOL. PLI and High Level Assembler programs.
ELAXFMFS	Sample procedure for creating IMS MFS screens.
ELAXFPLP	Sample procedure for doing DB2 preprocess of EXEC SQL statements embedded in PLI programs.
ELAXFPLT	Sample procedure for doing CICS translation of EXEC CICS statements embedded in PLI programs.
ELAXFPL1	Sample procedure for doing PL/I compiles, integrated CICS translate and integrated DB2 translate.
ELAXFPP1	Sample procedure for PL/I compiles with SCM preprocessor statements (-INC and ++INCLUDE).
ELAXFTSO	Sample procedure for running/debugging generated DB2 code in TSO mode.
ELAXFUOP	Sample procedure for generating the UOPT step when building programs that run in CICS or IMS subsystems.

The names of the procedures and the names of the steps in the procedures match the default properties that are shipped with the Developer for System z client. If you decide to change the name of a procedure or the name of a step in a procedure, the corresponding properties file on all the clients should also be updated. You should not change the procedure and step names.

Customize the sample build procedure members, `FEK.#CUST.PROCLIB(ELAXF*)`, as described within the members, and copy them to `SYS1.PROCLIB`. You have to provide the correct high-level qualifiers for different product libraries, as described in Table 9.

Table 9. ELAXF high-level qualifier checklist*

Product	Default HLQ	Value
Developer for System z	FEK	
CICS	CICSTS32.CICS	
DB2	DSN910	
IMS	IMS	
COBOL	IGY.V4R1M0	
PL/I	IBMZ.V3R8M0	
C/C++	CBC	
LE	CEE	
system LINKLIB	SYS1	
system MACLIB	SYS1	

If the ELAXF* procedures cannot be copied into a system procedure library, ask the Developer for System z users to add a JCLLIB card (right after the JOB card) to the job properties on the client.

```
//MYJOB    JOB <job parameters>
//PROCS    JCLLIB ORDER=(FEK.#CUST.PROCLIB)
```

Security definitions

Customize and submit sample member FEKRACF to create the security definitions for Developer for System z. The user submitting this job must have security administrator privileges, such as being RACF SPECIAL.

FEKRACF is located in FEK.#CUST.JCL, unless you specified a different location when you customized and submitted job FEK.SFEKSAMP(FEKSETUP). See “Customization setup” on page 13 for more details.

Note:

- For those sites that use CA ACF2™ for z/OS, please refer to your product page on the CA support site (https://support.ca.com) and check for the related Developer for System z Knowledge Document, TEC492389. This Knowledge Document has details on the security commands necessary to properly configure Developer for System z.
- For those sites that use CA Top Secret® for z/OS, please refer to your product page on the CA support site (https://support.ca.com) and check for the related Developer for System z Knowledge Document, TEC492091. This Knowledge Document has details on the security commands necessary to properly configure Developer for System z.

The following list of security-related definitions for Developer for System z are discussed in detail in Chapter 8, “Security definitions,” on page 127.

- Activate security settings and classes
- Define an OMVS segment for Developer for System z users
- Define data set profiles
- Define the JMON, RSED, and LOCKD started tasks
- Define JES command security
- Define RSE as a secure z/OS UNIX server
- Define MVS program controlled libraries for RSE
- Define application security for RSE
- Define PassTicket support for RSE
- Define z/OS UNIX program controlled files for RSE

Note: The sample FEKRACF job holds more than just RACF commands. The last step of the security definitions consists of making a z/OS UNIX file program controlled. Depending on the policies at your site, this might be a task for the system programmer and not the security administrator.

Attention: The client connection request will fail if PassTickets are not set up correctly.

FEJJCNFG, JES Job Monitor configuration file

JES Job Monitor (JMON) provides all JES-related services. The behavior of JES Job Monitor can be controlled with the definitions in FEJJCNFG.

FEJJCNFG is located in FEK.#CUST.PARMLIB, unless you specified a different location when you customized and submitted job FEK.SFEKSAMP(FEKSETUP). See "Customization setup" on page 13 for more details.

Customize the sample JES Job Monitor configuration member FEJJCNFG, as shown in the following sample. Comment lines start with a pound sign (#), when using a US code page. Data lines can only have a directive and its assigned value. Comments are not allowed on the same line.

Note: The JMON started task must be restarted to pick up any changes you make.

```
SERV_PORT=6715
TZ=EST5EDT
#_BPXK_SETIBMOPT_TRANSPORT=TCPIP
#APPLID=FEKAPPL
#AUTHMETHOD=SAF
#CODEPAGE=UTF-8
#CONCHAR=$
#CONSOLE_NAME=JMON
#GEN_CONSOLE_NAME=OFF
#HOST_CODEPAGE=IBM-1047
#LIMIT_COMMANDS=NOLIMIT
#LIMIT_CONSOLE=LIMITED
#LIMIT_VIEW=USERID
#LISTEN_QUEUE_LENGTH=5
#MAX_DATASETS=32
#MAX_THREADS=200
#TIMEOUT=3600
#TIMEOUT_INTERVAL=1200
#TRACE_STORAGE=OFF
#SEARCHALL=OFF
#SUBMIT_TIMEOUT=30
#SUBMITMETHOD=TSO
#TSO_TEMPLATE=FEK.#CUST.CNTL(FEJTSO)
```

Figure 6. FEJJCNFG, JES Job Monitor configuration file

SERV_PORT

The port number for JES Job Monitor host server. The default port is 6715. The port can be changed if desired.

Note:

- This value must match the port number set for JES Job Monitor in the rsed.envvars configuration file. If these values differ, RSE cannot connect the client to JES Job Monitor. Refer to "rsed.envvars, RSE configuration file" on page 30 to learn how to define the variable for RSE.
- Before selecting a port, verify that the port is available on your system with the TSO commands **NETSTAT** and **NETSTAT PORTL**.
- When using a version 7.1 or higher client, all communication on this port is confined to your z/OS host machine.

TZ Time zone selector. The default is EST5EDT. The default time zone is UTC +5 hours (Eastern Standard Time (EST) Eastern Daylight Savings Time (EDT)). Change this to represent your time zone. Additional information can be found in the *UNIX System Services Command Reference* (SA22-7802).

The following definitions are optional. If omitted, default values will be used as specified below:

_BPXK_SETIBMOPT_TRANSPORT

Specifies the name of the TCPIP stack to be used. The default is TCPIP. Uncomment and change to the requested TCPIP stack name, as defined in the TCPIPJOBNAME statement in the related TCPIP.DATA.

Note:

- Coding a SYSTCPD DD statement in the server JCL does not set the requested stack affinity.
- When this directive is not active, JES Job Monitor binds to every available stack on the system (BIND INADDRANY).

APPLID

Specifies the application identifier used for identifying JES Job Monitor to your security software. The default is FEKAPPL. Uncomment and change to the desired application ID.

Note: This value must match the application ID set for RSE in the rsed.envvars configuration file. If these values differ, RSE cannot connect the client to JES Job Monitor. Refer to "rsed.envvars, RSE configuration file" on page 30 to learn how to define the variable for RSE.

AUTHMETHOD

The default is SAF, which means that the System Authorization Facility (SAF) security interface is used. Do not change unless directed to do so by the IBM support center.

CODEPAGE

The workstation codepage. The default is UTF-8. The workstation codepage is set to UTF-8 and generally should not be changed. You might need to uncomment the directive and change UTF-8 to match the workstation's codepage if you have difficulty with NLS characters, such as the currency symbol.

CONCHAR

Specifies the JES console command character. CONCHAR defaults to CONCHAR=$ for JES2, or CONCHAR=* for JES3. Uncomment and change to the requested command character.

CONSOLE_NAME

Specifies the name of the EMCS console used for issuing commands against jobs (Hold, Release, Cancel, and Purge). The default is JMON. Uncomment and change to the desired console name, using the following guidelines.

- CONSOLE_NAME must be either a console name consisting of 2 to 8 alphanumeric characters, or '&SYSUID' (without quotes).
- If a console name is specified, a single console by that name is used for all users. If the console by that name happens to be in use, then the command issued by the client will fail.
- If &SYSUID is specified, the client user ID is used as the console name. Thus a different console is used for each user. If the console by that name happens to be in use (for example, the user is using the SDSF ULOG), then the command issued by the client might fail, depending on the GEN_CONSOLE_NAME setting.

No matter which console name is used, the user ID of the client requesting the command is used as the LU of the console, leaving a trace in syslog messages IEA630I and IEA631.

```
IEA630I OPERATOR console NOW ACTIVE,   SYSTEM=sysid, LU=id
IEA631I OPERATOR console NOW INACTIVE, SYSTEM=sysid, LU=id
```

GEN_CONSOLE_NAME

Enables or disables automatic generating of alternative console names. The default is OFF. Uncomment and change to ON to enable alternative console names.

This directive is only used when CONSOLE_NAME equals &SYSUID and the user ID is not available as console name.

If GEN_CONSOLE_NAME=ON, an alternative console name is generated by appending a single numeric digit to the user ID. The digits 0 through 9 are attempted. If no available console is found, the command issued by the client fails.

If GEN_CONSOLE_NAME=OFF, the command issued by the client fails.

Note: The only valid settings are ON and OFF.

HOST_CODEPAGE

The host codepage. The default is IBM-1047. Uncomment and change to match your host codepage.

From version 7.6.1 on, Developer for System z clients ignore the HOST_CODEPAGE value specified here and use the codepage specified locally in the properties of the "MVS Files" subsystem.

Note: Even for recent clients, JES Job Monitor will use the host codepage specified in HOST_CODEPAGE during initial client communication setup.

LIMIT_COMMANDS

Defines against which jobs the user can issue selected JES commands (Show JCL, Hold, Release, Cancel, and Purge). The default (LIMIT_COMMANDS=USERID) limits the commands to jobs owned by the user. Uncomment this directive and specify LIMITED or NOLIMIT to allow the user to issue commands against all spool files, if permitted by your security product.

Table 10. LIMIT_COMMANDS command permission matrix

	Job owner	
LIMIT_COMMANDS	**User**	**Other**
USERID (default)	Allowed	Not allowed
LIMITED	Allowed	Allowed only if explicitly permitted by security profiles
NOLIMIT	Allowed	Allowed if permitted by security profiles or when the JESSPOOL class is not active

Note: The only valid settings are USERID, LIMITED, and NOLIMIT.

| **LIMIT_CONSOLE**

| Defines how much authority is granted to the console used to execute
| supported JES commands (Hold, Release, Cancel, and Purge). The default
| (LIMIT_CONSOLE=LIMITED) limits authority to commands protected by a

security profile in the OPERCMDS class. Uncomment this directive and specify NOLIMIT to allow execution of supported JES commands that are not protected by a security profile.

Note that when a security profile exists for a command, the user must have sufficient permission to execute the command, regardless of the LIMIT_CONSOLE setting. The only valid settings are LIMITED and NOLIMIT.

LIMIT_VIEW
Defines what output the user can view. The default (LIMIT_VIEW=NOLIMIT) allows the user to view all JES output, if permitted by your security product. Uncomment this directive and specify USERID to limit the view to output owned by the user.

Note: The only valid settings are USERID and NOLIMIT.

LISTEN_QUEUE_LENGTH
The TCP/IP listen queue length. The default is 5. Do not change unless directed to do so by the IBM support center.

MAX_DATASETS
The maximum number of spooled output data sets that JES Job Monitor will return to the client (for example, SYSOUT, SYSPRINT, SYS00001, and so on). The default is 32. The maximum value is 2147483647.

MAX_THREADS
Maximum number of users that can be using one JES Job Monitor at a time. The default is 200. The maximum value is 2147483647. Increasing this number might require you to increase the size of the JES Job Monitor address space.

TIMEOUT
The length of time, in seconds, before a thread is killed due to lack of interaction with the client. The default is 3600 (1 hour). The maximum value is 2147483647. TIMEOUT=0 disables the function.

TIMEOUT_INTERVAL
The number of seconds between timeout checks. The default is 1200. The maximum value is 2147483647.

TRACE_STORAGE
Enable storage tracing. The default is OFF. Uncomment this directive and specify ON to write a storage report to DD SYSOUT after each command. The only valid values are ON and OFF. Use only when directed by the IBM support center.

SEARCHALL
Collect APPC and z/OS UNIX output that matches the JES Job Monitor filter, for example output written to SYSOUT by a Developer for System z CARMA server started using the CRASTART method. The default is OFF. Uncomment this directive and specify ON to collect the additional spool files. The only valid values are ON and OFF.

SUBMIT_TIMEOUT
The number of seconds that Developer for System z will wait for the completion of the TSO_TEMPLATE job. The default is 30. The maximum value is 2147483647. Note: SUBMIT_TIMEOUT has no effect unless SUBMITMETHOD=TSO is also specified.

SUBMITMETHOD=TSO
Submit jobs through TSO. The default (SUBMITMETHOD=JES) submits jobs

directly into JES. Uncomment this directive and specify TSO to submit the job through TSO **SUBMIT** command. This method allows TSO exits to be invoked; however, it has a performance drawback and for that reason it is not recommended.

Note:

- The only valid settings are TSO and JES.
- If SUBMITMETHOD=TSO is specified, then TSO_TEMPLATE must also be defined.

TSO_TEMPLATE

Wrapper JCL for submitting the job through TSO. The default value is FEK.#CUST.CNTL(FEJTSO). This statement refers to the fully qualified member name of the JCL to be used as a wrapper for the TSO **SUBMIT** command. See the SUBMITMETHOD statement for more information.

Note:

- A sample wrapper job is provided in FEK.#CUST.CNTL(FEJTSO). Refer to this member for more information about the customization needed.
- TSO_TEMPLATE has no effect unless SUBMITMETHOD=TSO is also specified.

rsed.envvars, RSE configuration file

The RSE lock daemon and the RSE server processes (RSE daemon, RSE thread pool, and RSE server) use the definitions in rsed.envvars. Optional Developer for System z and third-party services can use this configuration file also to define environment variables for their use.

Remote Systems Explorer (RSE) provides core services such as connecting the client to the host and starting other servers for specific services. Lock daemon provides tracking services for data set locks.

rsed.envvars is located in /etc/rdz/, unless you specified a different location when you customized and submitted job FEK.SFEKSAMP(FEKSETUP). See "Customization setup" on page 13 for more details. You can edit the file with the TSO **OEDIT** command.

See the following sample rsed.envvars file, which must be customized to match your system environment. Comment lines start with a pound sign (#), when using a US code page. Data lines can only have a directive and its assigned value, comments are not allowed on the same line. Line continuations and spaces around the equal sign (=) are not supported.

Note: The RSED and LOCKD started tasks must be restarted to pick up any changes you make.

```
  #=============================================================
  # (1) required definitions
| JAVA_HOME=/usr/lpp/java/J6.0
  RSE_HOME=/usr/lpp/rdz
  _RSE_RSED_PORT=4035
  _RSE_LOCKD_PORT=4036
  _RSE_JMON_PORT=6715
  _RSE_HOST_CODEPAGE=IBM-1047
  TZ=EST5EDT
  LANG=C
  PATH=/bin:/usr/sbin
  _CEE_DMPTARG=/tmp
  STEPLIB=NONE
  #STEPLIB=$STEPLIB:CEE.SCEERUN:CEE.SCEERUN2:CBC.SCLBDLL
  _RSE_JAVAOPTS=""
  _RSE_JAVAOPTS="$_RSE_JAVAOPTS -Xms1m -Xmx256m"
  _RSE_JAVAOPTS="$_RSE_JAVAOPTS -Ddaemon.log=/var/rdz/logs"
  _RSE_JAVAOPTS="$_RSE_JAVAOPTS -Duser.log=/var/rdz/logs"
  _RSE_JAVAOPTS="$_RSE_JAVAOPTS -DDSTORE_LOG_DIRECTORY="
  #_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dmaximum.clients=60"
  #_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dmaximum.threads=1000"
  #_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dminimum.threadpool.process=1"
  #_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dmaximum.threadpool.process=100"
  #_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dipv6=true"
  #_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dkeep.last.log=true"
  #_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Denable.standard.log=true"
  #_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Denable.port.of.entry=true"
  #_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Denable.certificate.mapping=false"
  #_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Denable.automount=true"
  #_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Denable.audit.log=true"
  #_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Daudit.cycle=30"
  #_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Daudit.retention.period=0"
  #_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Daudit.log.mode=RW.R."
| #_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Daudit.action=<user_exit>
| #_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Daudit.action.id=<userid>
| #_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dlogon.action=<user_exit>
| #_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dlogon.action.id=<userid>
  #_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Ddeny.nonzero.port=true"
  #_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dsingle.logon=false"
  #_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dprocess.cleanup.interval=0"
| #_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dreject.logon.threshold=1000000"
| #_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dinclude.c=/etc/rdz/include.conf"
| #_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dinclude.cpp=/etc/rdz/include.conf"
  #_RSE_JAVAOPTS="$_RSE_JAVAOPTS -DCPP_CLEANUP_INTERVAL=60000"
| #_RSE_JAVAOPTS="$_RSE_JAVAOPTS -DRIS_BUFFER=8"
  #_RSE_JAVAOPTS="$_RSE_JAVAOPTS -DAPPLID=FEKAPPL"
  #_RSE_JAVAOPTS="$_RSE_JAVAOPTS -DDENY_PASSWORD_SAVE=true"
  #_RSE_JAVAOPTS="$_RSE_JAVAOPTS -DHIDE_ZOS_UNIX=true"
  #_RSE_JAVAOPTS="$_RSE_JAVAOPTS -DDISABLE_DELETE_IN_SUBPROJECT=true"
  #_RSE_JAVAOPTS="$_RSE_JAVAOPTS -DDSTORE_IDLE_SHUTDOWN_TIMEOUT=3600000"
  #_RSE_JAVAOPTS="$_RSE_JAVAOPTS -DDSTORE_TCP_NO_DELAY=true"
  #_RSE_JAVAOPTS="$_RSE_JAVAOPTS -DDSTORE_TRACING_ON=true"
  #_RSE_JAVAOPTS="$_RSE_JAVAOPTS -DDSTORE_MEMLOGGING_ON=true"
  #=============================================================
  # (2) required definitions for TSO/ISPF Client Gateway
| CGI_ISPHOME=/usr/lpp/ispf
| CGI_ISPCONF=/etc/rdz
| CGI_ISPWORK=/var/rdz
| #STEPLIB=$STEPLIB:ISP.SISPLOAD:ISP.SISPLPA:SYS1.LINKLIB
| _RSE_ISPF_OPTS=""
| #_RSE_ISPF_OPTS="$_RSE_ISPF_OPTS&ISPPROF=&SYSUID..ISPPROF"
| #CGI_ISPPREF=&SYSPREF..ISPF.VCMISPF
  #=============================================================
  # (3) required definitions for SCLM Developer Toolkit
  _SCLMDT_CONF_HOME=/var/rdz/sclmdt
  #STEPLIB=$STEPLIB:FEK.SFEKAUTH:FEK.SFEKLOAD
  #_SCLMDT_TRANTABLE=FEK.#CUST.LSTRANS.FILE
  #ANT_HOME=/usr/lpp/Apache/Ant/apache-ant-1.7.1
  #=============================================================
```

Figure 7. rsed.envvars - RSE configuration file

```
# (4) optional definitions
#_RSE_PORTRANGE=8108-8118
#_BPXK_SETIBMOPT_TRANSPORT=TCPIP
#TMPDIR=/tmp
#_RSE_FEK_SAF_CLASS=FACILITY
#_RSE_LDAP_SERVER=ldap_server_url
#_RSE_LDAP_PORT=389
#_RSE_LDAP_PTC_GROUP_SUFFIX="o=PTC,c=DeveloperForZ"
#GSK_CRL_SECURITY_LEVEL=HIGH
#GSK_LDAP_SERVER=ldap_server_url
#GSK_LDAP_PORT=ldap_server_port
#GSK_LDAP_USER=ldap_userid
#GSK_LDAP_PASSWORD=ldap_server_password
#STEPLIB=$STEPLIB:FEK.SFEKLOAD:SYS1.CSSLIB:SYS1.SIXMLOD1
#=============================================================
# (5) do not change unless directed by IBM support center
_RSE_SAF_CLASS=/usr/include/java_classes/IRRRacf.jar
_CEE_RUNOPTS="ALL31(ON) HEAP(32M,32K,ANYWHERE,KEEP,,) TRAP(ON)"
_BPX_SHAREAS=YES
_BPX_SPAWN_SCRIPT=YES
_EDC_ADD_ERRNO2=1
JAVA_PROPAGATE=NO
RSE_LIB=$RSE_HOME/lib
PATH=.:$JAVA_HOME/bin:$RSE_HOME/bin:$CGI_ISPHOME/bin:$PATH
LIBPATH=$JAVA_HOME/bin:$JAVA_HOME/bin/classic:$RSE_LIB:$RSE_LIB/icuc
LIBPATH=.:/usr/lib:$LIBPATH
CLASSPATH=$RSE_LIB:$RSE_LIB/dstore_core.jar:$RSE_LIB/clientserver.jar
CLASSPATH=$CLASSPATH:$RSE_LIB/dstore_extra_server.jar
CLASSPATH=$CLASSPATH:$RSE_LIB/zosserver.jar
CLASSPATH=$CLASSPATH:$RSE_LIB/dstore_miners.jar
CLASSPATH=$CLASSPATH:$RSE_LIB/universalminers.jar:$RSE_LIB/mvsminers.jar
CLASSPATH=$CLASSPATH:$RSE_LIB/carma.jar:$RSE_LIB/luceneminer.jar
CLASSPATH=$CLASSPATH:$RSE_LIB/mvsluceneminer.jar:$RSE_LIB/cdzminer.jar
CLASSPATH=$CLASSPATH:$RSE_LIB/mvscdzminer.jar:$RSE_LIB/jesminers.jar
CLASSPATH=$CLASSPATH:$RSE_LIB/FAMiner.jar
CLASSPATH=$CLASSPATH:$RSE_LIB/mvsutil.jar:$RSE_LIB/jesutils.jar
CLASSPATH=$CLASSPATH:$RSE_LIB/lucene-core-2.3.2.jar
CLASSPATH=$CLASSPATH:$RSE_LIB/cdtparser.jar:$RSE_LIB/wdzBidi.jar
CLASSPATH=$CLASSPATH:$_RSE_SAF_CLASS
CLASSPATH=.:$CLASSPATH
_RSE_PTC=$_RSE_LDAP_PTC_GROUP_SUFFIX
_RSE_ISPF_OPTS="&SESSION=SPAWN$_RSE_ISPF_OPTS"
_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dldap.server.address=$_RSE_LDAP_SERVER"
_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dldap.server.port=$_RSE_LDAP_PORT"
_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dldap.ptc.group.name.suffix=$_RSE_PTC"
_RSE_JAVAOPTS="$_RSE_JAVAOPTS -DISPF_OPTS='$_RSE_ISPF_OPTS'"
_RSE_JAVAOPTS="$_RSE_JAVAOPTS -DA_PLUGIN_PATH=$RSE_LIB"
_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Xbootclasspath/p:$RSE_LIB/bidiTools.jar"
_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dfile.encoding=$_RSE_HOST_CODEPAGE"
_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dconsole.encoding=$_RSE_HOST_CODEPAGE"
_RSE_JAVAOPTS="$_RSE_JAVAOPTS -DDSTORE_SPIRIT_ON=true"
_RSE_JAVAOPTS="$_RSE_JAVAOPTS -DSPIRIT_EXPIRY_TIME=6"
_RSE_JAVAOPTS="$_RSE_JAVAOPTS -DSPIRIT_INTERVAL_TIME=6"
_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dcom.ibm.cacheLocalHost=true"
_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Duser.home=$HOME"
_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dclient.username=$RSE_USER_ID"
_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dlow.heap.usage.ratio=15"
_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dmaximum.heap.usage.ratio=40"
_RSE_JAVAOPTS="$_RSE_JAVAOPTS -DDSTORE_KEEPALIVE_ENABLED=true"
_RSE_JAVAOPTS="$_RSE_JAVAOPTS -DDSTORE_KEEPALIVE_RESPONSE_TIMEOUT=60000"
_RSE_JAVAOPTS="$_RSE_JAVAOPTS -DDSTORE_IO_SOCKET_READ_TIMEOUT=180000"
_RSE_JAVAOPTS="$_RSE_JAVAOPTS -DRSECOMM_LOGFILE_MAX=0"
```

Figure 8. rsed.envvars - RSE configuration file (continued)

```
  _RSE_JAVAOPTS="$_RSE_JAVAOPTS -Djob.monitor.port=$_RSE_JMON_PORT"
  _RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dlock.daemon.port=$_RSE_LOCKD_PORT"
  _RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dlock.daemon.cleanup.interval=1440"
  _RSE_JAVAOPTS="$_RSE_JAVAOPTS -showversion"
  _RSE_SERVER_CLASS=org.eclipse.dstore.core.server.Server
  _RSE_DAEMON_CLASS=com.ibm.etools.zos.server.RseDaemon
  _RSE_POOL_SERVER_CLASS=com.ibm.etools.zos.server.ThreadPoolProcess
  _RSE_LOCKD_CLASS=com.ibm.ftt.rse.mvs.server.miners.MVSLockDaemon
  _RSE_SERVER_TIMEOUT=120000
  _SCLMDT_BASE_HOME=$RSE_HOME
| _SCLMDT_WORK_HOME=$CGI_ISPHOME
  CGI_DTWORK=$_SCLMDT_WORK_HOME
| _CMDSERV_BASE_HOME=$CGI_ISPHOME
| _CMDSERV_CONF_HOME=$CGI_ISPCONF
| _CMDSERV_WORK_HOME=$CGI_ISPWORK
  #=============================================================
  # (6) additional environment variables
```

Figure 9. rsed.envvars - RSE configuration file (continued)

Note: Symbolic links are allowed when specifying directories in `rsed.envvars`.

The following definitions are required:

JAVA_HOME
Java home directory. The default is `/usr/lpp/java/J6.0`. Change to match your Java installation.

RSE_HOME
RSE home directory. The default is `/usr/lpp/rdz`. Change to match your Developer for System z installation.

_RSE_RSED_PORT
RSE daemon port number. The default is 4035. Can be changed if desired.

Note:

- Before selecting a port, verify that the port is available on your system with the TSO commands NETSTAT and NETSTAT PORTL.
- This port is used for client-host communication.
- The RSED started task can override the port number specified here.

_RSE_LOCKD_PORT
RSE lock daemon port number. The default is `4036`. Can be changed if desired.

Note:

- Before selecting a port, verify that the port is available on your system with the TSO commands **NETSTAT** and **NETSTAT PORTL**.
- All communication on this port is confined to your z/OS host machine.

_RSE_JMON_PORT
JES Job Monitor port number. The default is `6715`. Can be changed if desired.

Note:

- This value must match the port number set for JES Job Monitor in the `FEJJCNFG` configuration file. If these values differ, RSE cannot connect the client to JES Job Monitor. Refer to “FEJJCNFG, JES Job Monitor configuration file” on page 25 to learn how to define the variable for JES Job Monitor.

- Before selecting a port, verify that the port is available on your system with the TSO commands **NETSTAT** and **NETSTAT PORTL**.
- All communication on this port is confined to your z/OS host machine.

_RSE_HOST_CODEPAGE
The host codepage. The default is `IBM-1047`. Change to match your host codepage.

TZ Time zone selector. The default is `EST5EDT`. The default time zone is UTC +5 hours (Eastern Standard Time (EST) Eastern Daylight Savings Time (EDT)). Change to match your time zone.

Additional information can be found in the *UNIX System Services Command Reference* (SA22-7802).

LANG
Specifies the name of the default locale. The default is `C`. `C` specifies the POSIX locale and (for example) `Ja_JP` specifies the Japanese locale. Change to match your locale.

PATH Command path. The default is `/bin:/usr/sbin:.`. Can be changed if desired.

_CEE_DMPTARG
Language Environment (LE) z/OS UNIX dump location used by the Java Virtual Machine (JVM). The default is `/tmp`.

STEPLIB
Access MVS data sets not in LINKLIST/LPALIB. The default is `NONE`.

You can bypass the need of having (prerequisite) libraries in LINKLIST/LPALIB by uncommenting and customizing one or more of the following STEPLIB directives. Refer to “PARMLIB changes” on page 14 for more information about the usage of the libraries in the following list:

```
# RSE
STEPLIB=$STEPLIB:CEE.SCEERUN:CEE.SCEERUN2:CBC.SCLBDLL
# ISPF
STEPLIB=$STEPLIB:ISP.SISPLOAD:ISP.SISPLPA:SYS1.LINKLIB
# SCLM Developer Toolkit
STEPLIB=$STEPLIB:FEK.SFEKAUTH:FEK.SFEKLOAD
# zUnit, xUnit support for Enterprise COBOL and PL/I
STEPLIB=$STEPLIB:FEK.SFEKLOAD:SYS1.CSSLIB:SYS1.SIXMLOD1
```

Note:

- Using STEPLIB in z/OS UNIX has a negative performance impact.
- If one STEPLIB library is APF authorized, then all must be authorized. Libraries lose their APF authorization when they are mixed with non-authorized libraries in STEPLIB.
- Libraries that are designed for LPA placement might require additional program control and APF authorizations if they are accessed through LINKLIST or STEPLIB.
- Coding a STEPLIB DD statement in the server JCL does not set the requested STEPLIB concatenation.

_RSE_JAVAOPTS
Additional RSE-specific Java options. See “Defining extra Java startup parameters with _RSE_JAVAOPTS” on page 40 for more information about this definition.

The following definitions are required if ISPFs TSO/ISPF Client Gateway is used for the TSO Commands service or SCLM Developer Toolkit.

CGI_ISPHOME
Home directory for the ISPF code that provides the TSO/ISPF Client Gateway service. The default is `/usr/lpp/ispf`. Change to match your ISPF installation. This directive is only required when ISPFs TSO/ISPF Client Gateway is used.

CGI_ISPCONF
ISPF base configuration directory. The default is `/etc/rdz`. Change to match the location of `ISPF.conf`, the TSO/ISPF Client Gateway customization file. This directive is only required when ISPFs TSO/ISPF Client Gateway is used.

CGI_ISPWORK
ISPF base work directory. The default is `/var/rdz`. Change to match the location of the WORKAREA directory used by the TSO/ISPF Client Gateway. This directive is only required when ISPFs TSO/ISPF Client Gateway is used.

Note:

- The TSO/ISPF Client Gateway will add `/WORKAREA` to the path specified in `CGI_ISPWORK`. Do not add it yourself.
- If you did not use the `SFEKSAMP(FEKSETUP)` sample job to build the customizable environment, then you should verify that the WORKAREA directory exists in the path specified in `CGI_ISPWORK`. The directory permission bits must be 777.

STEPLIB
STEPLIB is described previously in the required definitions section.

_RSE_ISPF_OPTS
Additional TSO/ISPF Client Gateway specific Java options. The default is `""`. See “Defining extra Java startup parameters with _RSE_ISPF_OPTS” on page 45 for more information about this definition. This directive is only required when ISPFs TSO/ISPF Client Gateway is used.

CGI_ISPPREF
High level qualifier for the temporary data set created by the TSO/ISPF Client Gateway. The default is `&SYSPREF..ISPF.VCMISPF`. Uncomment and change to match your data set naming conventions. This directive is only required when ISPF’s TSO/ISPF Client Gateway is used.

The following variables can be used in the data set name:

- `&SYSUID.` to substitute the developer's user ID
- `&SYSPREF.` to substitute the developer's TSO prefix (or user ID if the TSO prefix cannot be determined)
- `&SYSNAME.` to substitute the system name as specified in the `IEASYMxx` parmlib member

Note: This directive requires ISPF APAR OA38740.

The following definitions are required if SCLM Developer Toolkit is used.

_SCLMDT_CONF_HOME
SCLM Developer Toolkit base configuration directory. The default is

/var/rdz/sclmdt. Change to match the location of the CONFIG directory used by SCLMDT to store SCLM project information. This directive is only required when SCLMDT is used.

Note: SCLMDT will add /CONFIG and /CONFIG/PROJECT to the path specified in SCLMDT_CNF_HOME. Do not add it yourself.

STEPLIB
STEPLIB is described previously in the required definitions section.

_SCLMDT_TRANTABLE
Name of the long/short name translation VSAM. The default is FEK.#CUST.LSTRANS.FILE. Uncomment and change to match the name used in the SCLM sample job ISP.SISPSAMP(FLM02LST). This directive is only required if the long/short name translation in SCLM Developer Toolkit is used.

ANT_HOME
Home directory for your Ant installation. The default is /usr/lpp/Apache/Ant/apache-ant-1.7.1. Change to match your Ant installation. This directive is only required when the JAVA/J2EE build support is used with SCLM Developer Toolkit.

The following definitions are optional. If omitted, default values will be used:

_RSE_PORTRANGE
Specifies the port range that the RSE server can open for communication with a client. Any port can be used by default. See “Defining the PORTRANGE available for RSE server” on page 39 for more information about this definition. This is an optional directive.

_BPXK_SETIBMOPT_TRANSPORT
Specifies the name of the TCP/IP stack to be used. The default is TCPIP. Uncomment and change to the requested TCP/IP stack name, as defined in the TCPIPJOBNAME statement in the related TCPIP.DATA. This is an optional directive.

Note:

- Coding a SYSTCPD DD statement in the server JCL does not set the requested stack affinity.
- When this directive is not active, RSE binds to every available stack on the system (BIND INADDRANY).

TMPDIR
Specifies the path used to store temporary files. The default is /tmp. Uncomment and change to use the requested path. This is an optional directive.

_RSE_FEK_SAF_CLASS
Specifies the security class where FEK.* profiles are defined. The default is FACILITY. Uncomment and change to enforce the usage of the specified value. This is an optional directive.

_RSE_LDAP_SERVER
Specifies the LDAP server host name used by the push-to-client function. The default is the current z/OS host name. Uncomment and change to enforce the usage of the specified value. This is an optional directive.

_RSE_LDAP_PORT
Specifies the LDAP server port used by the push-to-client function. The

default is 389. Uncomment and change to enforce the usage of the specified value. This is an optional directive.

_RSE_LDAP_PTC_GROUP_SUFFIX
Specifies the “O=<organization>,C=<country>” suffix needed to find the push-to-client groups within the LDAP server. The default is "O=PTC,C=DeveloperForZ". Uncomment and change to enforce the usage of the specified value. This is an optional directive.

GSK_CRL_SECURITY_LEVEL
Specifies the level of security SSL applications will use when contacting LDAP servers to check CRLs for revoked certificates during certificate validation. The default is MEDIUM. Uncomment and change to enforce the usage of the specified value. This is an optional directive. The following values are valid:

- LOW - Certificate validation will not fail if the LDAP server cannot be contacted.
- MEDIUM - Certificate validation requires the LDAP server to be contactable, but does not require a CRL to be defined. This is the default
- HIGH - Certificate validation requires the LDAP server to be contactable and a CRL to be defined.

Note: This directive requires z/OS 1.9 or higher.

GSK_LDAP_SERVER
Specifies one or more blank-separated LDAP server host names. Uncomment and change to enforce the usage of the specified LDAP servers to obtain their CRL. This is an optional directive.

The host name can either be a TCP/IP address or a URL. Each host name can contain an optional port number separated from the host name by a colon (:).

GSK_LDAP_PORT
Specifies the LDAP server port. The default is 389. Uncomment and change to enforce the usage of the specified value. This is an optional directive.

GSK_LDAP_USER
Specifies the distinguished name to use when connecting to the LDAP server. Uncomment and change to enforce the usage of the specified value. This is an optional directive.

GSK_LDAP_PASSWORD
Specifies the password to use when connecting to the LDAP server. Uncomment and change to enforce the usage of the specified value. This is an optional directive.

The following definitions are required, and should not be changed unless directed by the IBM support center:

_CEE_RUNOPTS
Language Environment (LE) runtime options. The default is "ALL31(ON) HEAP(32M,32K,ANYWHERE,KEEP,,) TRAP(ON)". Do not modify.

_BPX_SHAREAS
Run foreground processes in the same address space as the shell. The default is YES. Do not modify.

_BPX_SPAWN_SCRIPT
Run shell scripts directly from the spawn() function. The default is `YES`. Do not modify.

_EDC_ADD_ERRNO2
Show the reason code in z/OS UNIX error messages. The default is 1. Do not modify.

JAVA_PROPAGATE
Propagates the security and workload context during thread creation (Java version 1.4 and older only). The default is `NO`. Do not modify.

RSE_LIB
RSE library path. The default is `$RSE_HOME/lib`. Do not modify.

PATH
Command path. The default is `.:$JAVA_HOME/bin:$RSE_HOME/bin:$CGI_ISPHOME/bin:$PATH`. Do not modify.

LIBPATH
Library path. The default is too long to repeat. Do not modify.

CLASSPATH
Class path. The default is too long to repeat. Do not modify.

_RSE_ISPF_OPTS
Additional TSO Commands service-specific Java options. The default is `"&SESSION=SPAWN$_RSE_ISPF_OPTS"`. Do not modify.

_RSE_JAVAOPTS
Additional RSE-specific Java options. The default is too long to repeat. Do not modify.

_RSE_SERVER_CLASS
Java class for the RSE server. The default is `org.eclipse.dstore.core.server.Server`. Do not modify.

_RSE_DAEMON_CLASS
Java class for the RSE daemon. The default is `com.ibm.etools.zos.server.RseDaemon`. Do not modify.

_RSE_POOL_SERVER_CLASS
Java class for the RSE thread pool. The default is `com.ibm.etools.zos.server.ThreadPoolProcess`. Do not modify.

_RSE_LOCKD_CLASS
Java class for the RSE lock daemon. The default is `com.ibm.ftt.rse.mvs.server.miners.MVSLockDaemon`. Do not modify.

_RSE_SERVER_TIMEOUT
Time out value for the RSE server (waiting on the client) in milliseconds. The default is `120000` (2 minutes). Do not modify.

SCLMDT_BASE_HOME
Home directory for SCLM Developer Toolkit code. The default is `$RSE_HOME`. Do not modify.

SCLMDT_WORK_HOME
SCLM Developer Toolkit base work directory. The default is `$CGI_ISPHOME`. Do not modify.

CGI_DTWORK
SCLM Developer Toolkit support for older clients. The default is `$_SCLMDT_WORK_HOME`. Do not modify.

_CMDSERV_BASE_HOME
ISPF TSO/ISPF Client Gateway service support. The default is `$CGI_ISPHOME`. Do not modify.

_CMDSERV_CONF_HOME
ISPF TSO/ISPF Client Gateway service support. The default is `$CGI_ISPCONF`. Do not modify.

_CMDSERV_WORK_HOME
ISPF TSO/ISPF Client Gateway service support. The default is `$CGI_ISPWORK`. Do not modify.

Defining the PORTRANGE available for RSE server

This is a part of `rsed.envvars` customization that specifies the ports on which the RSE server can communicate with the client. This range of ports has no connection with the RSE daemon port.

To help understand the port usage, a brief description of RSE's connection process follows:

1. The client connects to host port 4035, RSE daemon.
2. The RSE daemon creates an RSE server thread.
3. The RSE server opens a host port for the client to connect. The selection of this port can be configured by the user, either on the client in the subsystem properties tab (this is not recommended) or through the `_RSE_PORTRANGE` definition in `rsed.envvars`.
4. The RSE daemon returns the port number to the client.
5. The client connects to the host port.

Note:

- The process is similar for the (optional) alternative connection method using REXEC/SSH.
- Refer to "Understanding Developer for System z" in the *Host Configuration Reference* (SC14-7290) for more information.

To specify the port range, for the client to communicate with z/OS, uncomment and customize the following line in `rsed.envvars`:

```
#_RSE_PORTRANGE=8108-8118
```

Note: Before selecting a port range, verify that the range is available on your system with the **NETSTAT** and **NETSTAT PORTL** commands.

The format of PORTRANGE is: `_RSE_PORTRANGE=min-max` (`max` is non-inclusive; for example `_RSE_PORTRANGE=8108-8118` means port numbers from `8108` up to `8117` are usable). The port number used by the RSE server is determined in the following order:

1. If a nonzero port number is specified in the subsystem properties on the client, then the specified port number is used. If the port is not available connect will fail. This setup is not recommended.

   Note: The host can deny this type of connection request by specifying the `deny.nonzero.port=true` directive in `rsed.envvars`. Refer to “Defining extra Java startup parameters with _RSE_JAVAOPTS” on page 40 for more information about this directive.

2. If the port number in the subsystem properties is 0, and if _RSE_PORTRANGE is specified in rsed.envvars, then the port range specified by _RSE_PORTRANGE is used. If no port in the range is available, connect will fail.

   RSE server does not need the port exclusively for the duration of the client connection. It is only in the time span between the (server) bind and the (client) connect that no other RSE server can bind to the port. This means that most connections will be using the first port in the range, with the rest of the range being a buffer in case of multiple simultaneous logons.
3. If the port number in the subsystem properties is 0, and _RSE_PORTRANGE is not specified in rsed.envvars, then any available port is used.

Defining extra Java startup parameters with _RSE_JAVAOPTS

With the different _RSE_*OPTS directives, rsed.envvars provides the possibility to give extra parameters to Java when it starts the RSE processes. The sample options included in rsed.envvars can be activated by uncommenting them.

_RSE_JAVAOPTS defines standard and RSE-specific Java options.

_RSE_JAVAOPTS=""
Variable initialization. Do not modify.

_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Xms1m -Xmx256m"
Set initial (Xms) and maximum (Xmx) heap size. The defaults are 1M and 256M respectively. Change to enforce the desired heap size values. If this directive is commented out, the Java default values will be used, which are 4M and 512M respectively (1M and 64M for Java 5.0).

Note: Refer to "Key resource definitions" in the *Host Configuration Reference* (SC14-7290) to determine the optimal values for this directive.

_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Ddaemon.log=/var/rdz/logs"
Directory holding the RSE daemon and server logging and RSE audit data. The default is /var/rdz/logs. Change to enforce the desired location. If this directive is commented out, the home directory of the user ID assigned to RSE daemon will be used. The home directory is defined in the OMVS security segment of the user ID.

Note: If this directive (or its counterpart, the home directory) does not specify an absolute path (the path does not start with a forward slash (/)), then the actual log location is relative to the configuration directory (by default /etc/rdz).

_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Duser.log=/var/rdz/logs"
Directory leading to the user-specific logs. The default is /var/rdz/logs. Change to enforce the desired location. If this directive is commented out or the value is a null string, the home directory of the client user ID will be used. The home directory is defined in the OMVS security segment of the user ID.

Note:

- If this directive (or its counterpart, the home directory) does not specify an absolute path (the path does not start with a forward slash (/)), then the actual log location is relative to the configuration directory (by default /etc/rdz).

- The complete path to the user logs is userlog/dstorelog/$LOGNAME/, where userlog is the value of the user.log directive, dstorelog is the value of the DSTORE_LOG_DIRECTORY directive and $LOGNAME is the clients user ID in uppercase.
- Ensure that the permission bits for userlog/dstorelog are set so that each client can create $LOGNAME.

_RSE_JAVAOPTS="$_RSE_JAVAOPTS -DDSTORE_LOG_DIRECTORY="

This directory is appended to the path specified in the user.log directive. Together they create the path leading to the user-specific logs. The default is a null-string. Change to enforce the usage of the specified directory. If this directive is commented out, .eclipse/RSE/ will be used.

Note:

- The complete path to the user logs is userlog/dstorelog/$LOGNAME/, where userlog is the value of the user.log directive, dstorelog is the value of the DSTORE_LOG_DIRECTORY directive, and $LOGNAME is the clients user ID in uppercase.
- The directory specified here is relative to the directory specified in user.log, and thus might not start with a forward slash (/).
- Ensure that the permission bits for userlog/dstorelog are set so that each client can create $LOGNAME.

The following directives are commented out by default.

#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dmaximum.clients=60"

Maximum amount of clients serviced by one thread pool. The default is 60. Uncomment and customize to limit the number of clients per thread pool. Note that other limits might prevent RSE from reaching this limit.

#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dmaximum.threads=1000"

Maximum amount of active threads in one thread pool to allow new clients. The default is 1000. Uncomment and customize to limit the number of clients per thread pool based on the number of threads in use. Note that each client connection uses multiple threads (16 or more) and that other limits might prevent RSE from reaching this limit.

Note: This value must be lower than the setting for MAXTHREADS and MAXTHREADTASKS in SYS1.PARMLIB(BPXPRMxx).

#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dminimum.threadpool.process=1"

The minimum number of active thread pools. The default is 1. Uncomment and customize to start at least the listed number of thread pool processes. Thread pool processes are used for load balancing the RSE server threads. More new processes are started when they are needed. Starting the new processes up front helps prevent connection delays but uses more resources during idle times.

Note: If the single.logon directive is active, then there will be at least 2 thread pools started, even if minimum.threadpool.process is set to 1. The default setting for single.logon in rsed.envvars is active.

#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dmaximum.threadpool.process=100"

The maximum number of active thread pools. The default is 100. Uncomment and customize to limit the number of thread pool processes. Thread pool processes are used for load balancing the RSE server threads, so limiting them will limit the amount of active client connections.

#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dipv6=true"
TCP/IP version. The default is `false`, which means that an IPv4 interface will be used. Uncomment and specify `true` to use an IPv6 interface.

#_RSE_JAVAOPTS="$_RSE_JAVAOPTS –Dkeep.last.log=true"
Keep a copy of the host log files belonging to the previous session. The default is `false`. Uncomment and specify `true` to rename the previous log files to `*.last` during server startup and client connect.

#_RSE_JAVAOPTS="$_RSE_JAVAOPTS –Denable.standard.log=true"
Write the stdout and stderr streams of the thread pools to a log file. The default is `false`. Uncomment and specify `true` to save the stdout and stderr streams. The resulting log files are located in the directory referenced by the `daemon.log` directive.

Note:

- The **MODIFY RSESTANDARDLOG** operator command can be used to dynamically stop or start the update of the stream log files.
- There are no user-specific `stdout.log` and `stderr.log` log files when the `enable.standard.log` directive is active. The user-specific data is now written to the matching RSE thread pool stream.

#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Denable.port.of.entry=true"
Port Of Entry (POE) check option. The default is `false`. Uncomment and specify `true` to enforce POE checking for client connections. During POE checking, the IP address of the client is mapped into a network access security zone by your security software. The client user ID must have permission to use the profile that defines the security zone.

Note:

- POE checking must also be enabled in your security product.
- Enabling POE checking will enable it for other z/OS UNIX services also, such as INETD.

#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Denable.certificate.mapping=false"
Use your security software to authenticate a logon with a X.509 certificate. The default is `true`. Uncomment and specify `false` to have RSE daemon do the authentication without relying on the X.509 support of your security software.

#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Denable.automount=true"
Support home directories created by z/OS UNIX automount. The default is `false`. Uncomment and specify `true` to ensure that z/OS UNIX automount uses the client user ID as owner of the directory.

Note: z/OS UNIX automount uses the user ID of the process that invoked the service when creating a file system. If this option is disabled, this process is the RSE thread pool server (user ID STCRSE). If this option is enabled, a new, temporary process is created using the client user ID before invoking the service.

#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Denable.audit.log=true"
Audit option. The default is `false`. Uncomment and specify `true` to enforce audit logging of actions done by clients. Audit logs are written to the RSE daemon log location. See the `daemon.log` option of the `_RSE_JAVAOPTS` variable to know where this is.

#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Daudit.cycle=30"
Number of days stored in 1 audit log file. The default is 30. Uncomment and customize to control how much audit data is written to 1 audit log file. The maximum value is 365.

#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Daudit.retention.period=0"
Number of days audit logs are kept. The default is 0 (no limit). Uncomment and customize to delete audit logs after a given number of days. The maximum value is 365.

#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Daudit.log.mode=RW.R."
Access permission mask for audit logs. The default is RW.R., which allows the owner read and write access. The owner's default group has read access and everyone else has no access. Uncomment and customize to set the desired access permissions.

UNIX standards dictate that permissions can be set for three types of users: owner, group, and other. The fields in the audit.log.mode mask match this order, and the fields are separated by a period (.). Each field can either be empty, or have R, W, or RW as value (R = read, W = write).

#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Daudit.action=<user exit>"
Name of a user exit which will be invoked when an audit log file is closed. There is no default value, but a sample exit is provided in /usr/lpp/rdz/samples/process_audit.rex. Uncomment and specify the full pathname of the user exit program to enable post-processing of audit logs.

#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Daudit.action.id=<userid>"
User ID to be used for running the exit specified in the audit.action variable. The default is the user ID assigned to RSE daemon. Uncomment and specify a user ID to use the specified ID for executing the audit post-processing exit.

#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dlogon.action=<user_exit>"
Name of a user exit which will be invoked when a user logs on. There is no default value, but a sample exit is provided in /usr/lpp/rdz/samples/process_logon.sh. Uncomment and specify the full pathname of the user exit program to enable post-processing of a logon.

#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dlogon.action.id=<userid>"
User ID to be used for running the exit specified in the logon.action variable. The default is the user ID assigned to RSE daemon. Uncomment and specify a user ID to use the specified ID for executing the logon post-processing exit.

#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Ddeny.nonzero.port=true"
Disallow the client to choose the communication port number. The default is false. Uncomment and specify true to refuse connections where the client specifies which host port must be used by RSE server for the connection. Refer to “Defining the PORTRANGE available for RSE server” on page 39 for more information.

#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dsingle.logon=false"
Disallow a user ID to log on multiple times. The default is true. Uncomment and specify false to allow a user ID to log on multiple times to a single RSE daemon.

Note:

- A second logon attempt will cause the first one to be canceled by the host if this directive is not active or set to `true`. This cancel will be accompanied by console message FEK210I.
- If the `single.logon` directive is active, then there will be at least 2 thread pools started, even if `minimum.threadpool.process` is set to 1. The default setting for `minimum.threadpool.process` in `rsed.envvars` is 1.

#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dprocess.cleanup.interval=0"
Automatically remove RSE thread pools that are in an unrecoverable error state. By default, erroneous RSE thread pools are not automatically removed. Uncomment and customize to automatically remove erroneous RSE thread pool servers at every interval (interval unit is seconds). Specifying `0` does not start an interval timer, but erroneous RSE thread pool servers are removed when the RSE daemon checks the RSE thread pools during a new client logon or the DISPLAY PROCESS command.

#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dreject.logon.threshold=1000000"
A thread pool opening a file larger than the specified size will not accept new logon requests until the file is loaded. The default file size is `1000000` (1 million) bytes. Uncomment and customize to specify the file size at which a thread pool is to ignore logon requests when such a file is opened. Note that other thread pools will still be eligible to accept new logon requests.

#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dinclude.c=/etc/rdz/include.conf"
This variable points to a fully qualified z/OS UNIX file containing a list of forced includes for content assist on C code. A forced include consists of a file or directory, data set, or data set member which will be parsed when a content assist operation is performed, regardless of whether that file or member was included in the source code using a pre-processor directive. Uncomment and customize to specify the name of the configuration file.

#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dinclude.cpp=/etc/rdz/include.conf"
This variable points to a fully qualified z/OS UNIX file containing a list of forced includes for content assist on C++ code. A forced include consists of a file or directory, data set, or data set member which will be parsed when a content assist operation is performed, regardless of whether that file or member was included in the source code using a pre-processor directive. Uncomment and customize to specify the name of the configuration file.

#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -DCPP_CLEANUP_INTERVAL=60000"
Cleanup interval for unused C/C++ header files in milliseconds. The default is `60000` (1 minute). Uncomment and customize to change the cleanup interval. Specifying 0 will prevent caching of C/C++ header files, reducing performance of remote content assist in the editor.

#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -DRIS_BUFFER=8"
Buffer size, in megabytes, used during remote index creation. The default is 8 MB. Uncomment and customize to change the buffer size. Valid values are whole numbers between 1 and 2000 (both inclusive). Note that a bigger buffer speeds up index creation, but uses a bigger portion of the thread pool's Java heap. The buffer is automatically flushed to the index if it is full before index creation ends.

#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -DAPPLID=FEKAPPL"
RSE server application ID. The default is `FEKAPPL`. Uncomment and customize this option to enforce the use of the desired application ID.

Note:

- The application ID must be defined to your security software. Failure to do so will prevent the client from logging on.
- Refer to "Using PassTickets" in *Host Configuration Reference* (SC14-7290) for the security implications when changing this value.
- This value must match the application ID set for JES Job Monitor in the `FEJJCNFG` configuration file. If these values differ, RSE cannot connect the client to JES Job Monitor. Refer to "FEJJCNFG, JES Job Monitor configuration file" on page 25 to learn how to define the variable for JES Job Monitor.

#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -DDENY_PASSWORD_SAVE=true"
Password save option. The default is `false`. Uncomment and specify `true` to prevent users from saving their host password on the client. Previously saved passwords will be removed. This option only works with clients version 7.1 and higher.

#_RSE_JAVAOPTS="$_RSE_JAVAOPTS –DHIDE_ZOS_UNIX=true"
Hide z/OS UNIX option. The default is `false`. Uncomment and specify `true` to prevent users from seeing z/OS UNIX elements (directory structure and command line) on the client. This option only works with clients version 7.6 and higher.

#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -DDISABLE_DELETE_IN_SUBPROJECT=true"
Disable the Delete menu item in the context menu of z/OS subprojects. The default is `false`. Uncomment and specify `true` to prevent users from using the Delete menu item in the context menu of z/OS subprojects. This option only works with clients version 8.0.1 and higher.

#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -DDSTORE_IDLE_SHUTDOWN_TIMEOUT=3600000"
Disconnect idle clients. By default, idle clients are not disconnected. Uncomment and customize to disconnect clients who are idle for the listed amount of milliseconds (3600000 equals 1 hour).

#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -DDSTORE_TCP_NO_DELAY=true"
Disable the TCP/IP DELAY ACK function. The default is `false`. Uncomment and specify `true` to stop TCP/IP from doing DELAY ACK for Developer for System z client-host communication.

#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -DDSTORE_TRACING_ON=true"
Start dstore tracing. Use only when directed by the IBM support center. Note that the resulting `.dstoreTrace` log file is created in Unicode (ASCII), not EBCDIC.

#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -DDSTORE_MEMLOGGING_ON=true"
Start dstore memory tracing. Use only when directed by the IBM support center. Note that the resulting `.dstoreMemLogging` log file is created in Unicode (ASCII), not EBCDIC.

Defining extra Java startup parameters with _RSE_ISPF_OPTS

With the different `_RSE_*OPTS` directives, `rsed.envvars` provides the possibility to give extra parameters to Java when it starts the RSE processes. The sample options included in `rsed.envvars` can be activated by uncommenting them.

The `_RSE_ISPF_OPTS` directives are RSE-specific Java options and are only in effect when ISPF's TSO/ISPF Client Gateway is used by the Developer for System z. (This is the default.)

_RSE_ISPF_OPTS=""
Variable initialization. Do not modify.

_RSE_ISPF_OPTS="$_RSE_ISPF_OPTS&ISPPROF=&SYSUID..ISPPROF"
Use an existing ISPF profile for the ISPF initialization. Uncomment and change the data set name to use the specified ISPF profile.

The following variables can be used in the data set name:

- `&SYSUID.` to substitute the developer's user ID
- `&SYSPREF.` to substitute the developer's TSO prefix (or user ID if the TSO prefix cannot be determined)
- `&SYSNAME.` to substitute the system name as specified in the `IEASYMxx` parmlib member

ISPF.conf, ISPF's TSO/ISPF Client Gateway configuration file

ISPF's TSO/ISPF Client Gateway uses the definitions in `ISPF.conf` to create a valid environment to execute batch TSO and ISPF commands. Developer for System z uses this environment to run some MVS based services. These services include the TSO Commands service and SCLM Developer Toolkit.

`ISPF.conf` is located in `/etc/rdz/`, unless you specified a different location when you customized and submitted job `FEK.SFEKSAMP(FEKSETUP)`. See “Customization setup” on page 13 for more details. You can edit the file with the TSO **OEDIT** command.

Definitions must start in column 1. Comment lines start with an asterisk (*) when using a US code page. Data lines can only have a directive and its assigned value. Comments are not allowed on the same line. Line continuations are not supported. When concatenating data set names, add them on the same line and separate the names with a comma (,).

In addition to providing the correct names for the ISPF data sets, you must also add the TSO Commands service data set name, `FEK.SFEKPROC`, to the `SYSPROC` or `SYSEXEC` statement, as shown in the following example.

```
* REQUIRED:
sysproc=ISP.SISPCLIB,FEK.SFEKPROC
ispmlib=ISP.SISPMENU
isptlib=ISP.SISPTENU
ispplib=ISP.SISPPENU
ispslib=ISP.SISPSLIB
ispllib=ISP.SISPLOAD

* OPTIONAL:
*allocjob = ISP.SISPSAMP(ISPZISP2)
*ISPF_timeout = 900
```

Figure 10. ISPF.conf - ISPF configuration file

Note:

- You can add your own DD-like statements and data set concatenations to customize the TSO environment, thus mimicking a TSO logon procedure. See "Customizing the TSO environment" in the *Host Configuration Reference* (SC14-7290) for more details.
- The TSO/ISPF Client Gateway might not function properly if you use a (third party) product that intercepts ISPF commands, such as **ISPSTART**. Check the documentation for that product on how it can be disabled for Developer for

System z. If the product requires the allocation of a specific DD statement to DUMMY, you can simulate this in ISPF.conf by allocating that DD statement to nullfile.

For example:

```
ISPTRACE=nullfile
```

- When using the allocjob directive, be careful not to undo the DD definitions done earlier in ISPF.conf.
- System abend 522 for module ISPZTSO is to be expected if the JWT parameter in the SMFPRMxx parmlib member is set lower than the ISPF_timeout value in ISPF.conf. This does not impact Developer for System z operations, as the TSO/ISPF Client Gateway is restarted automatically when needed.
- Changes are active for all new invocations. No server restart is needed.

Optional components

This section combines a variety of optional customization tasks. Follow the instructions in the appropriate section to configure the desired service.

Customizations to Developer for System z stand-alone components:

- Chapter 3, "(Optional) Common Access Repository Manager (CARMA)," on page 49
- Chapter 4, "(Optional) SCLM Developer Toolkit," on page 77
- Chapter 5, "(Optional) Application Deployment Manager," on page 85

Customizations to Developer for System z configuration files:

- "(Optional) pushtoclient.properties, Host-based client control" on page 93
- "(Optional) ssl.properties, RSE SSL encryption" on page 96
- "(Optional) rsecomm.properties, RSE tracing" on page 98

Developer for System z related customizations to (or for) other products:

- "(Optional) DB2 stored procedure" on page 101
- "(Optional) z/OS UNIX subprojects" on page 103
- "(Optional) File Manager support" on page 109
- "(Optional) Include preprocessor support" on page 104
- "(Optional) Enterprise Service Tools support" on page 106
- "(Optional) CICS bidirectional language support" on page 106
- "(Optional) Diagnostic IRZ messages for generated code" on page 107
- "(Optional) WORKAREA and /tmp cleanup" on page 109

Installation verification

The detailed description of the various installation verification programs (IVPs) is located in Chapter 7, "Installation verification," on page 111, because some of the IVPs are for the optional components.

You can test the basic functions with the following scenario:

1. Start the JMON started task (or user job). The startup information in DD STDOUT should end with the following message:

   ```
   JM200I Server initialization complete.
   ```

   If the job ends with return code 66, then FEK.SFEKAUTH is not APF authorized.

2. Start the LOCKD started task (or user job). The lock daemon issues the following console message upon successful startup:

   ```
   FEK501I Lock daemon started, port=4036, cleanup interval=1440, log level=1
   ```

3. Start the RSED started task (or user job) with the `IVP=IVP` parameter. With this parameter, the server will end after doing some installation verification tests. Check DD STDOUT for messages indicating that the following IVPs were successful:
   - Java startup
   - JES Job Monitor connection
   - Lock daemon connection
   - TCP/IP setup
   - PassTicket generation

4. Start the RSED started task (or user job) without the IVP parameter. RSE daemon issues the following console message upon successful startup:

   ```
   FEK002I RseDaemon started. (port=4035)
   ```

5. Issue the following operator commands and verify in the resulting console messages that the tests ran successfully:

   ```
   F RSED,APPL=IVP PASSTICKET,userid
   F RSED,APPL=IVP DAEMON,userid
   F RSED,APPL=IVP ISPF,userid
   ```

   Replace `userid` with a valid TSO user ID.

Chapter 3. (Optional) Common Access Repository Manager (CARMA)

Common Access Repository Manager (CARMA) is a server platform for Repository Access Managers (RAMs). A RAM is an Application Programming Interface (API) for a z/OS based Software Configuration Manager (SCM). By wrapping the SCM functionality in a RAM, a single API is available for a client to access any supported SCM.

Developer for System z provides multiple pre-built RAMs, as well as source code examples for creating your own RAM.

Requirements and checklist

You will need the assistance of a security administrator and a TCP/IP administrator to complete this customization task, which requires the following resources or special customization tasks:

- (Optional) TCP/IP port range for internal communication
- (Optional) Security rule to allow developers update to CARMA VSAM files
- (Optional) Security rule to allow users to submit CRA* jobs
- (Optional) LPA update

In order to start using CARMA at your site, you must perform the following tasks. Unless otherwise indicated, all tasks are mandatory.

1. Choose a method to start CARMA and choose which RAMs should be activated. Several combinations of RAMs and server startup methods are available as a preconfigured setup. For details, see “Select server startup method and active RAM” on page 50.
2. Create CARMA VSAM data sets. For details, see “CARMA VSAM data sets” on page 66 and “CARMA Repository Access Managers (RAMs)” on page 67.
3. Initial customization of RSE configuration files to interface with CARMA. The complete customization is dependent on the method chosen to start CARMA. For details, see “CRASRV.properties, RSE interface to CARMA” on page 60.
4. Depending on the chosen CARMA startup method and the chosen RAMs, do the required customization of the related configuration files. For details see:
   - “crastart*.conf, CRASTART server startup” on page 62
   - “CRASUB*, batch submit server startup” on page 65
5. Optionally customize CA Endevor® SCM specific configuration members. For details see “CRASHOW and CRATMAP, CA Endevor® SCM RAM configuration files” on page 69, “CRANDVRA, CA Endevor® SCM RAM allocation exec” on page 70, and “CA Endevor® SCM RAM batch actions” on page 70.
6. Optionally create a data set allocation exec. For details, see “(Optional) Custom allocation exec” on page 75.
7. Optionally create CRAXJCL as replacement for IRXJCL. For details, see “(Optional) IRXJCL versus CRAXJCL” on page 75.

Note: The sample members referenced in this chapter are located in `FEK.#CUST.*` and `/etc/rdz`, unless you specified a different location when you customized and submitted job `FEK.SFEKSAMP(FEKSETUP)`. See "Customization setup" on page 13 for more details.

Select server startup method and active RAM

Developer for System z supports multiple methods to start a CARMA server. Developer for System z also provides multiple Repository Access Managers (RAMs), which can be divided into two groups, production RAMs and sample RAMs. This publication describes several possible combinations of RAMs and server startup methods. Each of the described configuration scenarios is available as a preconfigured setup.

CARMA server startup

Developer for System z supports multiple methods to start a CARMA server. Each method has benefits and drawbacks.

CRASTART

The "CRASTART" method starts the CARMA server as a subtask within RSE. It provides a very flexible setup by using a separate configuration file that defines data set allocations and program invocations needed to start a CARMA server. This method provides the best performance and uses the fewest resources, but requires that module `CRASTART` is located in LPA.

Batch submit

The "batch submit" method starts the CARMA server by submitting a job. This is the default method used in the provided sample configuration files. The benefit of this method is that the CARMA logs are easily accessible in the job output. It also allows the use of custom server JCL for each developer, which is maintained by the developer himself. However, this method uses one JES initiator per developer starting a CARMA server.

(deprecated) TSO/ISPF Client Gateway

The "TSO/ISPF Client Gateway" method uses ISPF's TSO/ISPF Client Gateway to create a TSO or ISPF environment, in which the CARMA server is started. It allows for flexible data set allocations using the possibilities of `ISPF.conf`. However, this method is not suited to access SCMs that interfere with normal TSO or ISPF operations.

Note: The "TSO/ISPF Client Gateway" connection method has been marked as deprecated. Although it is still supported, this function will no longer be enhanced, and the documentation has moved to a white paper, *Using ISPF Client Gateway to provide CARMA services* (SC14-7292), available in the Developer for System z library, `http://www-01.ibm.com/software/awdtools/rdz/library/`.

Production RAMs

Production type RAMs are fully functional, pre-built RAMs that can be used to access a SCM in a production environment.

CA Endevor® SCM RAM

The IBM® Rational® Developer for System z Interface for CA Endevor® Software Configuration Manager gives Developer for System z clients direct access to CA Endevor® SCM. From here on, IBM® Rational® Developer for System z Interface for CA Endevor® SCM is abbreviated to CA Endevor® SCM RAM.

CA Endevor® SCM packages RAM

The CA Endevor® SCM packages RAM gives Developer for System z clients direct access to CA Endevor® SCM packages.

Sample RAMs

Sample RAMs are provided for the purpose of testing the configuration of your CARMA environment and as examples for developing your own RAMs (source code is included).

Do NOT use the provided sample RAMs in a production environment.

PDS RAM

The PDS RAM gives a data set list similar to **MVS Files -> My Data Sets** in the Remote Systems view.

Skeleton RAM

The skeleton RAM gives a functional framework that can be used as starting point to develop your own RAM.

SCLM RAM

The SCLM RAM gives a basic entry into SCLM, ISPF's Software Configuration Manager. The SCLM RAM is not enabled by default.

Preconfigured RAM and server startup combinations

Several combinations of RAMs and server startup methods are available as a preconfigured setup. The listed scenarios only need minor customization to fit your environment.

- "CRASTART with CA Endevor® SCM RAM"
- "CRASTART with sample RAMs" on page 54
- "Batch submit with CA Endevor® SCM RAM" on page 55
- "Batch submit with sample RAMs" on page 58

Detailed information on the different steps of each scenario can be found in "CARMA configuration details" on page 59.

Note that it is possible to add a RAM to any CARMA setup, now or somewhere in the future. Refer to "(Optional) Supporting multiple RAMs" on page 73 for more information on adding a RAM to an existing setup.

CRASTART with CA Endevor® SCM RAM

The information in this section describes how to set up CARMA with the following specifications:

- Server startup: CRASTART method (requires that CRASTART is in LPA)
- RAM: CA Endevor® SCM RAM

This customization step can be bypassed if you want to use one of the other scenarios with different specifications.

Create CARMA VSAM data sets

Customize and submit the following JCL jobs to define and populate the CARMA related VSAM data sets. Refer to the documentation within the member for customization instructions. Note that existing VSAM data sets will be replaced if present.

Refer to "CARMA VSAM data sets" on page 66 for more details on this step.

- FEK.#CUST.JCL(CRA$VCAD)
- FEK.#CUST.JCL(CRA$VCAS)
- FEK.#CUST.JCL(CRA$VMSG)

Customize CRASRV.properties

RSE server uses the settings in /etc/rdz/CRASRV.properties to start and connect to a CARMA server. You can edit the file with the TSO **OEDIT** command. Note that the RSED started task must be restarted before changes are in effect.

When you use the default file locations, the only required changes are changing the value of the clist.dsname directive to *CRASTART and changing the value of crastart.configuration.file to /etc/rdz/crastart.endevor.conf. Refer to "CRASRV.properties, RSE interface to CARMA" on page 60 for more information on the different directives.

```
port.start=5227
port.range=100
startup.script.name=/usr/lpp/rdz/bin/carma.startup.rex
clist.dsname=*CRASTART
crastart.stub=/usr/lpp/rdz/bin/CRASTART
crastart.configuration.file=/etc/rdz/crastart.endevor.conf
crastart.syslog=Partial
crastart.timeout=420
#crastart.steplib=FEK.SFEKLPA
#crastart.tasklib=TASKLIB
```

Figure 11. CRASRV.properties - CRASTART with CA Endevor® SCM RAM

Customize crastart.endevor.conf

CRASTART uses the definitions in /etc/rdz/crastart.endevor.conf to create a valid (TSO/ISPF) environment to start a CARMA server. You can edit the file with the TSO **OEDIT** command. Note that changes are in effect for all CARMA servers started after the update.

Refer to the documentation within the file for customization instructions. Refer to "crastart*.conf, CRASTART server startup" on page 62 for more information on the CRASTART startup method.

Note: Due to page width limitations, some lines in the following sample wrapped onto the next line. All lines that start with an indentation should be added to the end of the previous line.

```
* DD used by RAM
TYPEMAP = FEK.#CUST.PARMLIB(CRATMAP)
SHOWVIEW= FEK.#CUST.PARMLIB(CRASHOW)
* uncomment CRABCFG and CRABSKEL to use batch actions
*CRABCFG = FEK.#CUST.PARMLIB(CRABCFG)
*CRABSKEL= FEK.#CUST.CNTL
CONLIB  = CA.NDVR.CONLIB                                     * NDVR R12
*CONLIB  = CA.NDVR.CSIQLOAD                                  * NDVR R14
-COMMAND=ALLOC FI(JCLOUT)   SYSOUT(A) WRITER(INTRDR) RECFM(F) LRECL(80)
  BLKSIZE(80)
-COMMAND=ALLOC FI(EXT1ELM)  NEW DELETE DSORG(PS) RECFM(V,B) LRECL(4096)
  BLKSIZE(27998) SPACE(5,5) TRACKS UNIT(SYSALLDA)
-COMMAND=ALLOC FI(EXT2ELM)  NEW DELETE DSORG(PS) RECFM(V,B) LRECL(4096)
  BLKSIZE(27998) SPACE(5,5) TRACKS UNIT(SYSALLDA)
-COMMAND=ALLOC FI(EXT1DEP)  NEW DELETE DSORG(PS) RECFM(V,B) LRECL(4096)
  BLKSIZE(27998) SPACE(5,5) TRACKS UNIT(SYSALLDA)
C1EXMSGS= SYSOUT(H)
C1MSGS1 = SYSOUT(H)
MSG3FILE= DUMMY

* DD used by CARMA server (CRASERV)
TASKLIB = FEK.SFEKLOAD,CA.NDVR.AUTHLIB,CA.NDVRU.AUTHLIB      * NDVR R12
*TASKLIB = FEK.SFEKLOAD,CA.NDVR.CSIQAUTH,CA.NDVR.CSIQAUTU    * NDVR R14
CRADEF  = FEK.#CUST.CRADEF
CRAMSG  = FEK.#CUST.CRAMSG
CRASTRS = FEK.#CUST.CRASTRS
CARMALOG= SYSOUT(H)
SYSPRINT= SYSOUT(H)

* DD used by ISPF (via NDVRC1)
-COMMAND=ALLOC FI(ISPCTL0) NEW DELETE DSORG(PS) RECFM(F,B) LRECL(80)
  BLKSIZE(32720) SPACE(5,5) TRACKS UNIT(SYSALLDA)
-COMMAND=ALLOC FI(ISPCTL1) NEW DELETE DSORG(PS) RECFM(F,B) LRECL(80)
  BLKSIZE(32720) SPACE(5,5) TRACKS UNIT(SYSALLDA)
-COMMAND=ALLOC FI(ISPPROF) NEW DELETE DSORG(PO) RECFM(F,B) LRECL(80)
  BLKSIZE(32720) SPACE(5,5) TRACKS UNIT(SYSALLDA) DIR(5)
ISPTABL = -ISPPROF
ISPTLIB = -ISPPROF,ISP.SISPTENU
ISPMLIB = ISP.SISPMENU
ISPPLIB = ISP.SISPPENU
ISPSLIB = ISP.SISPSENU

* DD used by TSO (IKJEFT01)
SYSPROC = FEK.SFEKPROC                                       * CRANDVRA
SYSTSIN = DUMMY
SYSTSPRT= SYSOUT(H)

PROGRAM=IKJEFT01 %CRANDVRA NDVRC1 PGM(CRASERV) PARM(&CRAPRM1. &CRAPRM2.)
```

Figure 12. crastart.endevor.conf - CRASTART with CA Endevor® SCM RAM

(Optional) Additional CA Endevor® SCM RAM customization

The CA Endevor® SCM RAM has additional components that can be customized if you want to customize them.

- The CA Endevor® SCM RAM has two configuration files, `FEK.#CUST.PARMLIB(CRASHOW)` and `FEK.#CUST.PARMLIB(CRATMAP)`, that can be customized. Refer to "CRASHOW and CRATMAP, CA Endevor® SCM RAM configuration files" on page 69 for more information.
- The CA Endevor® SCM RAM has an allocation exec, `FEK.SFEKPROC(CRANDVRA)`, that can be customized. Refer to "CRANDVRA, CA Endevor® SCM RAM allocation exec" on page 70 for more information.

- The CA Endevor® SCM RAM supports doing CA Endevor® SCM actions in batch mode. Batch-actions requires a configuration file, `FEK.#CUST.PARMLIB(CRABCFG)`, and a skeleton JCL, `FEK.#CUST.CNTL(CRABATCA)`, that must be customized. See "CA Endevor® SCM RAM batch actions" on page 70 for more information.

CRASTART with sample RAMs

The information in this section describes how to set up CARMA with the following specifications:

- Server startup: CRASTART method (requires that CRASTART is in LPA)
- RAM: sample RAMs (not for production purposes)

This customization step can be bypassed if you want to use one of the other scenarios with different specifications.

Create CARMA VSAM data sets

Customize and submit the following JCL jobs to define and populate the CARMA related VSAM data sets. Refer to the documentation within the member for customization instructions. Note that existing VSAM data sets will be replaced if present.

Refer to "CARMA VSAM data sets" on page 66 and "CARMA Repository Access Managers (RAMs)" on page 67 for more details on this step.

CARMA

- `FEK.#CUST.JCL(CRA$VDEF)`
- `FEK.#CUST.JCL(CRA$VMSG)`
- `FEK.#CUST.JCL(CRA$VSTR)`

Sample RAMs

- `FEK.#CUST.JCL(CRA#VPDS)`

Customize CRASRV.properties

RSE server uses the settings in `/etc/rdz/CRASRV.properties` to start and connect to a CARMA server. You can edit the file with the TSO **OEDIT** command. Note that the RSED started task must be restarted before changes are in effect.

When using the default file locations, the only required change is changing the value of the `clist.dsname` directive to `*CRASTART`. Refer to "CRASRV.properties, RSE interface to CARMA" on page 60 for more information on the different directives.

```
port.start=5227
port.range=100
startup.script.name=/usr/lpp/rdz/bin/carma.startup.rex
clist.dsname=*CRASTART
crastart.stub=/usr/lpp/rdz/bin/CRASTART
crastart.configuration.file=/etc/rdz/crastart.conf
crastart.syslog=Partial
crastart.timeout=420
#crastart.steplib=FEK.SFEKLPA
#crastart.tasklib=TASKLIB
```

Figure 13. CRASRV.properties - CRASTART with sample RAMs

Customize crastart.conf

CRASTART uses the definitions in /etc/rdz/crastart.conf to create a valid (TSO/ISPF) environment to start a CARMA server. You can edit the file with the TSO **OEDIT** command. Note that changes are in effect for all CARMA servers started after the update.

Refer to the documentation within the file for customization instructions. Refer to "crastart*.conf, CRASTART server startup" on page 62 for more information on the CRASTART startup method.

```
* DD used by RAM
CRARAM1 = FEK.#CUST.CRARAM1                                * PDS RAM
*CRARAM2 = FEK.#CUST.CRARAM2                              * SCLM RAM

* DD used by CARMA server (CRASERV)
TASKLIB = FEK.SFEKLOAD
CRADEF  = FEK.#CUST.CRADEF
CRAMSG  = FEK.#CUST.CRAMSG
CRASTRS = FEK.#CUST.CRASTRS
CARMALOG= SYSOUT(H)
SYSPRINT= SYSOUT(H)

* DD used by ISPF (ISPSTART)
-COMMAND=ALLOC FI(ISPCTL0) NEW DELETE DSORG(PS) RECFM(F,B) LRECL(80)
  BLKSIZE(32720) SPACE(5,5) TRACKS UNIT(SYSALLDA)
-COMMAND=ALLOC FI(ISPCTL1) NEW DELETE DSORG(PS) RECFM(F,B) LRECL(80)
  BLKSIZE(32720) SPACE(5,5) TRACKS UNIT(SYSALLDA)
-COMMAND=ALLOC FI(ISPPROF) NEW DELETE DSORG(PO) RECFM(F,B) LRECL(80)
  BLKSIZE(32720) SPACE(5,5) TRACKS UNIT(SYSALLDA) DIR(5)
ISPTABL = -ISPPROF
ISPTLIB = -ISPPROF,ISP.SISPTENU
ISPMLIB = ISP.SISPMENU
ISPPLIB = ISP.SISPPENU
ISPSLIB = ISP.SISPSENU

* DD used by TSO (IKJEFT01)
SYSTSIN = DUMMY
SYSTSPRT= SYSOUT(H)

PROGRAM=IKJEFT01 ISPSTART PGM(CRASERV) PARM(&CRAPRM1. &CRAPRM2.)
```

Figure 14. crastart.conf - CRASTART with sample RAMs

Note: Due to page width limitations, some lines in the sample wrapped onto the next line. All lines that start with an indentation should be added to the end of the previous line.

Batch submit with CA Endevor® SCM RAM

The information in this section describes how to set up CARMA with the following specifications:

- Server startup: batch submit method (requires JES initiators)
- RAM: CA Endevor® SCM RAM

This customization step can be bypassed if you want to use one of the other scenarios with different specifications.

Create CARMA VSAM data sets

Customize and submit the following JCLs to define and populate the CARMA related VSAM data sets. Refer to the documentation within the member for customization instructions. Note that existing VSAM data sets will be replaced if present.

Refer to "CARMA VSAM data sets" on page 66 for more details on this step.

- FEK.#CUST.JCL(CRA$VCAD)
- FEK.#CUST.JCL(CRA$VCAS)
- FEK.#CUST.JCL(CRA$VMSG)

Customize CRASRV.properties

RSE server uses the settings in /etc/rdz/CRASRV.properties to start and connect to a CARMA server. You can edit the file with the TSO **OEDIT** command. Note that the RSED started task must be restarted before changes are in effect.

When using default file locations, the only required change is changing the value of the clist.dsname directive to FEK.#CUST.CNTL(CRASUBCA). Refer to "CRASRV.properties, RSE interface to CARMA" on page 60 for more information on the different directives.

```
port.start=5227
port.range=100
startup.script.name=/usr/lpp/rdz/bin/carma.startup.rex
clist.dsname='FEK.#CUST.CNTL(CRASUBCA)'
```

Figure 15. CRASRV.properties - batch submit with CA Endevor® SCM RAM

Customize CRASUBCA

The FEK.#CUST.CNTL(CRASUBCA) CLIST and embedded JCL submits a CARMA server. Note that changes are in effect for all CARMA servers started after the update.

Refer to the documentation within the member for customization instructions. Refer to "CRASUB*, batch submit server startup" on page 65 for more information on the batch submit startup method.

```
PROC 1 PORT TIMEOUT(420)
SUBMIT * END($$)
//CRA&PORT JOB CLASS=A,MSGCLASS=A,MSGLEVEL=(1,1)
//*
//RUN      EXEC PGM=IKJEFT01,DYNAMNBR=125,REGION=0M,TIME=NOLIMIT
//*
//* DD used by RAM
//TYPEMAP  DD DISP=SHR,DSN=FEK.#CUST.PARMLIB(CRATMAP)
//SHOWVIEW DD DISP=SHR,DSN=FEK.#CUST.PARMLIB(CRASHOW)
//* uncomment CRABCFG and CRABSKEL to use batch actions
//*CRABCFG  DD DISP=SHR,DSN=FEK.#CUST.PARMLIB(CRABCFG)
//*CRABSKEL DD DISP=SHR,DSN=FEK.#CUST.CNTL
//CONLIB   DD DISP=SHR,DSN=CA.NDVR.CONLIB                     * NDVR R12
//*CONLIB   DD DISP=SHR,DSN=CA.NDVR.CSIQLOAD                  * NDVR R14
//JCLOUT   DD SYSOUT=(A,INTRDR),DCB=(LRECL=80,RECFM=F,BLKSIZE=80)
//EXT1ELM  DD DISP=(NEW,DELETE),UNIT=SYSALLDA,
//            RECFM=VB,LRECL=4096,BLKSIZE=27998,SPACE=(TRK,(5,5))
//EXT2ELM  DD DISP=(NEW,DELETE),UNIT=SYSALLDA,
//            RECFM=VB,LRECL=4096,BLKSIZE=27998,SPACE=(TRK,(5,5))
//EXT1DEP  DD DISP=(NEW,DELETE),UNIT=SYSALLDA,
//            RECFM=VB,LRECL=4096,BLKSIZE=27998,SPACE=(TRK,(5,5))
//C1MSGS1  DD SYSOUT(H)
//C1EXMSGS DD SYSOUT(H)
//MSG3FILE DD DUMMY
//*
//* DD used by CARMA server (CRASERV)
//STEPLIB  DD DISP=SHR,DSN=FEK.SFEKLOAD
//         DD DISP=SHR,DSN=CA.NDVR.AUTHLIB                    * NDVR R12
//         DD DISP=SHR,DSN=CA.NDVRU.AUTHLIB                   * NDVR R12
//*         DD DISP=SHR,DSN=CA.NDVR.CSIQAUTH                  * NDVR R14
//*         DD DISP=SHR,DSN=CA.NDVR.CSIQAUTU                  * NDVR R14
//CRADEF   DD DISP=SHR,DSN=FEK.#CUST.CRADEF
//CRAMSG   DD DISP=SHR,DSN=FEK.#CUST.CRAMSG
//CRASTRS  DD DISP=SHR,DSN=FEK.#CUST.CRASTRS
//CARMALOG DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//*
//* DD used by ISPF (via NDVRC1)
//ISPPROF  DD DISP=(NEW,DELETE,DELETE),UNIT=SYSALLDA,
//            LRECL=80,RECFM=FB,SPACE=(TRK,(1,1,5))
//ISPCTL0  DD DISP=(NEW,DELETE,DELETE),UNIT=SYSALLDA,
//            LRECL=80,RECFM=FB,SPACE=(TRK,(5,5))
//ISPCTL1  DD DISP=(NEW,DELETE,DELETE),UNIT=SYSALLDA,
//            LRECL=80,RECFM=FB,SPACE=(TRK,(5,5))
//ISPMLIB  DD DISP=SHR,DSN=ISP.SISPMENU
//ISPPLIB  DD DISP=SHR,DSN=ISP.SISPPENU
//ISPSLIB  DD DISP=SHR,DSN=ISP.SISPSENU
//ISPTLIB  DD DISP=SHR,DSN=ISP.SISPTENU
//*
//* DD used by TSO (IKJEFT01)
//SYSPROC  DD DISP=SHR,DSN=FEK.SFEKPROC                       * CRANDVRA
//SYSTSPRT DD SYSOUT=*
//SYSTSIN  DD *
%CRANDVRA NDVRC1 PGM(CRASERV) PARM(&PORT &TIMEOUT)
$$
EXIT CODE(0)
```

Figure 16. CRASUBCA - batch submit with CA Endevor® SCM RAM

(Optional) Additional CA Endevor® SCM RAM customization

The CA Endevor® SCM RAM has additional components that can be customized if you want to customize them.

- The CA Endevor® SCM RAM has two configuration files, `FEK.#CUST.PARMLIB(CRASHOW)` and `FEK.#CUST.PARMLIB(CRATMAP)`, that can be customized. Refer to "CRASHOW and CRATMAP, CA Endevor® SCM RAM configuration files" on page 69 for more information.
- The CA Endevor® SCM RAM has an allocation exec, `FEK.SFEKPROC(CRANDVRA)`, that can be customized. Refer to "CRANDVRA, CA Endevor® SCM RAM allocation exec" on page 70 for more information.
- The CA Endevor® SCM RAM supports doing CA Endevor® SCM actions in batch mode. Batch-actions requires a configuration file, `FEK.#CUST.PARMLIB(CRABCFG)`, and a skeleton JCL, `FEK.#CUST.CNTL(CRABATCA)`, that must be customized. See "CA Endevor® SCM RAM batch actions" on page 70 for more information.

Batch submit with sample RAMs

The information in this section describes how to set up CARMA with the following specifications:

- Server startup: batch submit method (requires JES initiators)
- RAM: sample RAMs (not for production purposes)

This customization step can be bypassed if you want to use one of the other scenarios with different specifications.

Create VSAM data sets

Customize and submit the following JCL jobs to define and populate the CARMA related VSAM data sets. Refer to the documentation within the member for customization instructions. Note that existing VSAM data sets will be replaced if present.

Refer to "CARMA VSAM data sets" on page 66 and "CARMA Repository Access Managers (RAMs)" on page 67 for more details on this step.

CARMA

- `FEK.#CUST.JCL(CRA$VDEF)`
- `FEK.#CUST.JCL(CRA$VMSG)`
- `FEK.#CUST.JCL(CRA$VSTR)`

Sample RAMs

- `FEK.#CUST.JCL(CRA#VPDS)`

Customize CRASRV.properties

RSE server uses the settings in `/etc/rdz/CRASRV.properties` to start and connect to a CARMA server. You can edit the file with the TSO **OEDIT** command. Note that the RSED started task must be restarted before changes are in effect.

As this is the default scenario for Developer for System z, there are no changes required when starting from an unmodified copy of the file. Refer to "CRASRV.properties, RSE interface to CARMA" on page 60 for more information on the different directives.

```
port.start=5227
port.range=100
startup.script.name=/usr/lpp/rdz/bin/carma.startup.rex
clist.dsname='FEK.#CUST.CNTL(CRASUBMT)'
```

Figure 17. CRASRV.properties - batch submit with sample RAMs

Customize CRASUBMT

The FEK.#CUST.CNTL(CRASUBMT) CLIST and embedded JCL submits a CARMA server. Note that changes are in effect for all CARMA servers started after the update.

Refer to the documentation within the member for customization instructions. Refer to “CRASUB*, batch submit server startup” on page 65 for more information on the batch submit startup method.

```
PROC 1 PORT TIMEOUT(420)
SUBMIT * END($$)
//CRA&PORT JOB CLASS=A,MSGCLASS=A,MSGLEVEL=(1,1)
//*
//RUN      EXEC PGM=IKJEFT01,DYNAMNBR=125,REGION=0M,TIME=NOLIMIT
//*
//* DD used by RAM
//CRARAM1  DD DISP=SHR,DSN=FEK.#CUST.CRARAM1            * PDS RAM
//*CRARAM2  DD DISP=SHR,DSN=FEK.#CUST.CRARAM2          * SCLM RAM
//*
//* DD used by CARMA server (CRASERV)
//STEPLIB  DD DISP=SHR,DSN=FEK.SFEKLOAD
//CRADEF   DD DISP=SHR,DSN=FEK.#CUST.CRADEF
//CRAMSG   DD DISP=SHR,DSN=FEK.#CUST.CRAMSG
//CRASTRS  DD DISP=SHR,DSN=FEK.#CUST.CRASTRS
//CARMALOG DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//*
//* DD used by ISPF (ISPSTART)
//ISPPROF  DD DISP=(NEW,DELETE,DELETE),UNIT=SYSALLDA,
//            LRECL=80,RECFM=FB,SPACE=(TRK,(1,1,5))
//ISPCTL0  DD DISP=(NEW,DELETE,DELETE),UNIT=SYSALLDA,
//            LRECL=80,RECFM=FB,SPACE=(TRK,(5,5))
//ISPCTL1  DD DISP=(NEW,DELETE,DELETE),UNIT=SYSALLDA,
//            LRECL=80,RECFM=FB,SPACE=(TRK,(5,5))
//ISPMLIB  DD DISP=SHR,DSN=ISP.SISPMENU
//ISPPLIB  DD DISP=SHR,DSN=ISP.SISPPENU
//ISPSLIB  DD DISP=SHR,DSN=ISP.SISPSENU
//ISPTLIB  DD DISP=SHR,DSN=ISP.SISPTENU
//*
//* DD used by TSO (IKJEFT01)
//SYSTSPRT DD SYSOUT=*
//SYSTSIN  DD *
ISPSTART PGM(CRASERV) PARM(&PORT &TIMEOUT)
$$
EXIT CODE(0)
```

Figure 18. CRASUBMT - batch submit with sample RAMs

CARMA configuration details

The different configuration scenarios that are documented in this publication share many of the CARMA configuration files. The details of these configuration files are documented here, and they are referenced from within the various scenarios.

CRASRV.properties, RSE interface to CARMA

The CARMA server provides a standard API for other host-based products to access one or more Software Configuration Managers (SCMs). However, it does not provide methods for direct communication with a client PC. For this, it relies on other products, such as the RSE server. The RSE server uses the settings in `CRASRV.properties` to start and connect to a CARMA server.

`CRASRV.properties` is located in `/etc/rdz/`, unless you specified a different location when you customized and submitted job `FEK.SFEKSAMP(FEKSETUP)`. See "Customization setup" on page 13 for more details. You can edit the file with the TSO **OEDIT** command.

Note: The RSED started task must be restarted before changes are in effect.

```
# CRASRV.properties - CARMA configuration options
#
port.start=0
port.range=100
startup.script.name=/usr/lpp/rdz/bin/carma.startup.rex
clist.dsname='FEK.#CUST.CNTL(CRASUBMT)'
crastart.stub=/usr/lpp/rdz/bin/CRASTART
crastart.configuration.file=/etc/rdz/crastart.conf
crastart.syslog=Partial
crastart.timeout=420
#crastart.steplib=FEK.SFEKLPA
#crastart.tasklib=TASKLIB
```

Figure 19. CRASRV.properties – CARMA configuration file

port.start

When the value of `port.start` is `0` (zero), CARMA will use an ephemeral port for communication between CARMA and the RSE server. In this scenario, TCP/IP will assign a random free port number. When the value of `port.start` is non-zero, it is interpreted as the starting point of a port range used for communication between CARMA and the RSE server. The default port is `0`. Communication on this port is confined to your host machine.

Note: Before selecting a port, verify that the port is available on your system with the **NETSTAT** and **NETSTAT PORTL** commands. See "Reserved TCP/IP ports" in the *Host Configuration Reference* (SC14-7290) for more information.

port.range

Range of ports, starting at `port.start`, which will be used for CARMA communication if `port.start` is non-zero. The default is `100`. For example, when `port.start` is 5227 and `port.range` is 100, port 5227 until 5326 (inclusive) can be used by CARMA.

startup.script.name

Defines the absolute path of the CARMA startup script. The default is `/usr/lpp/rdz/bin/carma.startup.rex`. This REXX exec will trigger the startup of a CARMA server.

clist.dsname

Defines the startup method for the CARMA server. Refer to "Select server startup method and active RAM" on page 50 for more details about the different startup methods.

- *CRASTART indicates that the CARMA server should be started as a subtask within RSE using CRASTART. If you specify *CRASTART, you must also specify the crastart.* directives.
- *ISPF indicates that the CARMA server should be started using ISPF's TSO/ISPF Client Gateway. This startup method is deprecated.
- Any other value defines the location of the CRASUBMT CLIST, using TSO-like naming conventions. With quotes (') the data set name is an absolute reference, without quotes (') the data set name is prefixed with the client's user ID, not the TSO prefix. The latter requires that all CARMA users must maintain their own CRASUBMT CLIST.

The default is FEK.#CUST.CNTL(CRASUBMT). This CLIST will use the batch submit method to start a CARMA server that supports the sample RAMs.

The default is 'FEK.#CUST.CNTL(CRASUBMT)'. This CLIST will start a CARMA server when opening a connection using the batch submit method.

crastart.stub
z/OS UNIX stub for calling CRASTART. The default is /usr/lpp/rdz/bin/CRASTART. This stub makes the MVS based CRASTART load module available to z/OS UNIX processes. This directive is only used if the clist.dsname directive has *CRASTART as value.

crastart.configuration.file
Specifies the name of the CRASTART configuration file. The default is /etc/rdz/crastart.conf. This file specifies the data set allocations and program invocations needed to start a CARMA server. This directive is only used if the clist.dsname directive has *CRASTART as value.

crastart.syslog
Specifies how much information is written to the system log while CRASTART starts a CARMA server. The default is Partial. Valid values are:

A (All)	All tracing information is printed to SYSLOG
P (Partial)	Only connect, disconnect, and error information is printed to SYSLOG
anything else	Only error conditions are printed to SYSLOG

This directive is only used if the clist.dsname directive has *CRASTART as value.

crastart.timeout
The length of time, in seconds, before a CARMA server ends due to lack of activity. The default is 420 (7 minutes). This directive is only used if the clist.dsname directive has *CRASTART as value.

Note: System abend 522 for module CRASERV will occur if the JWT parameter in the SMFPRMxx parmlib member is set to a value lower than the crastart.timeout value in CRASRV.properties. This does not impact CARMA operations, because the server is restarted automatically if needed.

crastart.steplib
The location of the CRASTART module when accessed through the STEPLIB directive in rsed.envvars. The default is FEK.SFEKLPA. Uncomment and customize this directive if the CRASTART module cannot be part of LPA or LINKLIST. Note that program control and APF issues

might arise if the CRASTART module is not in LPA. This directive is only used if the `clist.dsname` directive has `*CRASTART` as value.

crastart.tasklib

Alternate name for the TASKLIB DD name in `crastart.conf`. The default is `TASKLIB`. Uncomment and customize this directive if DD name TASKLIB has a special meaning for your SCM or RAM and cannot be used as STEPLIB replacement. This directive is only used if the `clist.dsname` directive has `*CRASTART` as value.

crastart*.conf, CRASTART server startup

RSE invokes load module `CRASTART`, which uses the definitions in `crastart*.conf` to create a valid environment to execute batch TSO and ISPF commands. Developer for System z uses this environment to run the CARMA server, `CRASERV`.

`crastart*.conf` is located in `/etc/rdz/`, unless you specified a different location when you customized and submitted job `FEK.SFEKSAMP(FEKSETUP)`. See "Customization setup" on page 13 for more details. You can edit the file with the TSO **OEDIT** command.

Note: Changes are in effect for all CARMA servers started after the update.

Developer for System z provides multiple `crastart*.conf` configuration files. Each of these sample files is preconfigured for a specific customization scenario:

- `crastart.endevor.conf` is configured for CRASTART startup with CA Endevor® SCM RAM.
- `crastart.conf` is configured for CRASTART startup with sample RAMs.

The function of the `crastart*.conf` file is similar in concept to a JCL job stream, but is more restrictive.

- The following samples show valid line formats:

```
– * comment
– ddname=dsn1,dsn2,dsn3          * comment
– ddname=SYSOUT(c)               * comment
– ddname=DUMMY                   * comment
– -COMMAND=<any bpxwdyn command> * comment
– PROGRAM = progname parms       * comment
```

  Note: The **BPXWDYN** command is documented in *Using REXX and z/OS UNIX System Services* (SA22-7806) and allows complex allocation constructs.
- All input is translated to upper case.
- Line continuations are not supported.
- There is no limitation on line length.
- One ore more blank spaces ( ) are allowed around the equal sign (=).
- DD allocations must precede the related `PROGRAM` statement.
- DD names allocated here are freed at the end of program execution (they do not accumulate).
- DD names allocated by the invoked programs are NOT freed.
- Multiple data sets can be concatenated to a DD name. The data set names must be separated by a comma (,), and the concatenation will be searched in the listed order.

- All data set allocations are done with DISP=SHR, except for allocations done using -COMMAND.
- Inline data is not supported, all data must be in cataloged files.
- Variables can only be used on the right side of the equal sign (=)
- The following variables are supported:

&CRAUSER.	client user ID
&CRADATE.	current date in Dyyyyddd format (7 char Julian)
&CRATIME.	current time in Thhmmss format (hour min sec)
&CRAPRM1.	port number
&CRAPRM2.	timeout value
System symbol	any SYS1.PARMLIB(IEASYMxx) system symbol
-<ddname>	A dash (-) followed by a previously defined DD name acts like a *.ddname backward reference in JCL. The original DD must be allocated using the –COMMAND statement.

Note: There is no variable for the TSO prefix, because TSO is not active when the configuration file is interpreted. Refer to "(Optional) Custom allocation exec" on page 75 if you have a need for the TSO prefix or other variable that is not available.

Figure 20 shows a basic crastart*.conf skeleton that includes ISPF services.

```
* DD used by RAM

* DD used by CARMA server (CRASERV)
TASKLIB = FEK.SFEKLOAD
CRADEF  = FEK.#CUST.CRADEF
CRAMSG  = FEK.#CUST.CRAMSG
CRASTRS = FEK.#CUST.CRASTRS
CARMALOG= SYSOUT(H)
SYSPRINT= SYSOUT(H)

* DD used by ISPF (ISPSTART)
-COMMAND=ALLOC FI(ISPCTL0) NEW DELETE DSORG(PS) RECFM(F,B) LRECL(80)
  BLKSIZE(32720) SPACE(5,5) TRACKS UNIT(SYSALLDA)
-COMMAND=ALLOC FI(ISPCTL1) NEW DELETE DSORG(PS) RECFM(F,B) LRECL(80)
  BLKSIZE(32720) SPACE(5,5) TRACKS UNIT(SYSALLDA)
-COMMAND=ALLOC FI(ISPPROF) NEW DELETE DSORG(PO) RECFM(F,B) LRECL(80)
  BLKSIZE(32720) SPACE(5,5) TRACKS UNIT(SYSALLDA) DIR(5)
ISPTABL = -ISPPROF
ISPTLIB = -ISPPROF,ISP.SISPTENU
ISPMLIB = ISP.SISPMENU
ISPPLIB = ISP.SISPPENU
ISPSLIB = ISP.SISPSENU

* DD used by TSO (IKJEFT01)
SYSTSIN = DUMMY
SYSTSPRT= SYSOUT(H)

PROGRAM=IKJEFT01 ISPSTART PGM(CRASERV) PARM(&CRAPRM1. &CRAPRM2.)
```

Figure 20. crastart.conf - CARMA server startup using CRASTART*

Note:

- Due to page width limitations, some lines in the sample wrapped onto the next line. All lines that start with an indentation should be added to the end of the previous line.
- You can add your own DD statements and data set concatenations to customize the CARMA TSO environment, thus mimicking a TSO logon procedure.
- DD name TASKLIB acts like STEPLIB in JCL. Its DD name must match the value specified for crastart.tasklib in CRASRV.properties, which is described in "CRASRV.properties, RSE interface to CARMA" on page 60.
- Regular APF rules apply for TASKLIB allocations. Libraries lose their APF authorization when a non-APF authorized library is part of the concatenation.
- System abend 522 for module CRASERV will occur if the JWT parameter in the SMFPRMxx parmlib member is set to a value lower than the crastart.timeout value in CRASRV.properties. This does not impact CARMA operations, because the server is restarted automatically if needed.
- Details of the CARMA server startup are shown in rsecomm.log (when the server ends). Refer to "(Optional) rsecomm.properties, RSE tracing" on page 98 for more information on setting the detail level of rsecomm.log.

Collecting CRASTART log files

CRASTART creates a TSO environment as a child process of RSE, which will run in a separate address space. Non-trivial actions might be needed to keep the CARMA output sent to SYSOUT(*), which complicates collecting log files. This can be resolved by writing the log files to a (user-specific) data set, as shown in the following sample allocation:

```
-COMMAND=ALLOC FI(CARMALOG) MOD CATALOG DSORG(PS) RECFM(F,B) LRECL(133)
  BLKSIZE(27930) SPACE(5,5) TRACKS UNIT(SYSALLDA)
  DA(&CRAUSER..&SYSNAME..CRA.CARMALOG)
```

Note:

- Due to page width limitations, some lines in the sample wrapped onto the next line. All lines that start with an indentation should be added to the end of the previous line.
- This log file must be allocated using the -COMMAND statement to be able to create user-specific log files.
- You can also allocate the log data sets in an allocation exec if you need more flexibility (for example, only send the log to a data set for specific users). Refer to "(Optional) Custom allocation exec" on page 75 for more information on allocation execs.

When writing log files to SYSOUT, you should be aware that SYSOUT allocated by z/OS UNIX processes is treated as special output in JES. This is similar to SYSOUT allocated by APPC transactions.

- While the CARMA server is still active, the output can be seen using the **DA** command in SDSF. The job will have the user's user ID followed by a random one-digit number as job name and an STC job ID. The user will be the job owner.
- When the CARMA server ends, due to inactivity or the user ending the connection, the output can be seen using the **APPC ON** and **H ALL** commands in SDSF, if the output was written to a HOLD output class. The job name, job ID, and job owner remain the same. Each DD will show up as a separate spool file, without any indication which DD it is.

- JES Job Monitor is also able to show the output if SEARCHALL=ON is active in FEJJCNFG and the output resides on the spool in a HOLD output class. See "FEJJCNFG, JES Job Monitor configuration file" on page 25 for more information on the SEARCHALL directive.

CRASUB*, batch submit server startup

RSE invokes CLIST CRASUB*, which in turn submits an embedded JCL to create a valid environment to execute batch TSO and ISPF commands. Developer for System z uses this environment to run the CARMA server, CRASERV.

CRASUB* is located in FEK.#CUST.CNTL, unless you specified a different location when you customized and submitted job FEK.SFEKSAMP(FEKSETUP). See "Customization setup" on page 13 for more details.

Note: Changes are in effect for all CARMA servers started after the update.

Developer for System z provides multiple CRASUB* JCL jobs. Each of these sample files is pre-configured for a specific customization scenario:

- CRASUBCA is configured for batch startup with CA Endevor® SCM RAM
- CRASUBMT is configured for batch startup with sample RAMs

Figure 21 shows a basic CRASUB* skeleton that includes ISPF services.

```
PROC 1 PORT TIMEOUT(420)
SUBMIT * END($$)
//CRA&PORT JOB CLASS=A,MSGCLASS=A,MSGLEVEL=(1,1)
//*
//RUN      EXEC PGM=IKJEFT01,DYNAMNBR=125,REGION=0M,TIME=NOLIMIT
//*
//* DD used by RAM
//*
//* DD used by CARMA server (CRASERV)
//STEPLIB  DD DISP=SHR,DSN=FEK.SFEKLOAD
//CRADEF   DD DISP=SHR,DSN=FEK.#CUST.CRADEF
//CRAMSG   DD DISP=SHR,DSN=FEK.#CUST.CRAMSG
//CRASTRS  DD DISP=SHR,DSN=FEK.#CUST.CRASTRS
//CARMALOG DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//*
//* DD used by ISPF (ISPSTART)
//ISPPROF  DD DISP=(NEW,DELETE,DELETE),UNIT=SYSALLDA,
//            LRECL=80,RECFM=FB,SPACE=(TRK,(1,1,5))
//ISPCTL0  DD DISP=(NEW,DELETE,DELETE),UNIT=SYSALLDA,
//            LRECL=80,RECFM=FB,SPACE=(TRK,(5,5))
//ISPCTL1  DD DISP=(NEW,DELETE,DELETE),UNIT=SYSALLDA,
//            LRECL=80,RECFM=FB,SPACE=(TRK,(5,5))
//ISPMLIB  DD DISP=SHR,DSN=ISP.SISPMENU
//ISPPLIB  DD DISP=SHR,DSN=ISP.SISPPENU
//ISPSLIB  DD DISP=SHR,DSN=ISP.SISPSENU
//ISPTLIB  DD DISP=SHR,DSN=ISP.SISPTENU
//*
//* DD used by TSO (IKJEFT01)
//SYSTSPRT DD SYSOUT=*
//SYSTSIN  DD *
ISPSTART PGM(CRASERV) PARM(&PORT &TIMEOUT)
$$
EXIT CODE(0)
```

Figure 21. CRASUB - CARMA startup using batch submit*

Note:

- You can add your own DD statements and data set concatenations to customize the CARMA TSO environment, thus mimicking a TSO logon procedure.
- You can optionally change CARMA's timeout value by modifying the `PROC 1 PORT TIMEOUT(420)` line in the `CRASUB*` CLIST. The timeout value is the number of seconds CARMA will wait for the next command from the client. Setting a value of 0 results in the default timeout value, currently 420 seconds (7 minutes).
- Details of the CARMA startup process are shown in `rsecomm.log` (when the server ends). Refer to "(Optional) rsecomm.properties, RSE tracing" on page 98 for more information on setting the detail level of `rsecomm.log`.

CARMA VSAM data sets

The CARMA server requires READ access to three VSAM data sets. The sample members to create and populate these VSAM data sets are located in `FEK.#CUST.JCL`, unless you specified a different location when you customized and submitted job `FEK.SFEKSAMP(FEKSETUP)`. See "Customization setup" on page 13 for more details.

Note:

- Refer to sample job `FEK.#CUST.JCL(CRA#UADD)` if you need to merge the definitions for a (custom) RAM into an existing VSAM configuration. This job must be customized and submitted for each CARMA VSAM file that changes. Refer to the *Common Access Repository Manager Developer's Guide* (SC23-7660) for more information on the record structure used by the different CARMA VSAM files.
- Use sample job `FEK.#CUST.JCL(CRA#UQRY)` to extract the active definitions from a VSAM to a sequential data set.

CRADEF, configuration data set

This VSAM data set describes the functions supported by the defined RAMs. Note that RAM developers require UPDATE access to this data set. The data set can be created by one of these sample jobs:

- `CRA$VCAD` populates the data set with CA Endevor® SCM RAM data.
- `CRA$VDEF` populates the data set with sample RAM data.

CRAMSG, message data set

This VSAM data set holds messages issued by the CARMA server itself. The data set can be created by one of these sample jobs:

- `CRA$VMSG` populates the data set with generic server data.

CRASTRS, custom string data set

This VSAM data set holds the messages issued by the defined RAMs. Note that RAM developers require UPDATE access to this data set. The data set can be created by one of these sample jobs:

- `CRA$VCAS` populates the data set with CA Endevor® SCM RAM data.
- `CRA$VSTR` populates the data set with sample RAM data.

CARMA VSAM migration notes

1. Beginning with version 7.6.1, Developer for System z supports a new data structure layout for the CARMA custom information VSAM data set, `CRASTRS`, to remove message length limitations.

   Prior to Developer for System z version 7.6.1, strings defined in the CARMA custom information VSAM data set are limited to predefined lengths. This limitation forces RAM developers to shorten descriptive strings, or to use client-side plug-ins to display full-length strings.

The new VSAM record structure supports a variable-length data structure layout for the CARMA custom information VSAM data set, CRASTRS, where strings are separated by a delimiter character instead of being fixed length.

Customize and submit the FEK.SFEKSAMP(CRA#VS2) JCL to convert your existing, fixed-length, CARMA custom information VSAM data set, CRASTRS, to the new variable-length format.

Note:

- Beginning with version 7.6.1, the sample CARMA custom information VSAM data set is shipped in variable-length format.
- Beginning with version 7.6.1, the CARMA load module, CRASERV, supports both the fixed-length format and the variable-length format for the CARMA custom information VSAM data set.
- Older versions of the CARMA load module do not support the variable-length format and will produce garbled strings when used with a variable-length CARMA custom information VSAM data set.

2. The introduction of the CA Endevor® SCM packages RAM removes the need for providing package-related actions in the CA Endevor® SCM element RAM. Therefore, providing package-related actions in the CA Endevor® SCM RAM is deprecated, and the CARMA VSAM data sets in version 8.5 no longer hold the definitions required for this functionality.

   You can restore this functionality by customizing and submitting the FEK.SFEKSAMP(CRA#UADD) JCL to merge the removed items back into the CARMA VSAM data sets.

   - Merge FEK.SFEKVSM2(CRA0VPKD) into FEK.#CUST.CRADEF.
   - Merge FEK.SFEKVSM2(CRA0VPKS) into FEK.#CUST.CRASTRS.

   Note that this merge action is required each time the VSAM data sets are updated during product maintenance.

CARMA Repository Access Managers (RAMs)

A Repository Access Manager (RAM) is an Application Programming Interface (API) for a z/OS based Software Configuration Manager (SCM). In turn, Developer for System z (or user-written applications) can start a CARMA server which loads the RAMs and provides a standard interface to access any supported SCM.

The CARMA server must be able to find the RAM load modules, either through LINKLIST or STEPLIB/TASKLIB.

The CRAR* RAM load modules provided by Developer for System z are located in FEK.SFEKLOAD, and the sample source code and compile jobs are located in FEK.SFEKSAMP, unless you used a different high level qualifier during the SMP/E install of Developer for System z.

The following sections have customization notes for the RAMs that are available with Developer for System z. The referenced sample members are located in FEK.#CUST.*, unless you specified a different location when you customized and submitted job FEK.SFEKSAMP(FEKSETUP). See “Customization setup” on page 13 for more details.

See *Common Access Repository Manager Developer's Guide* (SC23-7660) for in-depth knowledge of CARMA and for more information on the sample RAMs and sample source code provided.

CA Endevor® SCM RAM

- Production-type RAM.
- The CA Endevor® SCM RAM gives Developer for System z clients direct access to CA Endevor® SCM elements.
- The load module name is `CRARNDVR`.
- The CA Endevor® SCM RAM has many additional settings compared to a vanilla CARMA setup. You should use one of the preconfigured setups that support the CA Endevor® SCM RAM as starting point, and customize it further to fit your needs.
- The (deprecated) TSO/ISPF Client Gateway startup method cannot be used together with the CA Endevor® SCM RAM.
- The CA Endevor® SCM RAM has two configuration files, `FEK.#CUST.PARMLIB(CRASHOW)` and `FEK.#CUST.PARMLIB(CRATMAP)`, that can be customized. Refer to "CRASHOW and CRATMAP, CA Endevor® SCM RAM configuration files" on page 69 for more information.
- The CA Endevor® SCM RAM has an allocation exec, `FEK.SFEKPROC(CRANDVRA)`, that can be customized. Refer to "CRANDVRA, CA Endevor® SCM RAM allocation exec" on page 70 for more information.
- The CA Endevor® SCM RAM supports doing CA Endevor® SCM actions in batch (background) mode. Batch-actions requires a configuration file, `FEK.#CUST.PARMLIB(CRABCFG)`, and a skeleton JCL, `FEK.#CUST.CNTL(CRABATCA)`, that must be customized. See "CA Endevor® SCM RAM batch actions" on page 70 for more information.

CA Endevor® SCM packages RAM

- Production-type RAM.
- The CA Endevor® SCM packages RAM gives Developer for System z clients direct access to CA Endevor® SCM packages.
- The load module name is `CRARPKGS`.
- The CA Endevor® SCM packages RAM has no customizable settings, and must be used in combination with the CA Endevor® SCM RAM.

PDS RAM

- Sample RAM, do not use in a production environment.
- The PDS RAM gives a data set list similar to **MVS Files -> My Data Sets** in the Remote Systems view.
- The load module name is `CRARPDS`.
- The PDS RAM expects that ISPF services are available.
- The PDS RAM requires an additional VSAM data set to be allocated to DD `CRARAM1`. This VSAM data set can be allocated and primed with sample job `FEK.#CUST.JCL(CRA#VPDS)`. Refer to the documentation within the member for customization instructions.
- Source code and compile jobs are available in `FEK.SFEKSAMP`. See *Common Access Repository Manager Developer's Guide* (SC23-7660) for more information.

Skeleton RAM

- Sample RAM, do not use in a production environment.

- The skeleton RAM gives a functional framework that can be used as starting point to develop your own RAM.
- The load module name is CRARTEST.
- Source code and compile jobs are available in FEK.SFEKSAMP. See *Common Access Repository Manager Developer's Guide* (SC23-7660) for more information.

SCLM RAM

- Sample RAM, do not use in a production environment.
- The SCLM RAM gives a basic entry into SCLM, ISPF's Software Configuration Manager. This RAM is not enabled by default.
- The load module name is CRARSCLM.
- The SCLM RAM expects that ISPF services are available.
- The SCLM RAM requires an additional VSAM data set to be allocated to DD CRARAM2. This VSAM data set can be allocated and primed with sample job FEK.#CUST.JCL(CRA#VSLM). Refer to the documentation within the member for customization instructions.
- The SCLM RAM expects that various user-specific data sets exist. Customize FEK.#CUST.JCL(CRA#ASLM) to allocate these data sets. Refer to the documentation within the member for customization instructions. Each user must submit CRA#ASLM once before using CARMA with the SCLM RAM. Failing to do so will result in an allocation error.
- The SCLM RAM is not enabled by default. To enable the RAM, it must be defined in the CARMA VSAM data sets referenced by DD CRADEF and CRASTRS. Use sample job FEK.#CUST.JCL(CRA#UADD) to merge FEK.SFEKVSM2(CRA0SLMD) into CRADEF and FEK.SFEKVSM2(CRA0SLMS) into CRASTRS. Refer to the documentation within the member for customization instructions.
- Source code and compile jobs are available in FEK.SFEKSAMP. See *Common Access Repository Manager Developer's Guide* (SC23-7660) for more information.

CRASHOW and CRATMAP, CA Endevor® SCM RAM configuration files

The following CA Endevor® SCM RAM specific CARMA components can be customized, regardless of the chosen server startup method. The sample members referenced below are located in FEK.#CUST.PARMLIB, unless you specified a different location when you customized and submitted job FEK.SFEKSAMP(FEKSETUP). See "Customization setup" on page 13 for more details.

CRASHOW, CA Endevor® SCM RAM default filters

CRASHOW defines default filters for CA Endevor® SCM environments, systems, and so forth. Refer to the documentation within the member for customization instructions if you want to change the defaults.

```
ENV=*
TOENV=
STGID=*
TOSTGID=
SYS=*
SUBSYS=*
ELEM=*
TOELEM=
TYPE=*
#FILTER-DEP=YES
```

Figure 22. CRASHOW - CA Endevor® SCM RAM default filters

Note: FILTER-DEP is not a common CA Endevor® SCM variable, but a Developer for System z specific variable that controls dependency scans for elements with footprint references to other CA Endevor® SCM repository locations.

CRATMAP, CA Endevor® SCM RAM file extension mappings

CRATMAP overrides CA Endevor® SCM type to file extension mappings. Refer to the documentation within the member for customization instructions if you want to change the defaults.

```
# *       = cbl
# COBOL   = cbl
# COPY    = cpy
# ASM     = asm
# MACRO   = asm
# PROCESS = jcl
```

Figure 23. CRATMAP - CA Endevor® SCM RAM default filters

CRANDVRA, CA Endevor® SCM RAM allocation exec

Both the batch submit and the CRASTART startup method invoke REXX exec CRANDVRA to allocate user-specific data sets used by CA Endevor® SCM RAM. The allocations are done in a separate exec, as an exec allows more flexibility than what is possible within the batch submit CRASUBCA JCL and CRASTART crastart.endevor.conf configuration file.

DD	Data set name	Type
DEPEND	&SYSPREF..&SYSUID..&SYSNAME..CRA$NDVR.DEPEND	Permanent
BROWSE	&SYSPREF..&SYSUID..&SYSNAME..CRA$NDVR.BROWSE	Temporary
C1PRINT	&SYSPREF..&SYSUID..&SYSNAME..CRA$NDVR.LISTING	Temporary
SPCLLIST	&SYSPREF..&SYSUID..&SYSNAME..CRA$NDVR.SPCLLIST	Temporary
PKGSCLS	&SYSPREF..&SYSUID..&SYSNAME..CRA$NDVR.PKGSCLS	Temporary

You can customize a copy of this allocation REXX exec if certain defaults, such as the data set name, do not match your site standards. CRANDVRA is located in FEK.SFEKPROC, unless you used a different high level qualifier during the SMP/E install of Developer for System z.

Refer to the documentation within the member for customization instructions. Refer to "(Optional) Custom allocation exec" on page 75 for more information on allocation execs.

Note: You should copy the sample allocation REXX to a new data set and customize this copy to avoid overwriting it when applying maintenance. When you do this, you must update the reference to SFEKPROC in the SYSEXEC DD of your chosen CARMA startup method to match your new data set name.

CA Endevor® SCM RAM batch actions

Normally, CA Endevor® SCM actions like "Generate Element" are executed "online", in the CARMA server address space. This causes problems if your CA

Endevor® SCM procedures invoke TSO, because TSO is already active and that means required DDs such as SYSTSIN and SYSTSPRT are in use.

To resolve this problem, the CA Endevor® SCM RAM supports "batch actions" since version 8.0.3. When batch-actions is enabled, the CA Endevor® SCM RAM will submit a (customizable) batch job to perform actions like "Generate Element". This allows the allocation of DDs like SYSTSIN and SYSTSPRT by your CA Endevor® SCM procedures, because the submitted JCL does not require TSO to be active.

Note that CA Endevor® SCM RAM batch-actions are the Developer for System z equivalent of background CA Endevor® SCM actions.

When a request is issued to execute an action that is supported by batch-actions, the CA Endevor® SCM RAM will check for the existence of the CRABCFG DD (in CRASUBCA or crastart.endevor.conf) and will check that the setup behind this DD is valid. If CRABCFG is there and the setup is valid, the action will be performed in batch. If CRABCFG is not there, the action will be performed online. Note that version 8.0.3 or higher clients can override this behavior.

For example:

```
//* uncomment CRABCFG and CRABSKEL to use batch actions
//*CRABCFG  DD DISP=SHR,DSN=FEK.#CUST.PARMLIB(CRABCFG)
//*CRABSKEL DD DISP=SHR,DSN=FEK.#CUST.CNTL
```

Note:

- The TSO-free environment is only available for selected CA Endevor® SCM actions. Batch-actions does not support a TSO-free environment outside this scope.
- The CRABCFG configuration file documents which CA Endevor® SCM actions are currently supported.
- A functional sample job, FEK.#CUST.CNTL(CRABATCA), is provided to execute the batch actions, but the intent of batch-actions is that this sample is customized to invoke your current CA Endevor® SCM procedures.
- Ensure that there are sufficient JES initiators available in the class used to submit the batch-action JCLs.
- When using JES in a SYSPLEX environment, ensure that the job runs on the current system, or that the completion information is routed back to the system hosting Developer for System z, so that the CA Endevor® SCM RAM can check the status.

CRABCFG, CA Endevor® SCM RAM batch-action configuration

CRABCFG defines the configuration variables related to CA Endevor® SCM RAM batch-actions.

CRABCFG is located in FEK.#CUST.PARMLIB, unless you specified a different location when you customized and submitted job FEK.SFEKSAMP(FEKSETUP). See "Customization setup" on page 13 for more details.

See the following sample CRABCFG file, which must be customized to match your system environment. Comment lines start with a pound sign (#), when using a US code page. Comments behind a directive and its assigned value are supported. Spaces around the equal sign (=) are supported. Line continuations are not supported.

Note: Changes are in effect for all CARMA servers started after the update.

```
  # Location of batch action JCL
  SKELETON-DD = CRABSKEL
  #
  # batch action JCL members within SKELETON-DD
  ADD-ELEMENT      = CRABATCA
  GENERATE-ELEMENT = CRABATCA
| MOVE-ELEMENT     = CRABATCA
| DELETE-ELEMENT   = CRABATCA
| RETRIEVE-ELEMENT = CRABATCA
| SIGNIN-ELEMENT   = CRABATCA
| PRINT-ELEMENT    = CRABATCA
| PRINT-MEMBER     = CRABATCA
  #
  # Command substitution key within batch action JCL
  BSTIPT01-KEY = <CRA_BSTIPT01>
```

Figure 24. CRABCFG - CA Endevor® SCM RAM batch-action configuration

SKELETON-DD
Name of the DD statement that references one or more PDS(E) data sets holding the batch-action skeleton JCLs. The sample value is CRABSKEL. Can be changed if desired. This DD must be defined to the CARMA server in CRASUBCA or crastart.endevor.conf.

GENERATE-ELEMENT and other CA Endevor® SCM actions
The key names represent the CA Endevor® SCM actions that are supported by batch-action and cannot be changed. The value assigned to each key is the member name of the related skeleton JCL. The sample value is CRABATCA for all keys. Can be changed if desired.

BSTIPT01-KEY
Substitution key for the actual CA Endevor® SCM command string. The sample value is <CRA_BSTIPT01>. Can be changed if desired. The first occurrence (that is not part of a comment) of this substitution key within the skeleton JCL will be replaced by the command string that will instruct CA Endevor® SCM to do the requested action against the requested element.

CRABATCA, CA Endevor® SCM RAM batch action JCL

CRABATCA is a sample skeleton JCL used for batch-actions. See the documentation within the member for customization instructions if you want to change the defaults.

CRABATCA is located in FEK.#CUST.CNTL, unless you specified a different location when you customized and submitted job FEK.SFEKSAMP(FEKSETUP). See "Customization setup" on page 13 for more details.

Note: Changes are in effect for all CARMA servers started after the update.

```
//<USERID>B JOB CLASS=A,MSGCLASS=A,MSGLEVEL=(1,1)
//*
//CRABATCA EXEC PGM=NDVRC1,DYNAMNBR=1500,REGION=4096K,PARM='C1BM3000'
//STEPLIB  DD DISP=SHR,DSN=CA.NDVRU.AUTHLIB                  * NDVR R12
//         DD DISP=SHR,DSN=CA.NDVR.AUTHLIB                   * NDVR R12
//*         DD DISP=SHR,DSN=CA.NDVR.CSIQAUTU                 * NDVR R14
//*         DD DISP=SHR,DSN=CA.NDVR.CSIQAUTH                 * NDVR R14
//CONLIB   DD DISP=SHR,DSN=CA.NDVR.CONLIB                    * NDVR R12
//*CONLIB   DD DISP=SHR,DSN=CA.NDVR.CSIQLOAD                 * NDVR R14
//C1MSGS1  DD SYSOUT=*
//C1MSGS2  DD SYSOUT=*
//C1PRINT  DD SYSOUT=*,DCB=(RECFM=FBA,LRECL=133)
//SYSOUT   DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYMDUMP  DD DUMMY
//SYSIN    DD DUMMY
//BSTIPT01 DD *
SET STOPRC 16 .
<CRA_BSTIPT01>
//*
```

Figure 25. CRABATCA - CA Endevor® SCM RAM batch-action JCL

CARMA return codes

CARMA can report various error codes to the client or in the host logs. The details provided with the error, and the information in Table 11, can help you locate the error and work towards a resolution.

Table 11. CARMA return codes

Error range	Error type
4-99	Generic CARMA errors
100-199	Generic RAM errors
200-399	CRASERV (CARMA server) errors
400-499	RSE (CARMA miner) errors
500-899	RAM specific errors
900-999	TSO and TCP/IP errors

Some common return codes are:

- 220: CARMA server ends due to inactivity timeout (this is not an error)
- 990: CARMA server is unable to connect to the port on which CARMA miner is listening

(Optional) Supporting multiple RAMs

CARMA allows that multiple RAMs are defined and can run them concurrently. However, since there is only one CARMA server active per user, even when there are multiple RAMs, some configuration changes might be required to make this setup work.

RAMs are defined by a RAM developer in the CARMA configuration VSAM data set, CRADEF. During startup, the CARMA server, CRASERV, will identify all defined RAMs and present the information to the CARMA client. The user can then select one or more RAMs, which will be loaded into the CARMA server.

Since RAMs are active as plug-ins of the CARMA server, you must ensure that all prerequisites (such as data set allocations) for each of the RAMs are available in the address space of the CARMA server. This might require changes to the CARMA configuration samples, such as CRASUBMT or crastart.conf, which are shipped with Developer for System z.

Example

In the following example, you start from an existing setup with the CA Endevor® SCM RAM, using the CRASTART startup method, and add the sample PDS RAM.

Definitions for the CA Endevor® SCM RAM:

- FEK.SFEKVSM2(CRA0VCAD) - CRADEF definitions
- FEK.SFEKVSM2(CRA0VCAS) - CRASTRS definitions
- /etc/rdz/crastart.endevor.conf - CRASTART configuration file

Definitions for the PDS RAM:

- FEK.SFEKVSM2(CRA0VDEF) - CRADEF definitions
- FEK.SFEKVSM2(CRA0VSTR) - CRASTRS definitions
- FEK.#CUST.CRARAM1 - CRARAM1 definitions

The process starts with a RAM developer gathering the data and information needed by the system programmer to complete the setup.

1. Extract the data specific for the PDS RAM from the SFEKVSM2 members (these members hold definitions for all sample RAMs, not just the PDS RAM).
2. Merge this data with the CA Endevor® SCM RAM SFEKVSM2 members.
3. Create a list of PDS RAM specific prerequisites:
   - DD CRARAM1, linked to FEK.#CUST.CRARAM1
   - TSO environment

The system programmer then uses this data to create the updated CARMA VSAM data sets and uses the prerequisite information to create a CRASTART configuration file that is capable of supporting both RAMs.

1. Use the combined data as input for the CRA$VDEF and CRA$VSTR jobs to create the updated CARMA configuration and custom information VSAM data sets, CRADEF and CRASTRS. The CRAMSG VSAM is specific for the CARMA server, and thus identical for both RAMs.
2. Add a CRARAM1 definition to crastart.endevor.conf:
   ```
   CRARAM1 = FEK.#CUST.CRARAM1
   ```
3. Verify the PROGRAM statement in crastart.endevor.conf to ensure it is capable of providing the environment needed by both RAMs:
   ```
   PROGRAM=IKJEFT01 %CRANDVRA NDVRC1 PGM(CRASERV)
     PARM(&CRAPRM1. &CRAPRM2.)
   ```
   - IKJEFT01: TSO, used to allow certain authorized calls in a non-authorized environment, and used as environment to run the CA Endevor® SCM RAM pre-allocation exec.
   - %CRANDVRA: CA Endevor® SCM RAM pre-allocation exec (located in FEK.SFEKPROC), that allocates temporary user-specific working data sets.
   - NDVRC1: CA Endevor® back end, which has a built in mechanism to execute TSO and ISPF commands.
   - PGM(CRASERV): command to start a CARMA server, in ISPF command format

- PARM(&CRAPRM1. &CRAPRM2.): parameters for CRASERV, in ISPF command format. &CRAPRM1 is the port to be used and &CRAPRM2 is the timeout value.

The CA Endevor® SCM RAM is active in an ISPF environment, which implies that the TSO environment required by the PDS RAM is also available.

(Optional) Custom allocation exec

All CARMA server startup methods have limitations when it comes to data set allocation. For example, TSO prefix substitution is not available in JCL or CRASTART.

However, by creating an exec that is invoked after TSO starts (or ISPF, depending on your needs) and before CARMA is started, you can use the whole range of variables and services available in TSO (or ISPF) to do the desired allocations.

To simplify the setup, the allocation exec should accept the actual startup command as argument and execute it after all allocations are done.

Note: When creating an allocation exec, ensure you do not destroy allocations done earlier in the CARMA startup process by CRASTART or your startup JCL.

FEK.SFEKPROC(CRANDVRA), the allocation exec for CA Endevor® SCM RAM, uses this technique to allocate cataloged temporary data sets that have the user's TSO prefix as high level qualifier.

The following samples show how to invoke an allocation exec that only requires TSO. Refer to FEK.SFEKPROC(CRANDVRA) for a sample on how to code the exec.

crastart*.conf

```
SYSPROC = my.exec.library
PROGRAM = IKJEFT01 %myexec ISPSTART PGM(CRASERV) PARM(&CRAPRM1.
&CRAPRM2.)
```

CRASUB*

```
//SYSPROC  DD DISP=SHR,DSN=my.exec.library
//SYSTSIN  DD *
%myexec ISPSTART PGM(CRASERV) PARM(&PORT &TIMEOUT)
//*
```

Note: Output generated by the allocation exec is shown in DD SYSTSPRT of the CARMA server.

(Optional) IRXJCL versus CRAXJCL

If the CARMA server is started using TSO (IKJEFTxx), you might experience problems if your RAMs call services which in turn call the IRXJCL REXX batch interface. The problem can occur when the processors called by the RAM previously ran either without TSO, or only in online TSO and dynamically allocates DD SYSTSIN or SYSTSPRT. A sample program, CRAXJCL, is provided to work around this problem.

Your processor might fail if it attempts to allocate SYSTSIN or SYSTSPRT (required for IRXJCL) because batch TSO (required for CARMA) already has those DD names allocated and open. The CRAXJCL replacement module attempts to allocate

SYSTSIN and SYSTSPRT to DUMMY but ignores the errors which occur if the allocations fail. It then calls IRXJCL to do the actual work.

This means that when your processors run in a CARMA environment started by TSO, the allocations to SYSTSIN and SYSTSPRT are the same as those used by CARMA. When the processors are run outside of TSO/CARMA, the SYSTSIN and SYSTSPRINT allocations will be created by CRAXJCL. Therefore, your processors must not rely on the contents of the data set allocated to SYSTSIN.

It is assumed that calls to IRXJCL use the PARM field to pass the REXX name and startup parameters, as documented in *TSO/E REXX Reference* (SA22-7790). This means that SYSTSIN can safely be used by CARMA. Any output sent to SYSTSPRT by IRXJCL will end up in CARMA's log.

Processors that call the CRAXJCL replacement module should not attempt to allocate DD SYSTSIN or SYSTSPRT before calling CRAXJCL.

Create CRAXJCL

The CRAXJCL replacement module is shipped in source format because you will need to customize it to specify the specific allocations you want to use for SYSTSPRT. SYSTSIN should usually be allocated to a dummy data set.

Sample assembler source code and a sample compile/bind job are available as `FEK.#CUST.ASM(CRAXJCL)` and `FEK.#CUST.JCL(CRA#CIRX)` respectively, unless you specified a different location when you customized and submitted job `FEK.SFEKSAMP(FEKSETUP)`. See "Customization setup" on page 13 for more details.

Customize the `CRAXJCL` assembler source code per your needs, using the documentation within the member. Afterwards, customize and submit the `CRA#CIRX` JCL to create the CRAXJCL load module. See the documentation within the member for customization instructions.

If you want to, you can rename IRXJCL to something else, adjust the CRAXJCL source to call this new name for IRXJCL and compile it, and then rename the CRAXJCL load module to IRXJCL. This setup might be easier than changing all your calls to IRXJCL.

Chapter 4. (Optional) SCLM Developer Toolkit

SCLM Developer Toolkit provides the tools needed to extend the capabilities of SCLM to the client. SCLM itself is a host-based source code manager that is shipped as part of ISPF.

The SCLM Developer Toolkit has an Eclipse-based plugin that interfaces to SCLM and provides for access to all SCLM processes for legacy code development as well as support for full Java and J2EE development on the workstation with synchronization to SCLM on the mainframe including building, assembling, and deployment of the J2EE code from the mainframe.

Requirements and checklist

You will need assistance of an SCLM administrator and optionally a security administrator to complete this customization task, which requires the following resources and special customization tasks:

- APF and LINKLIST updates
- Define SCLM language translators for JAVA/J2EE support
- Define SCLM types for JAVA/J2EE support
- (Optional) Security rule to allow users update to an SCLM VSAM
- (Optional) Installation of Ant

In order to start using SCLM Developer Toolkit at your site, you must perform the following tasks. Unless otherwise indicated, all tasks are mandatory.

1. Verify and adjust prerequisites and PARMLIB updates. For details, see “Prerequisites.”
2. Customize Developer for System z configuration files. For details see:
   - “ISPF.conf updates for SCLMDT” on page 78
   - “rsed.envvars updates for SCLMDT” on page 79
3. Optionally define long/short name translation support. For details, see “(Optional) Long/short name translation” on page 79.
4. Optionally install and customize Ant to use the JAVA/J2EE build support. For details, see “(Optional) Install and customize Ant” on page 82.
5. Update SCLM to define SCLMDT-specific parts. For details, see “SCLM updates for SCLMDT” on page 83.
6. Optionally set up automation to periodically clean up the SCLMDT work area. For details, see “Remove old files from WORKAREA and /tmp” on page 84.

Prerequisites

Refer to *IBM Rational Developer for System z Prerequisites* (SC23-7659) for a list of required SCLM maintenance.

This publication also documents the Ant specifications needed for JAVA/J2EE builds in SCLM Developer Toolkit.

Attention: SCLM Developer Toolkit requires the usage of ISPF's TSO/ISPF Client Gateway, which implies that z/OS 1.8 or higher is required.

As described in "PARMLIB changes" on page 14, SCLM Developer Toolkit requires additional customization of system settings. These changes include:

- (BPXPRMxx) Increase the maximum number of processes per z/OS UNIX user ID.
- (PROGxx) APF authorize SYS1.LINKLIB and the REXX runtime, REXX.V1R4M0.SEAGLPA or REXX.V1R4M0.SEAGALT.
- (PROGxx/LPALSTxx) Place ISP.SISPLPA, ISP.SISPLOAD, SYS1.LINKLIB and the REXX runtime in LINKLIST/LPALIB.

Also, SCLM Developer Toolkit uses SDSF or the TSO **OUTPUT** command to retrieve job completion status and job output. Both methods require some additional attention:

- SDSF must be ordered, installed, and configured separately. It also requires the usage of JES2.
- The default settings for the TSO **OUTPUT** command let a user retrieve job output that begins with his user ID only. If you want to use the **OUTPUT** facility fully, then the sample TSO/E exit IKJEFF53 might need to be modified so that a user can retrieve job output he owns, but that does not begin with his user ID. For more information about this exit, refer to *TSO/E Customization* (SA22-7783).

Users require READ, WRITE, and EXECUTE permission to the z/OS UNIX directories /tmp/ and /var/rdz/WORKAREA/. Directory WORKAREA/ is located in /var/rdz/, unless you specified a different location when you customized and submitted job FEK.SFEKSAMP(FEKSETUP). See "Customization setup" on page 13 for more details.

ISPF.conf updates for SCLMDT

SCLM Developer Toolkit uses the standard ISPF/SCLM skeletons, so ensure that skeleton library ISP.SISPSLIB is allocated to the ISPSLIB concatenation in ISPF.conf. The usage of the ISP.SISPSENU data set is optional.

ISPF.conf is located in /etc/rdz/, unless you specified a different location when you customized and submitted job FEK.SFEKSAMP(FEKSETUP). See "Customization setup" on page 13 for more details. You can edit the file with the TSO **OEDIT** command.

Note: Changes are in effect for all clients connecting to the host after the update.

The following sample code shows the ISPF.conf file, which must be customized to match your system environment. Comment lines start with an asterisk (*). Add data sets to the concatenation on the same line and separate the names with a comma (,). See "ISPF.conf, ISPF's TSO/ISPF Client Gateway configuration file" on page 46 for more details on customizing ISPF.conf.

```
* REQUIRED:
sysproc=ISP.SISPCLIB,FEK.SFEKPROC
ispmlib=ISP.SISPMENU
isptlib=ISP.SISPTENU
ispplib=ISP.SISPPENU
ispslib=ISP.SISPSLIB
ispllib=ISP.SISPLOAD

* OPTIONAL:
*allocjob = ISP.SISPSAMP(ISPZISP2)
*ISPF_timeout = 900
```

Figure 26. ISPF.conf updates for SCLMDT

Note:

- You can add your own DD-like statements and data set concatenations to customize the TSO environment, thus mimicking a TSO logon procedure. See "Customizing the TSO environment" in the *Host Configuration Reference* (SC14-7290) for more details.
- When you are doing batch builds, ensure that the customized version of the FLMLIBS skeleton is concatenated before the ISPF/SCLM skeleton library.

  ```
  ispslib=hlq.USERSKEL,ISP.SISPSLIB
  ```

rsed.envvars updates for SCLMDT

SCLM Developer Toolkit uses some directives set in rsed.envvars to locate data sets and directories.

rsed.envvars is located in /etc/rdz/, unless you specified a different location when you customized and submitted job FEK.SFEKSAMP(FEKSETUP). See "Customization setup" on page 13 for more details. You can edit the file with the TSO **OEDIT** command.

Note: The RSED started task must be restarted to pick up any changes you make.

The following code sample shows the SCLMDT directives in rsed.envvars, which must be customized to match your system environment. See "rsed.envvars, RSE configuration file" on page 30 for more details on customizing rsed.envvars.

```
_SCLMDT_CONF_HOME=/var/rdz/sclmdt
#STEPLIB=$STEPLIB:FEK.SFEKAUTH:FEK.SFEKLOAD
#_SCLMDT_TRANTABLE=FEK.#CUST.LSTRANS.FILE
#ANT_HOME=/usr/lpp/Apache/Ant/apache-ant-1.7.1
_SCLMDT_BASE_HOME=$RSE_HOME
_SCLMDT_WORK_HOME=$CGI_ISPHOME
CGI_DTWORK=$_SCLMDT_WORK_HOME
```

Figure 27. rsed.envvars updates for SCLMDT

(Optional) Long/short name translation

SCLM Developer Toolkit provides the ability to store long name files (which are files with names greater than 8 characters or in mixed case) into SCLM. This is achieved through the use of a VSAM file that contains the mapping of the long file name to the 8 character member name used in SCLM.

Note:

- For versions previous to z/OS 1.8, this facility is provided through a base ISPF/SCLM PTF that addresses APAR OA11426.

- The long/short name translation is also used by other SCLM-related products, such as IBM SCLM Administrator Toolkit.

Create LSTRANS.FILE, the long/short name translation VSAM

Customize and submit sample member FLM02LST in the ISPF sample library ISP.SISPSAMP, to create the long/short name translation VSAM. The configuration steps in this publication expect the VSAM to be named FEK.#CUST.LSTRANS.FILE, as shown in the following sample setup JCL.

```
//FLM02LST JOB <job parameters>
//*
//* CAUTION: This is neither a JCL procedure nor a complete job.
//* Before using this sample, you will have to make the following
//* modifications:
//* 1. Change the job parameters to meet your system requirements.
//* 2. Change ****** to the volume that will hold the VSAM.
//* 3. Change all references of FEK.#CUST.LSTRANS.FILE to
//*    match your naming convention for the SCLM translate VSAM.
//*
//CREATE   EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
  DELETE FEK.#CUST.LSTRANS.FILE
  SET MAXCC=0
  DEFINE CLUSTER(NAME(FEK.#CUST.LSTRANS.FILE) -
                 VOLUMES(******) -
                 RECORDSIZE(58 2048) -
                 SHAREOPTIONS(3 3) -
                 CYLINDERS(1 1) -
                 KEYS(8 0) -
                 INDEXED) -
         DATA   (NAME(FEK.#CUST.LSTRANS.FILE.DATA)) -
         INDEX  (NAME(FEK.#CUST.LSTRANS.FILE.INDEX))

  /* DEFINE ALTERNATE INDEX WITH NONUNIQUE KEYS -> ESDS */

  DEFINE ALTERNATEINDEX(-
                 NAME(FEK.#CUST.LSTRANS.FILE.AIX) -
                 RELATE(FEK.#CUST.LSTRANS.FILE) -
                 RECORDSIZE(58 2048) -
                 VOLUMES(******) -
                 CYLINDERS(1 1) -
                 KEYS(50 8) -
                 UPGRADE -
                 NONUNIQUEKEY) -
         DATA   (NAME(FEK.#CUST.LSTRANS.FILE.AIX.DATA)) -
         INDEX  (NAME(FEK.#CUST.LSTRANS.FILE.AIX.INDEX))
/*
//*
//PRIME    EXEC PGM=IDCAMS,COND=(0,LT)
//SYSPRINT DD SYSOUT=*
//INITREC  DD *
INITREC1
/*
//SYSIN    DD *
  REPRO INFILE(INITREC) -
        OUTDATASET(FEK.#CUST.LSTRANS.FILE)
  IF LASTCC = 4 THEN SET MAXCC=0

  BLDINDEX IDS(FEK.#CUST.LSTRANS.FILE) -
           ODS(FEK.#CUST.LSTRANS.FILE.AIX)

  IF LASTCC = 0 THEN -
    DEFINE PATH (NAME(FEK.#CUST.LSTRANS.FILE.PATH) -
           PATHENTRY (FEK.#CUST.LSTRANS.FILE.AIX))
/*
```

Figure 28. FLM02LST - long/short name translation setup JCL

Note: Users need UPDATE authority to this VSAM data set, as described in "Security considerations" in *Host Configuration Reference* (SC14-7290).

rsed.envvars updates for long/short name translation

Before using the long/short name translation, uncomment and set the `rsed.envvars` environment variable `_SCLMDT_TRANTABLE` to match the name of the long/short name translation VSAM.

`rsed.envvars` is located in `/etc/rdz/`, unless you specified a different location when you customized and submitted job `FEK.SFEKSAMP(FEKSETUP)`. See "Customization setup" on page 13 for more details. You can edit the file with the TSO **OEDIT** command.

Note: The RSED started task must be restarted to pick up any changes you make.

(Optional) Install and customize Ant

This step is only required if you plan to use the JAVA/J2EE build support in SCLM.

Apache Ant is an open source Java build tool and can be downloaded from `http://ant.apache.org/`. Ant consists of text files and scripts which are distributed in ASCII format and thus require an ASCII/EBCDIC translation to run in z/OS UNIX.

Perform the following steps to implement Ant on z/OS, and to define it to Developer for System z:

- Download, in binary format, the latest Ant compressed file into the z/OS UNIX file system. It is recommended that you download the .zip version of ANT due to problems that might be encountered on z/OS when extracting versions of suffix format `tar.gz` or `tar.bz2`.
- Open a z/OS UNIX command-line session to continue the installation, for example with the **TSO OMVS** command.
- Make a home directory for the Ant install with the **mkdir -p /home-dir** command and make it your current directory with the **cd /home-dir** command.
- Use the JAR extract command **jar -xf apache-ant-1.7.1.zip** to extract the file to the current directory. A Java bin directory must be in your local z/OS UNIX PATH to use the **jar** command. Otherwise, fully qualify the command with the Java bin location (for example, **/usr/lpp/java/J6.0/bin/jar -xf apache-ant-1.7.1.zip**).
- Convert all Ant text files to EBCDIC by (optionally customizing and) executing sample script `/usr/lpp/rdz/samples/BWBTRANT`.

  Note: Execute this script only once. Multiple runs will corrupt your Ant install.
- To check for successful translation, locate and browse a text file within the ANT directory, such as `apache-ant-1.7.1/README`. If the file is readable, then the translation was successful.
- Use the **chmod –R 755 *** command to enable all users to READ and EXECUTE files in the ANT directory.
- Before using Ant, set the `rsed.envvars` environment variables `JAVA_HOME` and `ANT_HOME`.
  - JAVA_HOME is required to point to the Java home directory, for example:
    ```
    JAVA_HOME=/usr/lpp/java/IBM/J6.0
    ```
  - ANT_HOME is required to point to the Ant home directory, for example:
    ```
    ANT_HOME=/usr/lpp/Apache/Ant/apache-ant-1.7.1
    ```

For example:

- `TSO OMVS`
- `mkdir –p /usr/lpp/Apache/Ant`
- `cd /usr/lpp/Apache/Ant`
- `jar –xf /u/userid/apache-ant-1.7.1`
- `/usr/lpp/rdz/samples/BWBTRANT`
- `cat ./apache-ant-1.7.1/README`
- `chmod -R 755 *`
- `oedit /etc/rsed.envvars`

To test that the Ant initialization has been successful:

- Add the Ant and Java bin directories to the environment variable PATH.

  Example:

  ```
  export PATH=/usr/lpp/Apache/Ant/apache-ant-1.7.1/bin:$PATH
  export PATH=/usr/lpp/java/J6.0/bin:$PATH
  ```

- Execute **ant -version** to display the version, if successfully installed.

  Example:

  ```
  ant -version
  ```

Note: Setting the `PATH` statement in this way is necessary for testing only, not for operational use.

SCLM updates for SCLMDT

SCLM itself also requires customization to work with SCLM Developer Toolkit. Refer to *IBM Rational Developer for System z SCLM Developer Toolkit Administrator's Guide* (SC23-9801) for more information about the required customization tasks:

- Define language translators for JAVA/J2EE support
- Define SCLM types for JAVA/J2EE support

To complete the customization and project definition tasks, the SCLM administrator needs to know several Developer for System z customizable values, as described in Table 12.

Table 12. SCLM administrator checklist

Description	• **Default value** • **Where to find the answer**	**Value**
Developer for System z sample library	• `FEK.SFEKSAMV` • SMP/E installation	
Developer for System z sample directory	• `/usr/lpp/rdz/samples` • SMP/E installation	
Java bin directory	• `/usr/lpp/java/J6.0/bin` • `rsed.envvars - $JAVA_HOME/bin`	
Ant bin directory	• `/usr/lpp/Apache/Ant/apache-ant-1.7.1/bin` • `rsed.envvars - $ANT_HOME/bin`	
WORKAREA home directory	• `/var/rdz` • `rsed.envvars - $CGI_ISPWORK`	

Table 12. SCLM administrator checklist (continued)

Description	• **Default value** • **Where to find the answer**	Value
SCLMDT project configuration home directory	• `/var/rdz/sclmdt` • `rsed.envvars - $_SCLMDT_CONF_HOME`	
Long/short name translation VSAM	• `FEK.#CUST.LSTRANS.FILE` • `rsed.envvars - $_SCLMDT_TRANTABLE`	

Remove old files from WORKAREA and /tmp

SCLM Developer Toolkit and ISPF's TSO/ISPF Client Gateway share the same WORKAREA and /tmp directory, both of which might need a periodic cleanup. Refer to "(Optional) WORKAREA and /tmp cleanup" on page 109 for more information about this.

Chapter 5. (Optional) Application Deployment Manager

Developer for System z uses certain functions of Application Deployment Manager as a common deployment approach for various components. The customization steps listed in this chapter are required if your developers use any of the following functions:

- Enterprise Service Tools
- BMS Screen Designer
- MFS Screen Designer
- CICSTS Code Generation

Note: Enterprise Service Tools encompasses multiple tools, such as the Service Flow Modeler (SFM) and XML Services for the Enterprise.

Customizing Application Deployment Manager adds the CICS Resource Definition (CRD) server, which runs as a CICS application on z/OS to support the following functions:

- CICS resource queries
- CICS resource definition install and uninstall requests in both CICSplex SM and non-CICSplex SM environments
- Program and mapset phase-in requests
- Pipeline scan requests
- Manifest export, import, and update requests

CICS administrators can find more information about the CRD server in "CICSTS considerations" in the *Host Configuration Reference* (SC14-7290).

Requirements and checklist

You will need assistance of a CICS administrator, a TCP/IP administrator and a security administrator to complete this customization task, which requires the following resources or special customization tasks:

- TCP/IP port for external communication
- Update CICS region JCL
- Update CICS region CSD
- Define group to CICS region
- Security rule to allow administrators update to an Application Deployment Manager VSAM
- CICSTS security setup
- (Optional) Define CICS transaction names
- (Optional) Security rule to allow users update to an Application Deployment Manager VSAM

In order to start using Application Deployment Manager at your site, you must perform the following tasks. Unless otherwise indicated, all tasks are mandatory.

1. Create the CRD repository. For details, see “CRD repository” on page 86.
2. Choose the CICS interface (RESTful or Web Service) to be used. (The interfaces can co-exist). For details see “RESTful versus Web Service” on page 87.

3. If desired, do the RESTful specific customizations. For details see "CRD server using the RESTful interface" on page 87.
   - Define the CRD server to the CICS primary connection region
   - Optionally define the CRD server to CICS non-primary connection regions.
   - Optionally customize the CRD server transaction IDs.
4. If desired, do the Web Service specific customizations. For details see "CRD server using the Web Service interface" on page 88.
   - Add the (possibly customized) pipeline message handler to the CICS RPL concatenation.
   - Define the CRD server to the CICS primary connection region.
   - Optionally define the CRD server to CICS non-primary connection regions.
5. Optionally create the manifest repository. For details, see "(Optional) Manifest repository" on page 90.

CRD repository

Customize and submit job ADNVCRD to allocate and initialize the CRD repository VSAM data set. Refer to the documentation within the member for customization instructions.

ADNVCRD is located in FEK.#CUST.JCL, unless you specified a different location when you customized and submitted job FEK.SFEKSAMP(FEKSETUP). See "Customization setup" on page 13 for more details.

You should create a separate repository for each CICS primary connection region. Sharing the repository implies that all related CICS regions will use the same values stored in the repository.

Note:

- An existing CRD server repository must be enlarged to enable the URIMAP support added to the Administrative utility in Developer for System z version 7.6.1. See "Administrative utility migration notes" in the *Host Configuration Reference* (SC14-7290) for more details.
- Unless notified otherwise, your current CRD server repository (holding your customized values) can be reused across Developer for System z releases.

Users require READ access to the CRD repository, CICS administrators require UPDATE access.

CICS administrative utility

Developer for System z provides the administrative utility to let CICS administrators provide the default values for CICS resource definitions. These defaults can be read-only, or can be editable by the application developer.

The administrative utility is invoked by sample job ADNJSPAU. The usage of this utility requires UPDATE access to the CRD repository.

ADNJSPAU is located in FEK.#CUST.JCL, unless you specified a different location when you customized and submitted job FEK.SFEKSAMP(FEKSETUP). See "Customization setup" on page 13 for more details.

More information is available in "CICSTS considerations" in the *Host Configuration Reference* (SC14-7290).

RESTful versus Web Service

CICS Transaction Server provides in version 4.1 and higher support for an HTTP interface designed using Representational State Transfer (RESTful) principles. This RESTful interface is now the strategic CICSTS interface for use by client applications. The older Web Service interface has been stabilized, and enhancements will be for the RESTful interface only.

Application Deployment Manager follows this statement of direction and requires the RESTful CRD server for all services that are new to Developer for System version 7.6 or higher.

The RESTful and Web Service interfaces can be active concurrently in a single CICS region, if desired. In this case, there will be two CRD servers active in the region. Both servers will share the same CRD repository. Note that CICS will issue some warnings about duplicate definitions when the second interface is defined to the region.

CRD server using the RESTful interface

The information in this section describes how to define the CRD server that uses the RESTful interface to communicate with the Developer for System z client.

The RESTful and Web Service interfaces can be active concurrently in a single CICS region, if desired. In this case, there will be two CRD servers active in the region. Both servers will share the same CRD repository. Note that CICS will issue some warnings about duplicate definitions when the second interface is defined to the region.

CICS primary connection region

The CRD server must be defined to the primary connection region. This is the Web Owning Region (WOR) that will process Web Service requests from Developer for System z.

- Place the load modules `FEK.SFEKLOAD(ADNCRD*, ADNANAL and ADNREST)` in the CICS RPL concatenation (DD statement DFHRPL) of the CICS primary connection region. You should do this by adding the installation data set to the concatenation so that applied maintenance is automatically available to CICS.
- Customize and submit job `ADNCSDRS` to update the CICS System Definition (CSD) for the CICS primary connection region. Refer to the documentation within the member for customization instructions.

  `ADNCSDRS` is located in `FEK.#CUST.JCL`, unless you specified a different location when you customized and submitted job `FEK.SFEKSAMP(FEKSETUP)`. See "Customization setup" on page 13 for more details.
- Use the appropriate CEDA command to install the Application Deployment Manager group for this region, for example:

  ```
  CEDA INSTALL GROUP(ADNPCRGP)
  ```

CICS non-primary connection regions

The CRD server can also be used with one or more additional non-primary connection regions, which are usually Application Owning Regions (AOR).

Note: It is not necessary to perform these steps if CICSPlex® SM Business Application Services (BAS) is used to manage your CICS resource definitions.

- Place the Application Deployment Manager load module FEK.SFEKLOAD(ADNCRD*) in the CICS RPL concatenation (DD statement DFHRPL) of these non-primary connection regions. It is recommended that you do this by adding the installation data set to the concatenation so that applied maintenance is automatically available to CICS.
- Customize and submit job ADNCSDAR to update the CSD for these non-primary, connection regions. Refer to the documentation within the member for customization instructions.

  ADNCSDAR is located in FEK.#CUST.JCL, unless you specified a different location when you customized and submitted job FEK.SFEKSAMP(FEKSETUP). See "Customization setup" on page 13 for more details.
- Use the appropriate CEDA command to install the Application Deployment Manager group for these regions, for example:

  ```
  CEDA INSTALL GROUP(ADNARRGP)
  ```

(Optional) Customize CRD server transaction IDs

Developer for System z supplies multiple transactions that are used by the CRD server when defining and inquiring CICS resources.

Table 13. Default CRD server transaction IDs

Transaction	Description
ADMS	For requests from the Manifest Processing tool to change CICS resources. Typically, this is intended for CICS administrators.
ADMI	For requests that define, install, or uninstall CICS resources.
ADMR	For all other requests that retrieve CICS environmental or resource information.

You can change the transaction IDs to match your site standards by following these steps:

1. Customize and submit ADNTXNC to create load module ADNRCUST. Refer to the documentation within the member for customization instructions.
2. Place the resulting ADNRCUST load module in the CICS RPL concatenation (DD statement DFHRPL) of the CICS regions where the CRD server is defined.
3. Customize and submit ADNCSDTX to define ADNRCUST as program to the CICS regions where the CRD server is defined. Refer to the documentation within the member for customization instructions.

Note: The RESTful CRD server will always try to load the ADNRCUST load module. So you can get a small performance benefit by creating and defining the ADNRCUST load module, even if you do not change the transaction IDs.

CRD server using the Web Service interface

The information in this section describes how to define the CRD server that uses the Web Service interface to communicate with the Developer for System z client.

The RESTful and Web Service interfaces can be active concurrently in a single CICS region, if desired. In this case, there will be two CRD servers active in the region. Both servers will share the same CRD repository. Note that CICS will issue some warnings about duplicate definitions when the second interface is defined to the region.

Pipeline message handler

The pipeline message handler (ADNTMSGH) is used for security by processing the user ID and password in the SOAP header. ADNTMSGH is referenced by the sample pipeline configuration file and must therefore be placed into the CICS RPL concatenation. Refer to "CICSTS considerations" in the *Host Configuration Reference* (SC14-7290) to learn more about the pipeline message handler and the required security setup.

Developer for System z supplies multiple transactions that are used by the CRD server when defining and inquiring CICS resources. These transaction IDs are set by ADNTMSGH, depending on the requested operation. Sample COBOL source code is provided to allow site-specific customizations to ADNTMSGH:

Table 14. Default CRD server transaction IDs

Transaction	Description
ADMS	For requests from the Manifest Processing tool to change CICS resources. Typically, this is intended for CICS administrators.
ADMI	For requests that define, install or uninstall CICS resources.
ADMR	For all other requests that retrieve CICS environmental or resource information.

Using the default:

- Place the FEK.SFEKLOAD(ADNTMSGH) load module in the CICS RPL concatenation (DD statement DFHRPL) of the CICS primary connection region. It is recommended that you do this by adding the installation data set to the concatenation so that applied maintenance is automatically available to CICS.

Customizing ADNTMSGH:

Sample members ADNMSGH* are located in FEK.#CUST.JCL and FEK.#CUST.COBOL, unless you specified a different location when you customized and submitted job FEK.SFEKSAMP(FEKSETUP). See “Customization setup” on page 13 for more details.

- Customize the sample Pipeline Message Handler (COBOL) source code, FEK.#CUST.COBOL(ADNMSGHS), to match your site’s standards.
- Customize and submit job FEK.#CUST.JCL(ADNMSGHC) to compile the customized ADNMSGHS source. Refer to the documentation within ADNMSGHC for customization instructions. Note that the resulting load module must be named ADNTMSGH.
- Place the resulting ADNTMSGH load module in the CICS RPL concatenation (DD statement DFHRPL) of the CICS primary connection region.

Note: Ensure that the customized ADNTMSGH load module is located before any reference to FEK.SFEKLOAD, otherwise the default one will be used.

CICS primary connection region

The CRD server must be defined to the primary connection region. This is the region that will process service requests from Developer for System z.

- Place the load modules FEK.SFEKLOAD(ADNCRD*, ADNANAL and ADNREST) in the CICS RPL concatenation (DD statement DFHRPL) of the CICS primary connection region. It is recommended that you do this by adding the installation data set to the concatenation so that applied maintenance is automatically available to CICS. Note that the pipeline message handler load module, ADNTMSGH, must also be placed in the RPL concatenation, as described in "Pipeline message handler" on page 89.

- Customize and submit job ADNCSDWS to update the CICS System Definition (CSD) for the CICS primary connection region. Refer to the documentation within the member for customization instructions. Note that the transaction IDs used in this job must match the ones used by the Pipeline message handler (which may have been customized).

  ADNCSDWS is located in FEK.#CUST.JCL, unless you specified a different location when you customized and submitted job FEK.SFEKSAMP(FEKSETUP). See "Customization setup" on page 13 for more details.
- Use the appropriate CEDA command to install the Application Deployment Manager group for this region, for example:

  ```
  CEDA INSTALL GROUP(ADNPCRGP)
  ```

CICS non-primary connection regions

The CRD server can also be used with one or more additional non-primary connection regions, which are usually Application Owning Regions (AOR).

Note: It is not necessary to perform these steps if CICSPlex SM Business Application Services (BAS) is used to manage your CICS resource definitions.

- Place the Application Deployment Manager load modules FEK.SFEKLOAD(ADNCRD*) in the CICS RPL concatenation (DD statement DFHRPL) of these non-primary connection regions. You should do this by adding the installation data set to the concatenation so that applied maintenance is automatically available to CICS.
- Customize and submit job ADNCSDAR to update the CSD for these non-primary, connection regions. Refer to the documentation within the member for customization instructions.

  ADNCSDAR is located in FEK.#CUST.JCL, unless you specified a different location when you customized and submitted job FEK.SFEKSAMP(FEKSETUP). See "Customization setup" on page 13 for more details.
- Use the appropriate CEDA command to install the Application Deployment Manager group for these regions, for example:

  ```
  CEDA INSTALL GROUP(ADNARRGP)
  ```

(Optional) Manifest repository

Developer for System z allows clients to browse and optionally change manifests describing selected CICS resources. Depending on permissions set by the CICS administrator, changes can be done directly or exported to the manifest repository for further processing by a CICS administrator.

Note:

- This step is only required for customers that export manifests from Developer for System z to be processed by the Manifest Processing tool.
- The Manifest Processing tool is a plug-in for IBM CICS Explorer.

Customize and submit job ADNVMFST to allocate and initialize the manifest repository VSAM data set, and to define it to the CICS primary connection region. Refer to the documentation within the member for customization instructions. A separate manifest repository must be created for each CICS primary connection region. All users need UPDATE access to the manifest repository.

ADNVMFST is located in FEK.#CUST.JCL, unless you specified a different location when you customized and submitted job FEK.SFEKSAMP(FEKSETUP). See "Customization setup" on page 13 for more details.

Chapter 6. (Optional) Other customization tasks

This section combines a variety of optional customization tasks. Follow the instructions in the appropriate section to configure the desired service.

Customizations to Developer for System z configuration files:

- "(Optional) pushtoclient.properties, Host-based client control"
- "(Optional) ssl.properties, RSE SSL encryption" on page 96
- "(Optional) rsecomm.properties, RSE tracing" on page 98
- "(Optional) include.conf, Forced includes for C/C++ content assist" on page 99

Developer for System z related customizations to (or for) other products:

- "(Optional) DB2 stored procedure" on page 101
- "(Optional) z/OS UNIX subprojects" on page 103
- "(Optional) Include preprocessor support" on page 104
- "(Optional) xUnit support for Enterprise COBOL and PL/I" on page 105
- "(Optional) Enterprise Service Tools support" on page 106
- "(Optional) CICS bidirectional language support" on page 106
- "(Optional) Diagnostic IRZ messages for generated code" on page 107
- "(Optional) Problem Determination Tools support" on page 108
- "(Optional) DB2 and IMS debug support" on page 108
- "(Optional) File Manager support" on page 109
- "(Optional) WORKAREA and /tmp cleanup" on page 109

(Optional) pushtoclient.properties, Host-based client control

This customization task does not require assistance, special resources, or special customization tasks for a basic setup.

If you enable group support, you will need the assistance of a security administrator or an LDAP administrator to complete this customization task, which requires the following resources or special customization tasks:

- Security rule to allow users access to FEK.PTC.* profiles
- Or define user membership of FEK.PTC.* LDAP groups

Developer for System z clients version 8.0.1 and higher can pull client configuration files and product update information from the host when they connect, ensuring that all clients have common settings and that they are up-to-date.

z/OS Projects can be defined individually through the z/OS Projects perspective on the client or can be defined centrally on the host and propagated to the client on a per-user basis. These "host-based projects" look and function exactly like projects defined on the client except that their structure, members, and properties cannot be modified by the client, and they are accessible only when connected to the host.

pushtoclient.properties tells the client if these functions are enabled, and where the related data is stored. (The data is maintained by a Developer for System z client administrator or a development project manager.)

pushtoclient.properties is located in /etc/rdz/, unless you specified a different location when you customized and submitted job FEK.SFEKSAMP(FEKSETUP). See "Customization setup" on page 13 for more details. You can edit the file with the TSO **OEDIT** command. Note that the RSED started task must be restarted before changes take effect.

Since version 8.0.3, the client administrator can create multiple client configuration sets and multiple client update scenarios to fit the needs of different developer groups. This allows users to receive a customized setup, based on criteria like membership of an LDAP group or permit to a security profile. See "Push-to-client considerations" in *Host Configuration Reference* (SC14-7290) for more information about supporting multiple groups.

The following code sample shows the pushtoclient.properties file, which must be customized to match your system environment. Comment lines start with a pound sign (#), when using a US code page. Data lines can only have a directive and its assigned value. Comments are not allowed on the same line. Line continuations are not supported.

```
#
# host-based client control
#
config.enabled=false
product.enabled=false
reject.config.updates=false
reject.product.updates=false
accept.product.license=false
primary.system=false
pushtoclient.folder=/var/rdz/pushtoclient
default.store=com.ibm.ftt.configurations.USS
file.permission=RWX.RWX.RX
```

Figure 29. pushtoclient.properties - Host-based client control configuration file

config.enabled

Indicates whether host-based client control is used for configuration files. The default is false. The valid values are true, false, LDAP, or SAF. See Table 15 on page 95 for the exact meaning of these values.

product.enabled

Indicates whether host-based client control is used for product updates. The default is false. The valid values are true, false, LDAP, or SAF. See Table 15 on page 95 for the exact meaning of these values.

reject.config.updates

Indicates whether a user is allowed to reject configuration updates that are pushed to the client. The default is false. The valid values are true, false, LDAP, or SAF. See Table 15 on page 95 for the exact meaning of these values.

reject.product.updates

Indicates whether a user is allowed to reject product updates that are pushed to the client. The default is false. The valid values are true, false, LDAP, or SAF. See Table 15 on page 95 for the exact meaning of these values.

accept.product.license

Indicates whether the product license is automatically accepted during updates initiated by push-to-client. If enabled, IBM Installation Manager does not ask to accept the license during client update. The default is `false`. The only valid values are `true` and `false`.

primary.system
Host-based client control supports storing system specific data per system, while maintaining common data on a single system to reduce management effort. This directive indicates whether this is the system that stores global, non-system specific, client definitions. The default is `false`. The only valid values are `true` and `false`.

Note: Ensure you have one, and only one, system defined as primary system. Developer for System z client administrators will not be able to export global configuration data unless the target system is a primary system. Developer for System z clients might show erratic behavior when connecting to multiple primary systems with out-of-sync configurations.

pushtoclient.folder
The base directory for the host-based client control definitions. The default is `/var/rdz/pushtoclient`.

default.store
Host-based client control supports different methods for storing the data that is pushed to the client. This directive identifies the "driver", or store, that is used to access the data. The default is `com.ibm.ftt.configurations.USS`, which supports the data being stored in z/OS UNIX flat files.

Note that Developer for System z only provides the `com.ibm.ftt.configurations.USS` store. A third-party store is needed when the data is located somewhere else.

file.permission
The `com.ibm.ftt.configurations.USS` store uses `file.permission` to determine the desired access permissions for files created by the store. The default is `RWX.RWX.RX`, which allows the owner and the owner's default group read and write access to the directory structure and the files within. Everyone else has only read access to the directory structure and the files within.

UNIX standards dictate that permissions can be set for three types of users: owner, group, and other. The fields in the `file.permission` mask match this order, and the fields are separated by a period (.). Each field can either be empty, or have `R`, `W`, `RW`, `X`, `RX`, `WX`, or `RWX` as value (R = read, W = write, X = execute or list directory content).

Table 15. Push-to-client group support

Key value	Is the related push-to-client function enabled?
False	No, disabled
True	Yes, enabled for all
LDAP	Yes, but availability is controlled by membership of LDAP groups
SAF	Yes, but availability is controlled by permit to security profiles

Note:

- In order to activate host-based client control, a `keymapping.xml` file must exist in `/var/rdz/pushtoclient`. This file is created and maintained by a Developer for System z client administrator.

- See "Push-to-client considerations" in the *Host Configuration Reference* (SC14-7290) for more information about host-based projects, host-based client configuration, and upgrade control.
- When a file is created, z/OS UNIX uses by default the effective UID (user ID) of the creating thread and the GID (group ID) of the owning directory, not the effective GID of the creating thread. Refer to "z/OS UNIX directory structure" in *Host Configuration Reference* (SC14-7290) for more information on how to change this behavior or how to adjust your host-based client control setup to get the desired GID assignment.

(Optional) ssl.properties, RSE SSL encryption

You will need assistance of a security administrator to complete this customization task, which requires the following resources or special customization tasks:

- LINKLIST update
- Security rule to add program controlled data sets
- (Optional) Security rule to add certificate for SSL

External (client-host) communication can be encrypted using SSL (Secure Socket Layer). This feature is disabled by default and is controlled by the settings in `ssl.properties`.

Note: Client authentication with an X.509 certificate requires the use of SSL encrypted communication.

`ssl.properties` is located in `/etc/rdz/`, unless you specified a different location when you customized and submitted job `FEK.SFEKSAMP(FEKSETUP)`. See "Customization setup" on page 13 for more details. You can edit the file with the TSO **OEDIT** command. Note that RSE must be restarted for the changes to take effect.

The client communicates with RSE daemon during connection setup and with RSE server during the actual session. Both data streams are encrypted when SSL is enabled.

RSE daemon and RSE server support different mechanisms to store certificates due to architectural differences between the two. This implies that SSL definitions are required for both RSE daemon and RSE server. A shared certificate can be used if RSE daemon and RSE server use the same certificate management method.

Table 16. SSL certificate storage mechanisms

Certificate storage	Created and managed by	RSE daemon	RSE server
key ring	SAF-compliant security product	supported	supported
key database	z/OS UNIX's gskkyman	supported	/
key store	Java's keytool	/	supported

Note:

- SAF-compliant key rings are the preferred method for managing certificates.

- SAF-compliant key rings can store the certificate's private key either in the security database or by using ICSF, the interface to System z cryptographic hardware. Access to ICSF is protected by profiles in the CSFSERV security class.

RSE daemon uses System SSL functions to manage SSL. This implies that SYS1.SIEALNKE must be program controlled by your security software and available to RSE using LINKLIST or the STEPLIB directive in rsed.envvars.

The following code sample shows the sample ssl.properties file, which must be customized to match your system environment. Comment lines start with a pound sign (#), when using a US code page. Data lines can only have a directive and its assigned value, comments are not allowed on the same line. Line continuations are not supported.

```
# ssl.properties – SSL configuration file
enable_ssl=false

# Daemon Properties

#daemon_keydb_file=
#daemon_keydb_password=
#daemon_key_label=

# Server Properties

#server_keystore_file=
#server_keystore_password=
#server_keystore_label=
#server_keystore_type=JCERACFKS
```

Figure 30. ssl.properties – SSL configuration file

The daemon and server properties only need to be set if you enable SSL. Refer to "Setting up SSL and X.509 authentication" in the *Developer for System z Host Configuration Reference* for more information about SSL setup.

enable_ssl

Enable or disable SSL communication. The default is false. The only valid options are true and false.

daemon_keydb_file

RACF (or similar security product) key ring name. Provide the key database name if you used **gskkyman** to create a key database instead of using a key ring. Uncomment and customize this directive if SSL is enabled.

daemon_keydb_password

Leave commented out or blank if you use a key ring, otherwise provide the key database password. Uncomment and customize this directive if SSL is enabled and you are using a **gskkyman** key database.

daemon_key_label

The certificate label used in the key ring or key database, if it is not defined as the default one. Must be commented out if the default is used. Uncomment and customize this directive if SSL is enabled and you are not using the default security certificate. Note that key labels are case sensitive.

server_keystore_file

Name of the key store created by Java's **keytool** command, or the RACF (or similar security product) key ring name if `server_keystore_type=JCERACFKS`. Uncomment and customize this directive if SSL is enabled.

server_keystore_password
Leave commented out or blank if you use a key ring, otherwise provide the key store password. Uncomment and customize this directive if SSL is enabled and you are using a **keytool** key store.

server_keystore_label
The certificate label used in the key ring or key store. The default is the first valid certificate encountered. Uncomment and customize this directive if SSL is enabled and you are not using the default security certificate. Note that key labels are case sensitive.

server_keystore_type
Key store type. The default is `JKS`. Valid values are:

Table 17. Valid keystore types

Keyword	Key store type
JKS	Java key store
JCERACFKS	SAF-compliant key ring, where the certificate's private key is stored in the security database.
JCECCARACFKS	SAF-compliant key ring, where the certificate's private key is stored using ICSF, the interface to System z cryptographic hardware.

Note: At the time of publication, IBM z/OS Java requires an update of the `/usr/lpp/java/J6.0/lib/security/java.security` file to support `JCECCARACFKS`. The following line must be added:

```
security.provider.1=com.ibm.crypto.hdwrCCA.provider.IBMJCECCA
```

The resulting file will look like this:

```
security.provider.1=com.ibm.crypto.hdwrCCA.provider.IBMJCECCA
security.provider.2=com.ibm.jsse2.IBMJSSEProvider2
security.provider.3=com.ibm.crypto.provider.IBMJCE
security.provider.4=com.ibm.security.jgss.IBMJGSSProvider
security.provider.5=com.ibm.security.cert.IBMCertPath
security.provider.6=com.ibm.security.sasl.IBMSASL
```

(Optional) rsecomm.properties, RSE tracing

This customization task does not require assistance, special resources, or special customization tasks.

Developer for System z supports different levels of tracing the internal program flow for problem solving purposes. RSE, and some of the services called by RSE, use the settings in `rsecomm.properties` to know the desired initial detail level in the output logs.

Attention: Changing these settings can cause performance degradations and should only be done under the direction of the IBM support center.

rsecomm.properties is located in /etc/rdz/, unless you specified a different location when you customized and submitted job FEK.SFEKSAMP(FEKSETUP). See "Customization setup" on page 13 for more details. You can edit the file with the TSO **OEDIT** command.

The following code sample shows the rsecomm.properties file, which can be customized to match your tracing needs. Comment lines start with a pound sign (#), when using a US code page. Data lines can only have a directive and its assigned value, comments are not allowed on the same line. Line continuations are not supported.

```
# server.version - DO NOT MODIFY!
server.version=5.0.0

# Logging level
# 0 - Log error messages
# 1 - Log error and warning messages
# 2 - Log error, warning and info messages
debug_level=1
```

Figure 31. rsecomm.properties – Logging configuration file

server.version

Logging server version. The default is 5.0.0. Do not modify.

debug_level

Detail level for output logs. The default is 1 (log error and warning messages). Note that debug_level controls the detail level of multiple services (and thus multiple output files). Increasing the detail level will cause performance degradations and should only be done under the direction of the IBM support center. Refer to "RSE tracing" in the *Host Configuration Reference* (SC14-7290) for more information about which logs are controlled by this directive.

The valid values are the following:

0	Log error messages only.
1	Log error and warning messages.
2	Log error, warning, and informational messages.

Note: debug_level can be changed dynamically for specific log files with the **modify rsecommlog**, **modify rseserverlog**, and **modify rsedaemonlog** operator commands, as described in Chapter 10, "Operator commands," on page 169.

(Optional) include.conf, Forced includes for C/C++ content assist

This customization task does not require assistance, special resources, or special customization tasks.

Content assist for C/C++ can use the definitions in include.conf to do forced includes of specified files or members. A forced include consists of a file or directory, data set, or data set member which will be parsed when a content assist operation is performed, regardless of whether that file or member was included in the source code using a pre-processor directive.

The file must be referenced in rsed.envvars by the include.c or include.cpp variables before it is used. This reference in rsed.envvars implies that you can specify a different file for usage by C and C++. The variables in rsed.envvars are disabled by default.

The sample include.conf is located in /etc/rdz/, unless you specified a different location when you customized and submitted job FEK.SFEKSAMP(FEKSETUP). See "Customization setup" on page 13 for more details. You can edit the file with the TSO **OEDIT** command.

Definitions must start in column 1. Comment lines start with a pound sign (#) when using a US code page. Data lines can only have the name of a directory, file, data set or member. Comments are not allowed on the same line. Line continuations are not supported.

```
# To include the stdio.h file from the /usr/include directory, input:
# /usr/include/stdio.h
#
# To include all files of the /usr/include directory and all of it's
# sub-directories, input:
# /usr/include
#
# Uncomment and customize variable FILETYPES to limit the z/OS UNIX
# wildcard include to selected (case sensitive) file types:
# The file types are specified in a comma-delimited list (no blanks)
# FILETYPES=H,h,hpp,C,c,cpp,cxx

# To include all members of the CBC.SCLBH.H data set, input:
# //CBC.SCLBH.H
#
# To include the STDIOSTR member of the CBC.SCLBH.H  data set, input:
# //CBC.SCLBH.H(STDIOSTR)
# The sample list contains some commonly used C standard library files
/usr/include/assert.h
/usr/include/ctype.h
/usr/include/errno.h
/usr/include/float.h
/usr/include/limits.h
/usr/include/locale.h
/usr/include/math.h
/usr/include/setjmp.h
/usr/include/signal.h
/usr/include/stdarg.h
/usr/include/stddef.h
/usr/include/stdio.h
/usr/include/stdlib.h
/usr/include/string.h
/usr/include/time.h
```

Figure 32. include.conf - Forced includes for C/C++ content assist

(Optional) DB2 stored procedure

You will need the assistance of a WLM administrator and a DB2 administrator to complete this customization task, which requires the following resources or special customization tasks:

- WLM update
- New PROCLIB member
- DB2 update

Developer for System z provides a sample DB2 stored procedure (PL/I and COBOL Stored Procedure Builder) for building COBOL and PL/I Stored Procedures from within the Developer for System z client.

Note: Sample members ELAXM* are located in FEK.#CUST.JCL and FEK.#CUST.PROCLIB, unless you specified a different location when you customized and submitted job FEK.SFEKSAMP(FEKSETUP). See “Customization setup” on page 13 for more details.

Workload Manager (WLM) changes

Use the workload management (WLM) panels to associate an application environment with the JCL procedure of the WLM address space for the PL/I and COBOL Stored Procedure Builder. Refer to *MVS Planning Workload Management* (SA22-7602) for information about how to do this.

Note: You can create a new application environment in WLM for the PL/I and COBOL Stored Procedure Builder, or you can add the necessary definitions to an existing one.

PROCLIB changes

Customize the sample Stored Procedure task FEK.#CUST.PROCLIB(ELAXMSAM), as described within the member, and copy it to SYS1.PROCLIB. As shown in the following code sample, you have to provide the following:

- The name of the application environment defined in WLM for this Stored Procedure
- The DB2 subsystem name
- The high-level qualifier of various data sets

```
//ELAXMSAM PROC RGN=0M,
//            NUMTCB=1,
//            APPLENV=#wlmwd4z,
//            DB2SSN=#ssn,
//            DB2PRFX='DSN810',
//            COBPRFX='IGY.V3R4M0',
//            PLIPRFX='IBMZ.V3R6M0',
//            LIBPRFX='CEE',
//            LODPRFX='FEK'
//*
//DSNX9WLM EXEC PGM=DSNX9WLM,REGION=&RGN,TIME=NOLIMIT,DYNAMNBR=10,
//            PARM='&DB2SSN,&NUMTCB,&APPLENV'
//STEPLIB  DD DISP=SHR,DSN=&DB2PRFX..SDSNEXIT
//         DD DISP=SHR,DSN=&DB2PRFX..SDSNLOAD
//         DD DISP=SHR,DSN=&LIBPRFX..SCEERUN
//         DD DISP=SHR,DSN=&COBPRFX..SIGYCOMP
//         DD DISP=SHR,DSN=&PLIPRFX..SIBMZCMP
//SYSEXEC  DD DISP=SHR,DSN=&LODPRFX..SFEKPROC
//SYSTSPRT DD SYSOUT=*
//CEEDUMP  DD SYSOUT=*
//SYSABEND DD DUMMY
//SYSUT1   DD UNIT=SYSALLDA,SPACE=(CYL,(1,1))
//SYSUT2   DD UNIT=SYSALLDA,SPACE=(CYL,(1,1))
//SYSUT3   DD UNIT=SYSALLDA,SPACE=(CYL,(1,1))
//SYSUT4   DD UNIT=SYSALLDA,SPACE=(CYL,(1,1))
//SYSUT5   DD UNIT=SYSALLDA,SPACE=(CYL,(1,1))
//SYSUT6   DD UNIT=SYSALLDA,SPACE=(CYL,(1,1))
//SYSUT7   DD UNIT=SYSALLDA,SPACE=(CYL,(1,1))
//*
```

Figure 33. ELAXMSAM - DB2 stored procedure task

Note:

- The DB2 stored procedure uses REXX exec ELAXMREX, located in FEK.SFEKPROC. Do not change this location if you want possible SMP/E maintenance to be activated automatically.
- See "Running multiple instances" in the *Host Configuration Reference* (SC14-7290) if you want to rename members ELAXMSAM or ELAXMREX.

DB2 changes

Customize and submit sample member ELAXMJCL in data set FEK.#CUST.JCL to define the Stored Procedure to DB2. Refer to the documentation within the member for customization instructions.

```
//ELAXMJCL JOB <job parameters>
//JOBLIB   DD DISP=SHR,DSN=#hlq.SDSNEXIT
//         DD DISP=SHR,DSN=#hlq.SDSNLOAD
//*
//RUNTIAD  EXEC PGM=IKJEFT01,DYNAMNBR=20
//SYSTSPRT DD SYSOUT=*
//SYSTSIN  DD *
 DSN S(#ssn) R(1) T(1)
 RUN PROGRAM(DSNTIAD) PLAN(#plan) -
 LIB('#hlq.RUNLIB.LOAD')
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
 CREATE PROCEDURE SYSPROC.ELAXMREX
  ( IN  FUNCTION_REQUEST   VARCHAR(20)      CCSID EBCDIC
  , IN  SQL_ROUTINE_NAME   VARCHAR(27)      CCSID EBCDIC
  , IN  SQL_ROUTINE_SOURCE VARCHAR(32672)   CCSID EBCDIC
  , IN  BIND_OPTIONS       VARCHAR(1024)    CCSID EBCDIC
  , IN  COMPILE_OPTIONS    VARCHAR(255)     CCSID EBCDIC
  , IN  PRECOMPILE_OPTIONS VARCHAR(255)     CCSID EBCDIC
  , IN  PRELINK_OPTIONS    VARCHAR(32672)   CCSID EBCDIC
  , IN  LINK_OPTIONS       VARCHAR(255)     CCSID EBCDIC
  , IN  ALTER_STATEMENT    VARCHAR(32672)   CCSID EBCDIC
  , IN  SOURCE_DATASETNAME VARCHAR(80)      CCSID EBCDIC
  , IN  BUILDOWNER         VARCHAR(8)       CCSID EBCDIC
  , IN  BUILDUTILITY       VARCHAR(18)      CCSID EBCDIC
  , OUT RETURN_VALUE       VARCHAR(255)     CCSID EBCDIC )
 PARAMETER STYLE GENERAL  RESULT SETS 1
 LANGUAGE REXX            EXTERNAL NAME   ELAXMREX
 COLLID   DSNREXCS        WLM ENVIRONMENT ELAXMSAM
 PROGRAM TYPE MAIN        MODIFIES SQL DATA
 STAY RESIDENT NO         COMMIT ON RETURN NO
 ASUTIME NO LIMIT         SECURITY USER;

 COMMENT ON PROCEDURE SYSPROC.ELAXMREX IS
 'PLI & COBOL PROCEDURE PROCESSOR (ELAXMREX), INTERFACE LEVEL 0.01';

 GRANT EXECUTE ON PROCEDURE SYSPROC.ELAXMREX TO PUBLIC;
//*
```

Figure 34. ELAXMJCL – DB2 stored procedure definition

Note: Make sure the WLM ENVIRONMENT clause in the CREATE PROCEDURE statement specifies the name of the WLM environment procedure which has been defined for the PL/I and COBOL Stored Procedure Builder (default ELAXMSAM).

(Optional) z/OS UNIX subprojects

This customization task does not require assistance, special resources, or special customization tasks.

REXEC (Remote Execution) is a TCP/IP service to let clients execute a command on the host. SSH (Secure Shell) is a similar service, but here all communication is encrypted using SSL (Secure Socket Layer). Developer for System z uses either service for doing remote (host-based) actions in z/OS UNIX subprojects.

Note:

- Developer for System z uses the z/OS UNIX version of REXEC, not the TSO version.
- If REXEC/SSH is not configured to use the default port, the Developer for System z client must define the correct port for use by z/OS UNIX subprojects.

This can be done by selecting the **Window > Preferences... > z/OS Solutions > USS Subprojects > Remote Action Options** preference page. Refer to “REXEC (or SSH) set up” to know which port is used.

REXEC (or SSH) set up

REXEC and SSH rely on services provided by INETD (Internet Daemon), which is another TCP/IP service. *Communications Server IP Configuration Guide* (SC31-8775) describes the steps required to set up INETD, REXEC, and SSH. For more details and alternate setup methods, refer to white paper *Using INETD, REXEC and SSH with Rational Developer for System z (SC14-7301)*, available in the Developer for System z library, http://www.ibm.com/software/rational/products/developer/systemz/library/index.html.

A common port used by REXEC is `512`. To verify this, you can check `/etc/inetd.conf` and `/etc/services` to find the port number used.

- Find the service name (1st word, `exec` in this example) of the `rexecd` server (7th word) in `/etc/inetd.conf`.

  ```
  exec  stream tcp nowait OMVSKERN /usr/sbin/orexecd rexecd –LV
  ```

- Find the port (2nd word, `512` in this example) attached to this service name (1st word) in `/etc/services`.

  ```
  exec      512/tcp      #REXEC   Command Server
  ```

The same principle applies to SSH. Its common port is `22`, and the server name is `sshd`.

(Optional) Include preprocessor support

This customization task does not require assistance, special resources, or special customization tasks.

Developer for System z supports interpreting and expanding COBOL and PL/I include statements, including select third-party include statements. Developer for System z also provides a sample REXX exec, `FEKRNPLI`, that can be invoked by the Developer for System z client to expand PL/I source by invoking the PL/I compiler.

`FEKRNPLI` is located in `FEK.#CUST.CNTL`, unless you specified a different location when you customized and submitted job `FEK.SFEKSAMP(FEKSETUP)`. See Chapter 2, “Basic customization,” on page 13 for more details.

Customize the sample `FEK.#CUST.CNTL(FEKRNPLI)` exec, as described within the member. You have to provide the following:

- `compiler_hlq`: The high-level qualifier for the PL/I compiler

The Developer for System z client uses the TSO Command Service to execute the exec. This implies that if the `FEKRNPLI` exec is placed in the `SYSPROC` or `SYSEXEC` concatenation for the TSO Command Service, the user does not need to know the exact location of the exec. The user only needs to know the name. By default, the TSO Command Service uses the ISPF Client Gateway to create a TSO environment, but APPC is also supported, as documented in the white paper, *Using APPC to provide TSO command services* (SC14-7291). When using the ISPF Client Gateway, the

SYSPROC or SYSEXEC concatenation is defined in ISPF.conf. See "ISPF.conf, ISPF's TSO/ISPF Client Gateway configuration file" on page 46 for more details on customizing this file.

(Optional) xUnit support for Enterprise COBOL and PL/I

This customization task does not require assistance, but does require the following resources or special customization tasks:

- LINKLIST update

Frameworks that assist developers in writing code to perform repeatable, self-checking unit tests are collectively known as xUnit. Developer for System z provides such a framework for unit testing of Enterprise COBOL and PL/I code, called zUnit.

In order to use the zUnit framework, developers need access to the AZU* and IAZU* load modules in the FEK.SFEKLOAD load library, either through STEPLIB or LINKLIST. The zUnit test runner, AZUTSTRN, in turn needs access to various system libraries, either through STEPLIB or LINKLIST:

- CEE.SCEERUN and CEE.SCEERUN2 (LE runtime)
- SYS1.CSSLIB (callable system services)
- SYS1.SIXMLOD1 (XML toolkit)

Note that the zUnit test runner also needs access to a load library that holds the different test cases. This library will likely be unique per developer.

The zUnit test runner, AZUTSTRN, can be invoked by the Developer for System z client in batch mode, from the TSO command line, and from the z/OS UNIX command line.

- Developer for System z provides a sample procedure, AZUZUNIT, to simplify invocation of the zUnit test runner in batch mode. AZUZUNIT is located in FEK.#CUST.PROCLIB, unless you specified a different location when you customized and submitted job FEK.SFEKSAMP(FEKSETUP). See Chapter 2, "Basic customization," on page 13 for more details.

  Customize the sample procedure, FEK.#CUST.PROCLIB(AZUZUNIT), as described within the member, and copy it to SYS1.PROCLIB.

  If the AZUZUNIT procedure cannot be copied into a system procedure library, ask the Developer for System z users to add a JCLLIB card (right after the JOB card) to their invocation job.

  ```
  //MYJOB    JOB <job parameters>
  //PROCS    JCLLIB ORDER=(FEK.#CUST.PROCLIB)
  ```

- For invocation of the zUnit test runner from z/OS UNIX (using the /usr/lpp/rdz/bin/zunit script), you can specify the required non-LINKLIST data sets in the STEPLIB directive of rsed.envvars, thus simplifying the setup for the developer.

  rsed.envvars is located in /etc/rdz/, unless you specified a different location when you customized and submitted job FEK.SFEKSAMP(FEKSETUP). See "Customization setup" on page 13 for more details. You can edit the file with the TSO **OEDIT** command.

  Note that the **zunit** script allows the user to specify data sets that will be added to the STEPLIB directive used by the script.

- For invocation of the zUnit test runner from the TSO command line (using the FEK.SFEKPROC(FEKZUNIT) exec), you should have the system libraries in LINKLIST. Otherwise developers must specify the system data set names on every invocation of the zUnit test runner. You can also write a wrapper exec that does the **TSOLIB** allocations of these data sets for them. (You can use FEKZUNIT itself as an example of how to code this wrapper exec.)

(Optional) Enterprise Service Tools support

This customization task does not require assistance, special resources, or special customization tasks.

The Developer for System z client has a code generation component called Enterprise Service Tools. Depending on the type of code being generated, this code relies on functions provided by the Developer for System z host install. Making these host functions available is described in the following sections:

- Chapter 5, "(Optional) Application Deployment Manager," on page 85
- "(Optional) CICS bidirectional language support"
- "(Optional) Diagnostic IRZ messages for generated code" on page 107

Note: Enterprise Service Tools encompasses multiple tools, such as the Service Flow Modeler (SFM) and XML Services for the Enterprise.

(Optional) CICS bidirectional language support

You will need the assistance of a CICS administrator to complete this customization task, which requires the following resources or special customization tasks:

- Update CICS region JCL
- Define a program to CICS

The Developer for System z Enterprise Service Tools component supports different formats of Arabic and Hebrew interface messages, as well as bidirectional data presentation and editing in all editors and views. In terminal applications, both left-to-right and right-to-left screens are supported, as well as numeric fields and fields with opposite-to-screen orientation.

Additional bidirectional features and functionality include the following:

- The Enterprise Service Tools service requestor dynamically specifies bidirectional attributes of interface messages.
- Bidirectional data processing in service flows is based on bidirectional attributes (text type, text orientation, numeric swapping, and symmetric swapping). These attributes can be specified in different stages of flow creation for both interface and terminal flows.
- Enterprise Service Tools-generated runtime code includes conversion of data between fields in messages that have different bidirectional attributes.

Additionally, Enterprise Service Tools-generated code can support bidi transformation in environments other than CICS SFR (Service Flow Runtime). One example is batch applications. You can make the Enterprise Service Tools generators to include calls to the bidirectional conversion routines by specifying the appropriate bidi transformation options in the Enterprise Service Tools generation wizards and linking the generated programs with the appropriate bidirectional conversion library, FEK.SFEKLOAD.

Perform the following tasks to activate CICS Bidirectional language support:

1. Place the FEK.SFEKLOAD load modules FEJBDCMP and FEJBDTRX in the CICS RPL concatenation (DD statement DFHRPL). You should do this by adding the installation data set to the concatenation so that applied maintenance is automatically available to CICS.

   Note: If you do not concatenate the installation data set but copy the modules into a new or existing data set, keep in mind that those modules are DLLs and MUST reside in a PDSE library.
2. Define FEJBDCMP and FEJBDTRX as programs to CICS using the appropriate CEDA command, for example:
   ```
   CEDA DEF PROG(FEJBDCMP) LANG(LE) G(xxx)
   CEDA DEF PROG(FEJBDTRX) LANG(LE) G(xxx)
   ```

(Optional) Diagnostic IRZ messages for generated code

This customization task does not require assistance, but does require the following resources or special customization tasks:

- LINKLIST update
- Update CICS region JCL

The Developer for System z client has a code generation component called Enterprise Service Tools. In order for code generated by Enterprise Service Tools to issue diagnostic error messages, all IRZM* and IIRZ* modules in the FEK.SFEKLMOD load library must be made available to the generated code. Enterprise Service Tools can generate code for the following environments:

- CICS
- IMS
- MVS batch

When the generated code is executed in a CICS transaction, then add all IRZM* and IIRZ* modules in FEK.SFEKLMOD to the DFHRPL DD of the CICS region. You should do this by adding the installation data set to the concatenation so that applied maintenance is automatically available.

In all other situations, make all IRZM* and IIRZ* modules in FEK.SFEKLMOD available either through STEPLIB or LINKLIST. You should do this by adding the installation data set to the concatenation so that applied maintenance is automatically available.

If you decide to use STEPLIB, you must define the modules not available through LINKLIST in the STEPLIB directive of the task that executes the code.

If the load modules are not available and an error is encountered by the generated code, then following message will be issued:

```
IRZ9999S Failed to retrieve the text of a Language Environment runtime
message. Check that the Language Environment runtime message module for
facility IRZ is installed in DFHRPL or STEPLIB.
```

Note:

- Module FEK.SFEKLMOD(IRZPWSIO) is statically linked during top-down IMS MPP code generation. Therefore, the module must not be available during runtime of the generated code. It should only be available during compile time.
- In version 8.5, the IRZ* and IIRZ* load modules and diagnostic messages have moved from load library FEK.SFEKLOAD to FEK.SFEKLMOD.

(Optional) Problem Determination Tools support

This customization task does not require assistance, special resources, or special customization tasks.

Developer for System z can integrate with various IBM z/OS Problem Determination Tools. The following sections describe how to make these tools available to the Developer for System z client:

- IBM Debug Tool for z/OS: See "(Optional) DB2 and IMS debug support."
- IBM File Manager for z/OS: See "(Optional) File Manager support" on page 109.
- IBM Fault Analyzer for z/OS: No Developer for System z host configuration required.

(Optional) DB2 and IMS debug support

This customization task does not require assistance, special resources, or special customization tasks for the Developer for System z configuration. However, there are requirements for the IBM Debug Tool for z/OS configuration.

IBM Debug Tool for z/OS provides a customized Language Environment (LE) user exit (CEEBXITA), which returns the TEST runtime options when called by the LE initialization logic in IMS and DB2 Stored Procedures. IBM Debug Tool for z/OS also provides the Debug Tool extension for the Problem Determination Tools Common Components server, to create and manage the TEST runtime options data set on the z/OS system. Developer for System z can use and enhance IBM Debug Tool for z/OS's support for managing debug profiles for the IMS and DB2 Stored Procedure run-times.

The IBM Debug Tool for z/OS documentation describes in detail the required setup, which is briefly mentioned here.

- Specifying the TEST runtime options through the Language Environment user exit (hlq.SEQA*)
- Adding support for the DTSP Profile view
  - Install the Problem Determination Tools Common Components Server (hlq.SIPV*, job IPVGSVRJ)
  - Install and configure the Debug Tool extension for the Problem Determination Tools Common Components (hlq.SEQA*)

Note:

- The IBM Debug Tool for z/OS product, version 11.1, must be ordered, installed, and configured separately. The installation and customization of this product is not described in this manual.
- The Developer for System z client does not use the DTSP Profile view plug-in for Eclipse.

- The Developer for System z client does not use the Language Environment user exit for regular batch-mode debugging.
- The Developer for System z client communicates directly with the Problem Determination Tools Common Components server, which implies that the end-user needs to know this port number, and that the port used by this server must be opened on your firewall protecting the z/OS host.

(Optional) File Manager support

This customization task does not require assistance, special resources, or special customization tasks for the Developer for System z configuration. However, there are requirements for the IBM File Manager for z/OS configuration.

Developer for System z's initial integration with IBM File Manager for z/OS has been marked as deprecated in Developer for System z version 8.0.3 and is no longer supported in version 8.5. The services provided by this function have been moved to different areas. Some functions, such as unformatted QSAM editing, are now part of regular data set handling by Developer for System z. More advanced functions, such as formatted data editing using copybooks or include files, require that the "IBM File Manager plug-in for Eclipse" is installed on the Developer for System z client. This plug-in is available from the IBM Problem Determination Tools Plug-ins web page, `http://www-01.ibm.com/software/awdtools/deployment/pdtplugins/`.

The IBM File Manager plug-in for Eclipse uses the File Manager Server (`FMNCAS`) to access the File Manager services. This server is not used by the File Manager ISPF panel interface. This implies that you have to perform additional File Manager setup tasks, specific for the File Manager Server. See your File Manager documentation for more details.

The port number used by the File Manager server must be specified in `rsed.envvars` directive `FM_SERVER_PORT`.

`rsed.envvars` is located in `/etc/rdz/`, unless you specified a different location when you customized and submitted job `FEK.SFEKSAMP(FEKSETUP)`. See "Customization setup" on page 13 for more details. You can edit the file with the TSO **OEDIT** command.

Note:

- The IBM File Manager for z/OS product, version 11.1, must be ordered, installed, and configured separately. The installation and customization of this product is not described in this manual.
- The Developer for System z client communicates directly with the File Manager server, which implies that the port used by this server must be opened on your firewall protecting the z/OS host.

(Optional) WORKAREA and /tmp cleanup

This customization task does not require assistance, special resources, or special customization tasks.

ISPF's TSO/ISPF Client Gateway and the SCLM Developer Toolkit function use the WORKAREA and /tmp directories to store temporary work files, which are removed before the session is closed. However, temporary output is sometimes left behind, for example, if there is a communication error while processing. For this reason, it is recommended that you clear out the WORKAREA and /tmp directories from time to time.

z/OS UNIX provides a shell script, skulker, that deletes files based upon the directory they are in and their age. Combined with the z/OS UNIX cron daemon, which runs commands at specified dates and times, you can set up an automated tool that periodically cleans out the WORKAREA and /tmp directories. Refer to *UNIX System Services Command Reference* (SA22-7802) for more information about the skulker script and the cron daemon.

Note: WORKAREA is located in /var/rdz/, unless you specified a different location when you customized and submitted job FEK.SFEKSAMP(FEKSETUP). See "Customization setup" on page 13 for more details.

Chapter 7. Installation verification

After completing the product customization, you can use the Installation Verification Programs (IVPs) described in this chapter to verify the successful setup of key product components.

Verify started tasks

JMON, JES Job Monitor

Start the `JMON` started task (or user job). The startup information in DD STDOUT should end with the following message:

```
JM200I Server initialization complete.
```

If the job ends with return code 66, then `FEK.SFEKAUTH` is not APF authorized.

Note: Start JES Job Monitor before continuing with the other IVP tests.

LOCKD, Lock daemon

Start the `LOCKD` started task (or user job). The lock daemon issues the following console message upon successful startup:

```
FEK501I Lock daemon started, port=4036, cleanup interval=1440,
 log level=1
```

Note: Start the lock daemon before continuing with the other IVP tests.

RSED, RSE daemon

Start the `RSED` started task (or user job) with the `IVP=IVP` parameter. With this parameter, the server will end after doing some installation verification tests. The output of these tests is available in DD STDOUT. In case of certain errors, data will also be available in DD STDERR. Check DD STDOUT for messages indicating that the following IVPs were successful:

- Java startup
- JES Job Monitor connection
- Lock daemon connection
- TCP/IP setup
- PassTicket generation

Note: The PassTicket IVP will fail with “`pthread_security_np error : EACCES`” if the user ID assigned to the RSED started task is protected, because no password activity is allowed for a protected ID. This failure might also happen if the RSED started task user ID has the same z/OS UNIX UID as another user ID which is protected, because there is no way to control which user ID will be used by your security software when UIDs are shared.

The STDOUT data should look like the following sample:

```
-------------------------------------------------------------
RSE daemon startup script
-------------------------------------------------------------

arguments: IVP -C/etc/rdz –P
```

```
RSE daemon IVP test

CDFMVS08 -- Fri Mar 23 17:50:52 2012 UTC
uid=8(STCRSE) gid=1(STCGROUP)

started from /usr/lpp/rdz/bin/rsed.sh
startup script version Jan09,2012

configuration files located in /etc/rdz –- startup argument
daemon port is 4035 -– set in rsed.envvars
debug level is 1 –- set in rsecomm.properties
TMPDIR=/tmp -- default

-------------------------------------------------------------
current environment variables
-------------------------------------------------------------
@="/usr/lpp/rdz/bin/rsed.sh" @[1]="-C/etc/rdz" @[2]="-P"
ANT_HOME="/usr/lpp/Apache/Ant/apache-ant-1.7.1"
CGI_DTWORK="/var/rdz"
CGI_ISPCONF="/etc/rdz"
CGI_ISPHOME="/usr/lpp/ispf"
CGI_ISPWORK="/var/rdz"
CGI_TRANTABLE="FEK.#CUST.LSTRANS.FILE"
CLASSPATH=".:/usr/lpp/rdz/lib:/usr/lpp/rdz/lib/dstore_core.jar:/usr/lpp/
ERRNO="0"
HOME="/tmp"
IFS="
"
JAVA_HOME="/usr/lpp/java/J6.0"
JAVA_PROPAGATE="NO"
LANG="C"
LIBPATH=".:/usr/lib:/usr/lpp/java/J6.0/bin:/usr/lpp/java/J6.0/bin/classi
LINENO="66"
LOGNAME="STCRSE"
MAILCHECK="600"
OLDPWD="/tmp"
OPTIND="1"
PATH=".:/usr/lpp/java/J6.0/bin:/usr/lpp/rdz/bin:/usr/lpp/ispf/bin:/bin:/
PPID="33554711"
PS1="\$ "
PS2="> "
PS3="#? "
PS4="+ "
PWD="/etc/rdz"
RANDOM="27298"
RSE_CFG="/etc/rdz"
RSE_HOME="/usr/lpp/rdz"
RSE_LIB="/usr/lpp/rdz/lib"
SECONDS="0"
SHELL="/bin/sh"
STEPLIB="NONE"
TMPDIR="/tmp"
TZ="EST5EDT"
X_ARG="-T"
X_C="-- startup argument"
X_KEY="-T"
X_L="-- set in rsecomm.properties"
X_LOG="1"
X_P="-- set in rsed.envvars"
X_PORT="4035"
X_VAL=""
_="-------------------------------------------------------------"
_BPX_SHAREAS="YES"
_BPX_SPAWN_SCRIPT="YES"
_CEE_DMPTARG="/tmp"
_CEE_RUNOPTS="ALL31(ON) HEAP(32M,32K,ANYWHERE,KEEP,,) TRAP(ON)"
```

```
_CMDSERV_BASE_HOME="/usr/lpp/ispf"
_CMDSERV_CONF_HOME="/etc/rdz"
_CMDSERV_WORK_HOME="/var/rdz"
_EDC_ADD_ERRNO2="1"
_RSE_ISPF_OPTS="&SESSION=SPAWN"
_RSE_DAEMON_CLASS="com.ibm.etools.zos.server.RseDaemon"
_RSE_DAEMON_IVP_TEST="1"
_RSE_HOST_CODEPAGE="IBM-1047"
_RSE_JAVAOPTS=" -DISPF_OPTS='&SESSION=SPAWN' -DA_PLUGIN_PATH=
_RSE_JMON_PORT="6715"
_RSE_LOCKD_CLASS="com.ibm.ftt.rse.mvs.server.miners.MVSLockDaemon"
_RSE_LOCKD_PORT="4036"
_RSE_LOG_LEVEL="1"
_RSE_POOL_SERVER_CLASS="com.ibm.etools.zos.server.ThreadPoolProcess"
_RSE_RSED_PORT="4035"
_RSE_SAF_CLASS="/usr/include/java_classes/IRRRacf.jar"
_RSE_SCRIPT_VERSION="Jan09,2012"
_RSE_SERVER_CLASS="org.eclipse.dstore.core.server.Server"
_RSE_SERVER_TIMEOUT="120000"
_SCLMDT_BASE_HOME="/usr/lpp/rdz"
_SCLMDT_CONF_HOME="/var/rdz/sclmdt"
_SCLMDT_TRANTABLE="FEK.#CUST.LSTRANS.FILE"
_SCLMDT_WORK_HOME="/var/rdz"
debug_level="1"

-------------------------------------------------------------
Address Space size limits
-------------------------------------------------------------
current address space size limit is 1913626624 (1825.0 MB)
maximum address space size limit is 2147483647 (2048.0 MB)

-------------------------------------------------------------
service history
-------------------------------------------------------------
Wed May 23 13:47:39 2012 -- COPY -- HHOP850 v8500 created 23 May 2012

-------------------------------------------------------------
java service level
-------------------------------------------------------------
java full version "J2RE 1.5.0 IBM z/OS build pmz31dev-20100813 (SR12

-------------------------------------------------------------
LE runtime options
-------------------------------------------------------------
Options Report for Enclave main 05/23/12 1:50:52 PM
Language Environment V01 R11.00

LAST WHERE SET                 OPTION
-------------------------------------------------------------------------------
Installation default             ABPERC(NONE)
Programmer default               ABTERMENC(RETCODE)
Installation default           NOAIXBLD
Invocation command               ALL31(ON)
Programmer default               ANYHEAP(32768,16384,ANYWHERE,FREE)
Installation default           NOAUTOTASK
Programmer default               BELOWHEAP(32768,16384,FREE)
Installation default             CBLOPTS(ON)
Installation default             CBLPSHPOP(ON)
Installation default             CBLQDA(OFF)
Installation default
CEEDUMP(60,SYSOUT=*,FREE=END,SPIN=UNALL
Installation default             CHECK(ON)
Installation default             COUNTRY(US)
Installation default           NODEBUG
Installation default             DEPTHCONDLMT(10)
Installation default             DYNDUMP(*USERID,NODYNAMIC,TDUMP)
Installation default             ENVAR("")
Installation default             ERRCOUNT(0)
Installation default             ERRUNIT(6)
Installation default             FILEHIST
Installation default             FILETAG(NOAUTOCVT,NOAUTOTAG)
Default setting                NOFLOW
Invocation command               HEAP(33554432,32768,ANYWHERE,KEEP,16384
Installation default             HEAPCHK(OFF,1,0,0,0)
Installation default             HEAPPOOLS(OFF,8,10,32,10,128,10,256,10,
Installation default             INFOMSGFILTER(OFF,,,,)
Installation default             INQPCOPN
Installation default             INTERRUPT(OFF)
Programmer default               LIBSTACK(32768,16384,FREE)
Installation default             MSGFILE(SYSOUT,FBA,121,0,NOENQ)
Installation default             MSGQ(15)
Installation default             NATLANG(ENU)
Ignored                        NONONIPTSTACK(See THREADSTACK)
Installation default             OCSTATUS
Installation default           NOPC
Installation default             PLITASKCOUNT(20)
Programmer default               POSIX(ON)
Installation default             PROFILE(OFF,"")
Installation default             PRTUNIT(6)
Installation default             PUNUNIT(7)
Installation default             RDRUNIT(5)
Installation default             RECPAD(OFF)
Invocation command               RPTOPTS(ON)
Installation default             RPTSTG(OFF)
Installation default           NORTEREUS
Installation default           NOSIMVRD
Programmer default
STACK(65536,65536,ANYWHERE,KEEP,524288,131072)
Installation default             STORAGE(NONE,NONE,NONE,0)
Installation default             TERMTHDACT(TRACE,,96)
Installation default           NOTEST(ALL,"*","PROMPT","INSPPREF")
Installation default             THREADHEAP(4096,4096,ANYWHERE,KEEP)
Installation default
THREADSTACK(OFF,4096,4096,ANYWHERE,KEEP,131072,
Installation default             TRACE(OFF,4096,DUMP,LE=0)
Invocation command               TRAP(ON,SPIE)
Installation default             UPSI(00000000)
Installation default           NOUSRHDLR(,)
Installation default             VCTRSAVE(OFF)
Installation default             XPLINK(OFF)
Installation default             XUFLOW(AUTO)

-------------------------------------------------------------
java startup test...
-------------------------------------------------------------
java full version "J2RE 1.5.0 IBM z/OS build pmz31dev-20100701a (SR12
java version "1.5.0"
Java(TM) 2 Runtime Environment, Standard Edition (build pmz31dev-2010070

IBM J9 VM (build 2.3, J2RE 1.5.0 IBM J9 2.3 z/OS s390-31 j9vmmz3123-2010
J9VM - 20100629_60535_bHdSMr
JIT  - 20100623_16197_r8
GC   - 20100211_AA)
JCL  - 20100813

-------------------------------------------------------------
JES Job Monitor test...
-------------------------------------------------------------

executed on CDFMVS08 -- Fri Mar 23 17:50:52 EDT 2012
executed by uid=8(STCRSE) gid=1(STCGROUP)
using /etc/rdz/rsed.envvars

current address space size limit is 1913626624 (1825.0 MB)
```

```
maximum address space size limit is 2147483647 (2048.0 MB)

testing JES Job Monitor on port 6715...
hostName=CDFMVS08
hostAddr=9.42.112.75
Waiting for JES Job Monitor response...
ACKNOWLEDGE01v03
Success

--------------------------------------------------------------
Lock Daemon test...
--------------------------------------------------------------

executed on CDFMVS08 -- Fri Mar 23 17:50:53 EDT 2012
executed by uid=8(STCRSE) gid=1(STCGROUP)
using /etc/rdz/rsed.envvars

current address space size limit is 1913626624 (1825.0 MB)
maximum address space size limit is 2147483647 (2048.0 MB)

testing RSE Lock Daemon on port 4036...
hostName=CDFMVS08
hostAddr=9.42.112.75

Registering user STCRSE to Lock Daemon...
Waiting for Lock Daemon response...

Querying to Lock Daemon...
Waiting for Lock Daemon response...
STCRSE

Unregistering user STCRSE from Lock Daemon...
Waiting for Lock Daemon response...

Querying to Lock Daemon...
Waiting for Lock Daemon response...

Success

--------------------------------------------------------------
TCP/IP IVP test...
--------------------------------------------------------------

executed on CDFMVS08 -- Fri Mar 23 17:50:53 EDT 2012
executed by uid=8(STCRSE) gid=1(STCGROUP)
using /etc/rdz/rsed.envvars

current address space size limit is 1913626624 (1825.0 MB)
maximum address space size limit is 2147483647 (2048.0 MB)
--------------------------------------------------------------
TCP/IP resolver configuration (z/OS UNIX search order):
--------------------------------------------------------------
Resolver Trace Initialization Complete -> 2012/05/23 17:50:54.208378

res_init Resolver values:
 Global Tcp/Ip Dataset  = None
 Default Tcp/Ip Dataset = None
 Local Tcp/Ip Dataset   = /etc/resolv.conf
 Translation Table      = Default
 UserId/JobName         = STCRSE
 Caller API             = LE C Sockets
 Caller Mode            = EBCDIC
 (L) DataSetPrefix = TCPIP
 (L) HostName      = CDFMVS08
 (L) TcpIpJobName  = TCPIP
 (L) DomainOrigin  = RALEIGH.IBM.COM
 (L) NameServer    = 9.42.206.2
```

```
                   9.42.206.3
 (L) NsPortAddr    = 53           (L) ResolverTimeout    = 10
 (L) ResolveVia    = UDP          (L) ResolverUdpRetries = 1
 (*) Options NDots = 1
 (*) SockNoTestStor
 (*) AlwaysWto     = NO           (L) MessageCase        = MIXED
 (*) LookUp        = DNS LOCAL
res_init Succeeded
res_init Started: 2012/05/23 17:50:54.229888
res_init Ended: 2012/05/23 17:50:54.229898
************************************************************************
MVS TCP/IP NETSTAT CS V1R11       TCPIP Name: TCPIP           17:50:54
Tcpip started at 11:31:40 on 05/23/2012 with IPv6 enabled

-------------------------------------------------------------
host IP address:
-------------------------------------------------------------
hostName=CDFMVS08
hostAddr=9.42.112.75
bindAddr=9.42.112.75
localAddr=9.42.112.75

Success, addresses match

-------------------------------------------------------------
PassTicket IVP test...
-------------------------------------------------------------

executed on CDFMVS08 -- Fri Mar 23 17:50:52 EDT 2012
executed by uid=8(STCRSE) gid=1(STCGROUP)
the default applid=FEKAPPL
Success, PassTicket IVP finished normally

-------------------------------------------------------------
RSE daemon IVP ended  -- return code 0 -- Fri Mar 23 17:50:55 EDT 2012
-------------------------------------------------------------
```

Note: Start the RSE daemon, without the IVP parameter, before continuing with the other IVP tests. RSE daemon issues the following console message upon successful startup:

```
FEK002I RseDaemon started. (port=4035)
```

IVP operator commands

An active RSE daemon supports the **IVP** modify command, which allows you to do selected IVPs from the console.

PassTicket reusability

Developer for System z requires that the PassTickets it generates are reusable, because PassTicket generation is limited to one per user per second. Verify PassTicket reusability by executing the following operator command. Replace `userid` with a valid TSO user ID.

```
MODIFY RSED,APPL=IVP PASSTICKET,userid
```

The command should return an output like that in the following sample:

```
MODIFY RSED,APPL=IVP PASSTICKET,IBMUSER

+FEK900I PASSTICKET IVP: the default applid=FEKAPPL
+FEK900I PASSTICKET IVP: Success, PassTicket IVP finished normally
+FEK901I PASSTICKET IVP  Exit code = 0
```

RSE daemon connection

Verify the RSE daemon connection by executing the following command. Replace `userid` with a valid TSO user ID.

```
MODIFY RSED,APPL=IVP DAEMON,userid
```

Note that this command is functionally identical to the `fekfivpd` IVP described in "Verify services" on page 118, but with the benefit that no password is required. RSE will generate a PassTicket and use this as password. The command should return an output like that in the following sample:

```
F RSED,APPL=IVP DAEMON,IBMUSER

+FEK900I DAEMON IVP: SSL is disabled
+FEK900I DAEMON IVP: connected
+FEK900I DAEMON IVP: 1343
+FEK900I DAEMON IVP: 8878350
+FEK900I DAEMON IVP: Success
+FEK901I DAEMON IVP  Exit code = 0
```

ISPF Client Gateway

Verify the ISPF Client Gateway connection by executing the following command. Replace `userid` with a valid TSO user ID.

```
MODIFY RSED,APPL=IVP ISPF,userid
```

Note that this command is functionally identical to the `fekfivpi` IVP described in "Verify services" on page 118. The command should return an output like that in the following sample:

```
F RSED,APPL=IVP ISPF,IBMUSER

+FEK900I ISPF IVP: executed on CDFMVS08 -- Tue Sep 13 22:29:28 EDT 2011
+FEK900I ISPF IVP: executed by uid=1(IBMUSER) gid=0(SYS1)
+FEK900I ISPF IVP: using /etc/rdz/rsed.envvars
+FEK900I ISPF IVP: current address space size limit is 2147483647
(2048.0 MB)
+FEK900I ISPF IVP: maximum address space size limit is 2147483647
(2048.0 MB)
+FEK900I ISPF IVP: -----------------------------------------------------
--------
+FEK900I ISPF IVP: /etc/rdz/ISPF.conf content:
+FEK900I ISPF IVP: -----------------------------------------------------
--------
+FEK900I ISPF IVP: ispllib=ISP.SISPLOAD
+FEK900I ISPF IVP: ispmlib=ISP.SISPMENU
+FEK900I ISPF IVP: isptlib=ISP.SISPTENU
+FEK900I ISPF IVP: ispplib=ISP.SISPPENU
+FEK900I ISPF IVP: ispslib=ISP.SISPSLIB
+FEK900I ISPF IVP: sysproc=ISP.SISPCLIB,FEK.SFEKPROC
+FEK900I ISPF IVP: -----------------------------------------------------
--------
+FEK900I ISPF IVP: Host install verification for RSE
+FEK900I ISPF IVP: Review IVP log messages from HOST below :
+FEK900I ISPF IVP: -----------------------------------------------------
--------
+FEK900I ISPF IVP: Service level 22Feb2011
+FEK900I ISPF IVP: RSE connection and base TSO/ISPF session initializati
on check only
+FEK900I ISPF IVP: *** CHECK : ENVIRONMENT VARIABLES - key variables
displayed below :
+FEK900I ISPF IVP: Server PATH         = .:/usr/lpp/java/J6.0/bin:/usr/l
pp/rdz/bin:/usr/lpp/ispf/bin:/bin:/usr/sbin
+FEK900I ISPF IVP: STEPLIB             = NONE
+FEK900I ISPF IVP: Temporary directory = /tmp
```

```
+FEK900I ISPF IVP: CGI_ISPHOME  = /usr/lpp/ispf
+FEK900I ISPF IVP: CGI_ISPCONF  = /etc/rdz
+FEK900I ISPF IVP: CGI_ISPWORK  = /var/rdz
+FEK900I ISPF IVP: ----------------------------------------------------
--------
+FEK900I ISPF IVP: *** CHECK : USS MODULES
+FEK900I ISPF IVP: Checking ISPF Directory : /usr/lpp/ispf
+FEK900I ISPF IVP: Checking modules in /usr/lpp/ispf/bin directory
+FEK900I ISPF IVP: Checking for ISPF configuration file ISPF.conf
+FEK900I ISPF IVP: RC=0
+FEK900I ISPF IVP: MSG: SUCCESSFUL
+FEK900I ISPF IVP: ----------------------------------------------------
--------
+FEK900I ISPF IVP: *** CHECK : TSO/ISPF INITIALIZATION
+FEK900I ISPF IVP: ( TSO/ISPF session will be initialized )
+FEK900I ISPF IVP: RC=0
+FEK900I ISPF IVP: MSG: SUCCESSFUL
+FEK900I ISPF IVP: ----------------------------------------------------
--------
+FEK900I ISPF IVP: *** CHECK: Shutting down TSO/ISPF IVP session
+FEK900I ISPF IVP: RC=0
+FEK900I ISPF IVP: MSG: SUCCESSFUL
+FEK900I ISPF IVP: ----------------------------------------------------
--------
+FEK900I ISPF IVP: Host installation verification completed successfully
+FEK900I ISPF IVP: ----------------------------------------------------
--------
+FEK901I ISPF IVP  Exit code = 0
```

Verify services

The Developer for System z installation provides several Installation Verification Programs (IVP) for the basic and optional services. The IVP scripts are located in the installation directory, default /usr/lpp/rdz/bin/.

Table 18. IVPs for services

fekfivpc	"(Optional) CARMA connection" on page 124
fekfivpd	"RSE daemon connection" on page 121
fekfivpi	"ISPF's TSO/ISPF Client Gateway connection" on page 123
fekfivpj	"JES Job Monitor connection" on page 122
fekfivpl	"Lock daemon connection" on page 122
fekfivps	"(Optional) SCLMDT connection" on page 125
fekfivpt	"TCP/IP setup" on page 120

The tasks described below expect you to be active in z/OS UNIX. This can be done by issuing the TSO command **OMVS**. Use the **exit** command to return to TSO.

A large region size is required for the user ID that executes the IVPs, because functions such as Java, which require a lot of memory, will be executed. You should set the region size to 131072 kilobytes (128 megabytes) or higher.

The following sample error is a clear indication of an insufficient region size. (But other errors can occur, too. For example, Java might fail to start.)

```
CEE5213S The signal SIGPIPE was received.
%z/OS UNIX command%: command was killed by signal number 13
    %line-number% *-*   %REXX command%
       +++ RC(137) +++
```

Note: The Developer for System z started tasks must be active before starting the IVP test.

IVP initialization

All sample commands in this section expect that certain environment variables are set. This way, the IVP scripts are available through the PATH statement and the location of the customized configuration files is known. Use the **pwd** and **cd** commands to verify and change your current directory to the directory with the customized configuration files. The ivpinit shell script can then be used to set the RSE environment variables, such as in the following sample ($ is the z/OS UNIX prompt):

```
$ pwd
/u/userid
$ cd /etc/rdz
$ . ./ivpinit
RSE configuration files located in /etc/rdz --default
added /usr/lpp/rdz/bin to PATH
```

The first "." (dot) in . ./ivpinit is a z/OS UNIX command to run the shell in the current environment, so that the environment variables set in the shell are effective even after exiting the shell. The second one is referring to the current directory.

Note:

- If . ./ivpinit is NOT executed before the fekfivp* scripts, the path to these scripts must be specified when calling them, as in the following sample:

  ```
  /usr/lpp/rdz/bin/fekfivpr 512 USERID
  ```

  Also, all fekfivp* scripts will ask for the location of the customized rsed.envvars if . ./ivpinit is not executed first.
- Some IVP tests use the TCP/IP REXX socket API, which requires that the TCP/IP load library, default TCPIP.SEZALOAD, is in LINKLIST or STEPLIB. The following commands might be necessary to be able to execute these IVP tests ($ is the z/OS UNIX prompt):

  ```
  $ EXPORT STEPLIB=$STEPLIB:TCPIP.SEZALOAD
  ```

  Note that adding a non-APF authorized library to an existing STEPLIB removes the APF authorization for existing STEPLIB data sets.

  Also note that if CEE.SCEELKED is in LINKLIST or STEPLIB, TCPIP.SEZALOAD must be placed before CEE.SCEELKED. Failure to do so will result in a 0C1 system abend for the TCP/IP REXX socket calls.

For information about diagnosing RSE connection problems, see "Troubleshooting configuration problems" in the *Host Configuration Reference* (SC14-7290) and the Technotes on the Developer for System z Support Page http://www-306.ibm.com/software/awdtools/rdz/support/.

Port availability

The JES Job Monitor, RSE daemon, and Lock daemon port availability can be verified by issuing the **netstat** command. The result should show the ports used by these services, as in the following samples ($ is the z/OS UNIX prompt):

IPv4

```
$ netstat
MVS TCP/IP NETSTAT CS VxRy    TCPIP Name: TCPIP       13:57:36
User Id  Conn     Local Socket      Foreign Socket    State
-------  ----     ------------      --------------    -----
RSED     0000004B 0.0.0.0..4035     0.0.0.0..0        Listen
LOCKD    0000004C 0.0.0.0..4036     0.0.0.0..0        Listen
JMON     00000037 0.0.0.0..6715     0.0.0.0..0        Listen
```

IPv6

```
$ netstat
MVS TCP/IP NETSTAT CS VxRy    TCPIP Name: TCPIP       14:03:35
User Id  Conn     State
-------  ----     -----
RSED     0000004B Listen
  Local Socket:   0.0.0.0..4035
  Foreign Socket: 0.0.0.0..0
LOCKD    0000004C Listen
  Local Socket:   0.0.0.0..4036
  Foreign Socket: 0.0.0.0..0
JMON     00000037 Listen
  Local Socket:   0.0.0.0..6715
  Foreign Socket: 0.0.0.0..0
```

TCP/IP setup

Developer for System z is dependent upon TCP/IP having the correct hostname when it is initialized. This implies that the different TCP/IP and Resolver configuration files must be set up correctly. Refer to "Setting up TCP/IP" in the *Host Configuration Reference* (SC14-7290) for more information about TCP/IP and Resolver setup. Verify the current settings by executing the following command:

```
fekfivpt
```

Note: This IVP issues the TCPIP **netstat -u** command, which might be protected against execution by your security software (`EZB.NETSTAT.mvsname.tcpprocname.UP` profile in the `SERVAUTH` class).

The command should return an output like that in this sample ($ is the z/OS UNIX prompt):

```
$ fekfivpt

executed on CDFMVS08 -- Wed Jul  2 13:11:54 EDT 2008
executed by uid=1(USERID) gid=0(GROUP)
using /etc/rdz/rsed.envvars

current address space size limit is 1914675200 (1826.0 MB)
maximum address space size limit is 2147483647 (2048.0 MB)

-------------------------------------------------------------
TCP/IP resolver configuration (z/OS UNIX search order):
-------------------------------------------------------------
Resolver Trace Initialization Complete -> 2008/07/02 13:11:54.745964

res_init Resolver values:
 Global Tcp/Ip Dataset  = None
 Default Tcp/Ip Dataset = None
 Local Tcp/Ip Dataset   = /etc/resolv.conf
 Translation Table      = Default
 UserId/JobName         = USERID
 Caller API             = LE C Sockets
 Caller Mode            = EBCDIC
 (L) DataSetPrefix = TCPIP
 (L) HostName      = CDFMVS08
 (L) TcpIpJobName  = TCPIP
 (L) DomainOrigin  = RALEIGH.IBM.COM
 (L) NameServer    = 9.42.206.2
                       9.42.206.3
 (L) NsPortAddr    = 53           (L) ResolverTimeout    = 10
 (L) ResolveVia    = UDP          (L) ResolverUdpRetries = 1
 (*) Options NDots = 1
 (*) SockNoTestStor
 (*) AlwaysWto     = NO           (L) MessageCase        = MIXED
 (*) LookUp        = DNS LOCAL
res_init Succeeded
res_init Started: 2008/07/02 13:11:54.755363
res_init Ended: 2008/07/02 13:11:54.755371
************************************************************************
MVS TCP/IP NETSTAT CS V1R9       TCPIP Name: TCPIP           13:11:54
Tcpip started at 01:28:36 on 06/23/2008 with IPv6 enabled

-------------------------------------------------------------
host IP address:
-------------------------------------------------------------
hostName=CDFMVS08
hostAddr=9.42.112.75
bindAddr=9.42.112.75
localAddr=9.42.112.75

Success, addresses match
```

RSE daemon connection

Verify the RSE daemon connection by executing the following command.

```
fekfivpd
```

After prompting you for a password, the command should return an output similar to what appears in the following sample ($ is the z/OS UNIX prompt):

```
$ fekfivpd

executed on CDFMVS08 -- Wed Jul  2 15:00:27 EDT 2008
executed by uid=1(USERID) gid=0(GROUP)
using /etc/rdz/rsed.envvars

current address space size limit is 1914675200 (1826.0 MB)
maximum address space size limit is 2147483647 (2048.0 MB)

attempting to connect userid USERID using port 4035 ...

Password:
SSL is disabled
connected
8108
570655399
Success
```

When testing an SSL-enabled connection, ensure that the user ID executing the IVP has access to all certificates that are involved (including the CA certificates used to sign the Developer for System z certificate). Note that the operator command version of this IVP (`F RSED,APPL=IVP DAEMON,userid`) uses the SSL setup done for RSE host, and is therefore less error prone. The following list documents some common SSL-related errors:

- Verify that the user ID executing the IVP has access to all certificates that are involved if you get this error message: `gsk_environment_init() failed: Error detected while opening the certificate data base`

- Verify that the signing CA certificates are also on the keyring if you get this error message: `gsk_secure_socket_init() failed: Certificate validation error`

JES Job Monitor connection

Verify the JES Job Monitor connection by executing the following command.

```
fekfivpj
```

The command should return the JES Job Monitor acknowledge message, similar to what appears in the following sample ($ is the z/OS UNIX prompt):

```
$ fekfivpj

executed on CDFMVS08 -- Wed Jul  2 15:00:27 EDT 2008
executed by uid=1(USERID) gid=0(GROUP)
using /etc/rdz/rsed.envvars

current address space size limit is 1914675200 (1826.0 MB)
maximum address space size limit is 2147483647 (2048.0 MB)

testing JES Job Monitor on port 6715...
hostName=CDFMVS08
hostAddr=9.42.112.75
Waiting for JES Job Monitor response...
ACKNOWLEDGE01v03

Success
```

Lock daemon connection

Verify the lock daemon connection by executing the following command.

```
fekfivpl
```

The command should return an output similar to what appears in the following sample ($ is the z/OS UNIX prompt):

```
$ fekfivpl

executed on CDFMVS08 -- Mon Jun 29 08:00:38 EDT 2009
executed by uid=1(USERID) gid=0(GROUP)
using /etc/rdz/rsed.envvars

current address space size limit is 1914675200 (1826.0 MB)
maximum address space size limit is 2147483647 (2048.0 MB)

testing RSE Lock Daemon on port 4036...
hostName=CDFMVS08
hostAddr=9.42.112.75

Registering user to Lock Daemon...
Waiting for Lock Daemon response...

Querying to Lock Daemon...
Waiting for Lock Daemon response...
USERID


Unregistering user from Lock Daemon...
Waiting for Lock Daemon response...

Querying to Lock Daemon...
```

```
Waiting for Lock Daemon response...


Success
```

ISPF's TSO/ISPF Client Gateway connection

Verify the connection to ISPF's TSO/ISPF client Gateway by executing the following command:

```
fekfivpi
```

The command should return the result of ISPF's TSO/ISPF client Gateway-related checks (variables, HFS modules, starting and stopping TSO/ISPF session), similar to what appears in the following sample ($ is the z/OS UNIX prompt):

```
$ fekfivpi

executed on CDFMVS08 -- Wed Jul  2 15:00:27 EDT 2008
executed by uid=1(USERID) gid=0(GROUP)
using /etc/rdz/rsed.envvars

current address space size limit is 1914675200 (1826.0 MB)
maximum address space size limit is 2147483647 (2048.0 MB)

-------------------------------------------------------------
/etc/rdz/ISPF.conf content:
-------------------------------------------------------------

ispmlib=ISP.SISPMENU
isptlib=ISP.SISPTENU
ispplib=ISP.SISPPENU
ispslib=ISP.SISPSLIB
ispllib=ISP.SISPLOAD
sysproc=ISP.SISPCLIB,FEK.SFEKPROC

-------------------------------------------------------------
Host install verification for RSE
Review IVP log messages from HOST below :
-------------------------------------------------------------

RSE connection and base TSO/ISPF session initialization check only

*** CHECK : ENVIRONMENT VARIABLES - key variables displayed below :

Server PATH         =
/usr/lpp/java/J6.0/bin:/usr/lpp/rdz/lib:/usr/lpp/ispf/bin:
/bin:/usr/sbin:.

STEPLIB             = FEK.SFEKAUTH:FEK.SFEKLOAD

CGI_ISPHOME  = /usr/lpp/ispf
CGI_ISPCONF  = /etc/rdz
CGI_ISPWORK  = /var/rdz
-------------------------------------------------------------
*** CHECK : USS MODULES
Checking ISPF Directory : /usr/lpp/ispf
Checking modules in /usr/lpp/ispf/bin directory
Checking for ISPF configuration file ISPF.conf
RC=0
MSG: SUCCESSFUL

-------------------------------------------------------------
*** CHECK : TSO/ISPF INITIALIZATION
( TSO/ISPF session will be initialized )
RC=0
MSG: SUCCESSFUL
```

```
-------------------------------------------------------------
*** CHECK: Shutting down TSO/ISPF IVP session
RC=0
MSG: SUCCESSFUL

-------------------------------------------------------------
Host installation verification completed successfully
-------------------------------------------------------------
```

Note: If any of the ISPF checks fail, more detailed information will be shown.

`fekfivpi` has the following optional, non-positional, parameters:

-file `fekfivpi` can produce large amounts of output (hundreds of lines). The `-file` parameter sends this output to a file, `$TMPDIR/fekfivpi.log`, where `$TMPDIR` is the value of the `TMPDIR` directive in `rsed.envvars` (default `/tmp`).

-debug
The `-debug` parameter creates detailed test output. Do not use this option unless directed by the IBM support center.

(Optional) CARMA connection

Verify the connection to CARMA by executing the following command:

```
fekfivpc
```

The command should return the result of CARMA related checks, like in the following sample ($ is the z/OS UNIX prompt):

```
$ fekfivpc

executed on CDFMVS08 -- Fri Aug 20 14:15:46 EDT 2010
executed by uid=1(USERID) gid=0(GROUP)
using /etc/rdz/rsed.envvars

current address space size limit is  140484608 ( 134.0 MB)
maximum address space size limit is 2147483647 (2048.0 MB)

*** /etc/rdz/CRASRV.properties content:
port.start = 5227
port.range = 100
startup.script.name = /usr/lpp/rdz/bin/carma.startup.rex
clist.dsname = *CRASTART
crastart.stub = /usr/lpp/rdz/bin/CRASTART
crastart.configuration.file = /etc/rdz/crastart.endevor.conf
crastart.syslog = Partial
crastart.timeout = 420

*** Creating /tmp/fekfivpc.log

*** Verifying CARMA installation...
1. Creating CARMA connection (timeout after 60 seconds)
2. Initializing CARMA
3. Retrieving RAM list
   The following RAMs were found
        00 CA Endevor SCM       Unique ID: COM.IBM.CARMA.ENDEVORRAM
4. Getting customization data for RAM 00
5. Initializing RAM 00
6. Retrieving Repository Instance List
   Found 6 Repository Instance(s)
7. Terminating RAM 00
8. Terminating CARMA

***  IVP Successful!!!!
```

Note: Verify the content of `/tmp/fekfivpc.log` if the IVP fails. This log documents the communication between RSE and CARMA and can hold valuable clues to find the root cause of the failure.

`fekfivpc` has the following optional, non-positional, parameters:

-noram
: By default, `fekfivpc` will start the first RAM that is defined in the CRADEF VSAM data set. If you do not want to test the RAM (for example, a third party RAM is listed first, and it requires unexpected input), you can use the `–noram` startup argument to skip the RAM specific steps (step 4 through 7) of the IVP test.

(Optional) SCLMDT connection

Verify the connection to SCLM Developer Toolkit by executing the following command:

```
fekfivps
```

The command should return the result of SCLM Developer Toolkit related checks (variables, HFS modules, REXX runtime, starting and stopping TSO/ISPF session), like that in the following sample ($ is the z/OS UNIX prompt):

```
$ fekfivps

executed on CDFMVS08 -- Wed Jul  2 15:00:27 EDT 2008
executed by uid=1(USERID) gid=0(GROUP)
using /etc/rdz/rsed.envvars

current address space size limit is 1914675200 (1826.0 MB)
maximum address space size limit is 2147483647 (2048.0 MB)

-------------------------------------------------------------
/etc/rdz/ISPF.conf content:
-------------------------------------------------------------

ispmlib=ISP.SISPMENU
isptlib=ISP.SISPTENU
ispplib=ISP.SISPPENU
ispslib=ISP.SISPSLIB
ispllib=ISP.SISPLOAD
sysproc=ISP.SISPCLIB,FEK.SFEKPROC
-------------------------------------------------------------
Host install verification for RSE
Review IVP log messages from HOST below :
-------------------------------------------------------------

*** CHECK : ENVIRONMENT VARIABLES - key variables displayed below :

Server PATH         = /usr/lpp/java/J6.0/bin:/usr/lpp/rdz/lib:/usr/lpp/ispf/bin:
/bin:/usr/sbin:.

STEPLIB             = FEK.SFEKAUTH:FEK.SFEKLOAD

CGI_ISPHOME  = /usr/lpp/ispf
CGI_ISPCONF  = /etc/rdz
CGI_ISPWORK  = /var/rdz
_SCLMDT_CONF_HOME  = /var/rdz/sclmdt
_SCLMDT_WORK_HOME  = /var/rdz
_SCLMDT_TRANTABLE  = FEK.#CUST.LSTRANS.FILE

-------------------------------------------------------------
*** CHECK : JAVA PATH SETUP VERIFICATION
RC=0
```

```
MSG: SUCCESSFUL

-------------------------------------------------------------
*** CHECK : USS MODULES
Checking ISPF Directory : /usr/lpp/ispf
Checking modules in /usr/lpp/ispf/bin directory
Checking for ISPF configuration file ISPF.conf
Checking install bin Directory : /usr/lpp/rdz/bin
RC=0
MSG: SUCCESSFUL

-------------------------------------------------------------
*** CHECK : REXX RUNTIME ENVIRONMENT
RC=0
MSG: SUCCESSFUL

-------------------------------------------------------------
*** CHECK : TSO/ISPF INITIALIZATION
( TSO/ISPF session will be initialized )
RC=0
MSG: SUCCESSFUL

-------------------------------------------------------------
*** CHECK: Shutting down TSO/ISPF IVP session
RC=0
MSG: SUCCESSFUL

-------------------------------------------------------------
Host installation verification completed successfully
-------------------------------------------------------------
```

Note: If any of the SCLMDT checks fail, more detailed information will be shown.

`fekfivps` has the following optional, non-positional, parameters:

-file `fekfivps` can produce large amounts of output (hundreds of lines). The `-file` parameter sends this output to a file, `$TMPDIR/fekfivps.log`, where `$TMPDIR` is the value of the `TEMPDIR` directive in `rsed.envvars` (default `/tmp`).

-debug
The `-debug` parameter creates detailed test output. Do not use this option unless directed by the IBM support center.

Chapter 8. Security definitions

Customize and submit sample member FEKRACF, which has sample RACF and z/OS UNIX commands to create the basic security definitions for Developer for System z.

FEKRACF is located in FEK.#CUST.JCL, unless you specified a different location when you customized and submitted job FEK.SFEKSAMP(FEKSETUP). See "Customization setup" on page 13 for more details.

Refer to the *RACF Command Language Reference* (SA22–7687), for more information about RACF commands.

Note:

- For those sites that use CA ACF2™ for z/OS, please refer to your product page on the CA support site (https://support.ca.com) and check for the related Developer for System z Knowledge Document, TEC492389. This Knowledge Document has details on the security commands necessary to properly configure Developer for System z.
- For those sites that use CA Top Secret® for z/OS, please refer to your product page on the CA support site (https://support.ca.com) and check for the related Developer for System z Knowledge Document, TEC492091. This Knowledge Document has details on the security commands necessary to properly configure Developer for System z.

The following sections describe the required steps, optional configuration and possible alternatives.

Requirements and checklist

To complete the security setup, the security administrator needs to know the values listed in Table 19. These values were defined during previous steps of the installation and customization of Developer for System z.

Table 19. Security setup variables

Description	• **Default value** • **Where to find the answer**	**Value**
Developer for System z product high level qualifier	• FEK • SMP/E installation	
Developer for System z customization high level qualifier	• FEK.#CUST • FEK.SFEKSAMP(FEKSETUP), as described in "Customization setup" on page 13.	
JES Job Monitor started task name	• JMON • FEK.#CUST.PROCLIB(JMON), as described in "PROCLIB changes" on page 20.	

Table 19. Security setup variables (continued)

Description	• **Default value** • **Where to find the answer**	Value
RSE daemon started task name	• `RSED` • `FEK.#CUST.PROCLIB(RSED)`, as described in "PROCLIB changes" on page 20.	
Lock daemon started task name	• `LOCKD` • `FEK.#CUST.PROCLIB(LOCKD)`, as described in "PROCLIB changes" on page 20.	
Application ID	• `FEKAPPL` • `/etc/rdz/rsed.envvars`, as described in "Defining extra Java startup parameters with _RSE_JAVAOPTS" on page 40	

The following list is an overview of the required actions to complete the basic security setup of Developer for System z. As documented in the sections below, different methods can be used to fulfill these requirements, depending on the desired security level. Refer to the previous sections for information about the security setup of optional Developer for System z services.

- "Activate security settings and classes"
- "Define an OMVS segment for Developer for System z users" on page 130
- "Define data set profiles" on page 130
- "Define the Developer for System z started tasks" on page 134
- "Define JES command security" on page 135
- "Define RSE as a secure z/OS UNIX server" on page 137
- "Define MVS program controlled libraries for RSE" on page 137
- "Define application protection for RSE" on page 138
- "Define PassTicket support for RSE" on page 139
- "Define z/OS UNIX program controlled files for RSE" on page 140
- "Verify security settings" on page 140

Activate security settings and classes

Developer for System z utilizes a variety of security mechanisms to ensure a secure and controlled host environment for the client. In order to do so, several classes and security settings must be active, as shown with the following sample RACF commands:

- Display current settings
  -
    ```
    SETROPTS LIST
    ```
- Activate facility class for z/OS UNIX and digital certificate profiles
  -
    ```
    SETROPTS GENERIC(FACILITY)
    ```

-
  ```
  SETROPTS CLASSACT(FACILITY) RACLIST(FACILITY)
  ```
- Activate started task definitions
  -
    ```
    SETROPTS GENERIC(STARTED)
    ```
  -
    ```
    RDEFINE STARTED ** STDATA(USER(=MEMBER) GROUP(STCGROUP) TRACE(YES))
    ```
  -
    ```
    SETROPTS CLASSACT(STARTED) RACLIST(STARTED)
    ```
- Activate console security for JES Job Monitor
  -
    ```
    SETROPTS GENERIC(CONSOLE)
    ```
  -
    ```
    SETROPTS CLASSACT(CONSOLE) RACLIST(CONSOLE)
    ```
- Activate operator command protection for JES Job Monitor
  -
    ```
    SETROPTS GENERIC(OPERCMDS)
    ```
  -
    ```
    SETROPTS CLASSACT(OPERCMDS) RACLIST(OPERCMDS)
    ```
- Activate application protection for RSE
  -
    ```
    SETROPTS GENERIC(APPL)
    ```
  -
    ```
    SETROPTS CLASSACT(APPL) RACLIST(APPL)
    ```
- Activate secured signon using PassTickets for RSE
  -
    ```
    SETROPTS GENERIC(PTKTDATA)
    ```
  -
    ```
    SETROPTS CLASSACT(PTKTDATA) RACLIST(PTKTDATA)
    ```
- Activate program control to ensure that only trusted code can be loaded by RSE
  -
    ```
    RDEFINE PROGRAM ** ADDMEM('SYS1.CMDLIB'//NOPADCHK) UACC(READ)
    ```
  -
    ```
    SETROPTS WHEN(PROGRAM)
    ```

    Note: Do not create the ** profile if you already have a * profile in the PROGRAM class. It obscures and complicates the search path used by your security software. In this case, you must merge the existing * and the new ** definitions. IBM recommends to use the ** profile, as documented in *Security Server RACF Security Administrator's Guide* (SA22-7683).

Attention: Some products, such as FTP, require being program controlled if "WHEN PROGRAM" is active. Test this before activating it on a production system.

- (Optional) Activate X.509 HostIdMappings and extended Port Of Entry (POE) support
  -
    ```
    SETROPTS GENERIC(SERVAUTH)
    ```

-

```
SETROPTS CLASSACT(SERVAUTH) RACLIST(SERVAUTH)
```

Define an OMVS segment for Developer for System z users

A RACF OMVS segment (or equivalent) that specifies a valid non-zero z/OS UNIX user ID (UID), home directory, and shell command must be defined for each user of Developer for System z. Their default group also requires an OMVS segment with a group id.

Replace in the following sample RACF commands the #userid, #user-identifier, #group-name and #group-identifier placeholders with actual values:

-
  ```
  ALTUSER #userid
  OMVS(UID(#user-identifier) HOME(/u/#userid) PROGRAM(/bin/sh) NOASSIZEMAX)
  ```

-
  ```
  ALTGROUP #group-name OMVS(GID(#group-identifier))
  ```

Although it is advised not to do so, you can use the shared OMVS segment defined in the BPX.DEFAULT.USER profile of the FACILITY class to fulfill the OMVS segment requirement for Developer for System z.

Define data set profiles

READ access for users and ALTER for system programmers suffices for most Developer for System z data sets. Replace the #sysprog placeholder with valid user IDs or RACF group names. Also ask the system programmer who installed and configured the product for the correct data set names. FEK is the default high-level qualifier used during installation and FEK.#CUST is the default high-level qualifier for data sets created during the customization process.

-
  ```
  ADDGROUP (FEK) OWNER(IBMUSER) SUPGROUP(SYS1)
  DATA('RATIONAL DEVELOPER FOR SYSTEM Z - HLQ STUB')
  ```

-
  ```
  ADDSD  'FEK.*.**' UACC(READ)
  DATA('RATIONAL DEVELOPER FOR SYSTEM Z')
  ```

-
  ```
  PERMIT 'FEK.*.**' CLASS(DATASET) ACCESS(ALTER) ID(#sysprog)
  ```

-
  ```
  SETROPTS GENERIC(DATASET) REFRESH
  ```

Note:

- You are strongly advised to protect FEK.SFEKAUTH against updates since this data set is APF authorized. The same is true for FEK.SFEKLOAD and FEK.SFEKLPA, but here because these data sets are program controlled.
- The sample commands in this publication and in the FEKRACF job assume that EGN (Enhanced Generic Naming) is active. This allows the usage of the ** qualifier to represent any number of qualifiers in the DATASET class. Substitute ** with * if EGN is not active on your system. Refer to *Security Server RACF Security Administrator's Guide* (SA22-7683) for more information about EGN.

Some of the optional Developer for System z components require additional security data set profiles. Replace the `#sysprog`, `#ram-developer` and `#cicsadmin` placeholders with valid user ID's or RACF group names:

- If SCLM Developer Toolkit's long/short name translation is used, then users require UPDATE access to the mapping VSAM, FEK.#CUST.LSTRANS.FILE.
  –
  ```
  ADDSD  'FEK.#CUST.LSTRANS.*.**' UACC(UPDATE)
   DATA('RATIONAL DEVELOPER FOR SYSTEM Z - SCLMDT')
  ```
  –
  ```
  PERMIT 'FEK.#CUST.LSTRANS.*.**' CLASS(DATASET) ACCESS(ALTER) ID(#sysprog)
  ```
  –
  ```
  SETROPTS GENERIC(DATASET) REFRESH
  ```
- CARMA RAM (Repository Access Manager) developers require UPDATE access to the CARMA VSAMs, FEK.#CUST.CRA*.
  –
  ```
  ADDSD  'FEK.#CUST.CRA*.**' UACC(READ)
  DATA('RATIONAL DEVELOPER FOR SYSTEM Z - CARMA')
  ```
  –
  ```
  PERMIT 'FEK.#CUST.CRA*.**' CLASS(DATASET) ACCESS(ALTER) ID(#sysprog)
  ```
  –
  ```
  PERMIT 'FEK.#CUST.CRA*.**' CLASS(DATASET) ACCESS(UPDATE) ID(#ram-developer)
  ```
  –
  ```
  SETROPTS GENERIC(DATASET) REFRESH
  ```
- If Application Deployment Manager's CRD server (CICS Resource Definition) is used, then CICS administrators require UPDATE access to the CRD repository VSAM.
  –
  ```
  ADDSD 'FEK.#CUST.ADNREP*.**' UACC(READ)
  DATA('RATIONAL DEVELOPER FOR SYSTEM Z - ADN')
  ```
  –
  ```
  PERMIT 'FEK.#CUST.ADNREP*.**' CLASS(DATASET) ACCESS(ALTER) ID(#sysprog)
  ```
  –
  ```
  PERMIT 'FEK.#CUST.ADNREP*.**' CLASS(DATASET) ACCESS(UPDATE) ID(#cicsadmin)
  ```
  –
  ```
  SETROPTS GENERIC(DATASET) REFRESH
  ```
- If Application Deployment Manager's manifest repository is defined, then all CICS Transaction Server users require UPDATE access to the manifest repository VSAM.
  –
  ```
  ADDSD 'FEK.#CUST.ADNMAN*.**' UACC(UPDATE)
  DATA('RATIONAL DEVELOPER FOR SYSTEM Z - ADN')
  ```
  –
  ```
  PERMIT 'FEK.#CUST.ADNMAN*.**' CLASS(DATASET) ACCESS(ALTER) ID(#sysprog)
  ```
  –
  ```
  SETROPTS GENERIC(DATASET) REFRESH
  ```

Use the following sample RACF commands for a more secure setup where READ access is also controlled.

- uacc(none) data set protection
  –

```
ADDGROUP (FEK)
DATA('RATIONAL DEVELOPER FOR SYSTEM Z - HLQ STUB')
OWNER(IBMUSER) SUPGROUP(SYS1)"
```

-
  ```
  ADDSD 'FEK.*.**'     UACC(NONE)
  DATA('RATIONAL DEVELOPER FOR SYSTEM Z')
  ```

-
  ```
  ADDSD 'FEK.SFEKAUTH' UACC(NONE)
  DATA('RATIONAL DEVELOPER FOR SYSTEM Z')
  ```

-
  ```
  ADDSD 'FEK.SFEKLOAD' UACC(NONE)
  DATA('RATIONAL DEVELOPER FOR SYSTEM Z')
  ```

-
  ```
  ADDSD 'FEK.SFEKPROC' UACC(NONE)
  DATA('RATIONAL DEVELOPER FOR SYSTEM Z')
  ```

-
  ```
  ADDSD 'FEK.#CUST.PARMLIB' UACC(NONE)
  DATA('RATIONAL DEVELOPER FOR SYSTEM Z')
  ```

-
  ```
  ADDSD 'FEK.#CUST.CNTL' UACC(NONE)
  DATA('RATIONAL DEVELOPER FOR SYSTEM Z')
  ```

-
  ```
  ADDSD 'FEK.#CUST.LSTRANS.*.**' UACC(NONE)
  DATA('RATIONAL DEVELOPER FOR SYSTEM Z - SCLMDT')
  ```

-
  ```
  ADDSD 'FEK.#CUST.CRA*.**' UACC(NONE)
  DATA('RATIONAL DEVELOPER FOR SYSTEM Z - CARMA')
  ```

-
  ```
  ADDSD 'FEK.#CUST.ADNREP*.**' UACC(NONE)
  DATA('RATIONAL DEVELOPER FOR SYSTEM Z - ADN')
  ```

-
  ```
  ADDSD 'FEK.#CUST.ADNMAN*.**' UACC(NONE)
  DATA('RATIONAL DEVELOPER FOR SYSTEM Z - ADN')
  ```

- permit system programmer to manage all libraries

  -
    ```
    PERMIT 'FEK.*.**     CLASS(DATASET) ACCESS(ALTER) ID(#sysprog)
    ```

  -
    ```
    PERMIT 'FEK.SFEKAUTH CLASS(DATASET) ACCESS(ALTER) ID(#sysprog)
    ```

  -
    ```
    PERMIT 'FEK.SFEKLOAD CLASS(DATASET) ACCESS(ALTER) ID(#sysprog)
    ```

  -
    ```
    PERMIT 'FEK.SFEKLOAD CLASS(DATASET) ACCESS(ALTER) ID(#sysprog)
    ```

  -
    ```
    PERMIT 'FEK.SFEKLOAD CLASS(DATASET) ACCESS(ALTER) ID(#sysprog)
    ```

  -
    ```
    PERMIT 'FEK.SFEKPROC CLASS(DATASET) ACCESS(ALTER) ID(#sysprog)
    ```

  -
    ```
    PERMIT 'FEK.#CUST.PARMLIB CLASS(DATASET) ACCESS(ALTER) ID(#sysprog)
    ```

  -
    ```
    PERMIT 'FEK.#CUST.CNTL CLASS(DATASET) ACCESS(ALTER) ID(#sysprog)
    ```

  -

    ```
    PERMIT 'FEK.#CUST.LSTRANS.*.**' CLASS(DATASET) ACCESS(ALTER) ID(#sysprog)
    ```
  -

    ```
    PERMIT 'FEK.#CUST.CRA*.**' CLASS(DATASET) ACCESS(ALTER) ID(#sysprog)
    ```
  -

    ```
    PERMIT 'FEK.#CUST.ADNREP*.**' CLASS(DATASET) ACCESS(ALTER) ID(#sysprog)
    ```
  -

    ```
    PERMIT 'FEK.#CUST.ADNMAN*.**' CLASS(DATASET) ACCESS(ALTER) ID(#sysprog)
    ```
- permit clients to access the load and exec libraries
  -

    ```
    PERMIT 'FEK.SFEKAUTH' CLASS(DATASET) ACCESS(READ) ID(*)
    ```
  -

    ```
    PERMIT 'FEK.SFEKLOAD' CLASS(DATASET) ACCESS(READ) ID(*)
    ```
  -

    ```
    PERMIT 'FEK.SFEKPROC' CLASS(DATASET) ACCESS(READ) ID(*)
    ```
  -

    ```
    PERMIT 'FEK.#CUST.CNTL' CLASS(DATASET) ACCESS(READ) ID(*)
    ```

  Note: No permits are needed for FEK.SFEKLPA, as all code that resides in LPA is accessible by everyone.
- permit JES Job Monitor to access the load and parameter library
  -

    ```
    PERMIT 'FEK.SFEKAUTH' CLASS(DATASET) ACCESS(READ) ID(STCJMON)
    ```
  -

    ```
    PERMIT 'FEK.#CUST.PARMLIB' CLASS(DATASET) ACCESS(READ) ID(STCJMON)
    ```
- (optional) permit clients to update the long/short name translation VSAM for SCLMDT
  -

    ```
    PERMIT 'FEK.#CUST.LSTRANS.*.**' CLASS(DATASET) ACCESS(UPDATE) ID(*)
    ```
- (optional) permit RAM developers to update the CARMA VSAMs for CARMA
  -

    ```
    PERMIT 'FEK.#CUST.CRA*.**' CLASS(DATASET) ACCESS(UPDATE) ID(#ram-developer)
    ```
- (optional) permit CICS users to read the CRD repository VSAM for Application Deployment Manager
  -

    ```
    PERMIT 'FEK.#CUST.ADNREP*.**' CLASS(DATASET) ACCESS(READ) ID(*)
    ```
- (optional) permit CICS administrators to update the CRD repository VSAM for Application Deployment Manager
  -

    ```
    PERMIT 'FEK.#CUST.ADNREP*.**' CLASS(DATASET) ACCESS(UPDATE) ID(#cicsadmin)
    ```
- (optional) permit CICS users to update the manifest repository VSAM for Application Deployment Manager
  -

    ```
    PERMIT 'FEK.#CUST.ADNMAN*.**' CLASS(DATASET) ACCESS(UPDATE) ID(*)
    ```
- (optional) permit CICS TS server to access the load library for bidi and Application Deployment Manager
  -

    ```
    PERMIT 'FEK.SFEKLOAD' CLASS(DATASET) ACCESS(READ) ID(#cicsts)
    ```

- (optional) permit DB2 server to access the exec library for DB2 stored procedure builder
  –
  ```
  PERMIT 'FEK.SFEKPROC' CLASS(DATASET) ACCESS(READ) ID(#db2)
  ```
- activate security profiles
  –
  ```
  SETROPTS GENERIC(DATASET) REFRESH
  ```

When controlling READ access to system data sets, you must provide Developer for System z servers and users permission to READ the following data sets:

- CEE.SCEERUN
- CEE.SCEERUN2
- CBC.SCLBDLL
- ISP.SISPLOAD
- ISP.SISPLPA
- SYS1.LINKLIB
- SYS1.SIEALNKE
- REXX.V1R4M0.SEAGLPA

Note: When you use the Alternate Library for REXX product package, the default REXX runtime library name is REXX.*.SEAGALT. instead of REXX.*.SEAGLPA, as used in the sample above.

Define the Developer for System z started tasks

The following sample RACF commands create the JMON, RSED, and LOCKD started tasks, with protected user IDs (STCJMON, STCRSE, and STCLOCK respectively) and group STCGROUP assigned to them. Replace the #group-id and #user-id-* placeholders with valid OMVS IDs.

-
  ```
  ADDGROUP STCGROUP OMVS(GID(#group-id))
  DATA('GROUP WITH OMVS SEGMENT FOR STARTED TASKS')
  ```

-
  ```
  ADDUSER STCJMON DFLTGROUP(STCGROUP) NOPASSWORD NAME('RDZ - JES JOBMONITOR')
  OMVS(UID(#user-id-jmon) HOME(/tmp) PROGRAM(/bin/sh) NOASSIZEMAX
  NOTHREADSMAX)
  DATA('RATIONAL DEVELOPER FOR SYSTEM Z')
  ```

-
  ```
  ADDUSER STCRSE DFLTGROUP(STCGROUP) NOPASSWORD NAME('RDZ - RSE DAEMON')
  OMVS(UID(#user-id-rse) HOME(/tmp) PROGRAM(/bin/sh) ASSIZEMAX(2147483647)
  NOTHREADSMAX)
  DATA('RATIONAL DEVELOPER FOR SYSTEM Z')
  ```

-
  ```
  ADDUSER STCLOCK DFLTGROUP(STCGROUP) NOPASSWORD NAME('RDZ - LOCK DAEMON')
  OMVS(UID(#user-id-lock) HOME(/tmp) PROGRAM(/bin/sh) NOASSIZEMAX)
  NOTHREADSMAX)
  DATA('RATIONAL DEVELOPER FOR SYSTEM Z')
  ```

-
  ```
  RDEFINE STARTED JMON.* DATA('RDZ - JES JOBMONITOR')
  STDATA(USER(STCJMON) GROUP(STCGROUP) TRUSTED(NO))
  ```
-

```
RDEFINE STARTED RSED.* DATA('RDZ - RSE DAEMON')
STDATA(USER(STCRSE) GROUP(STCGROUP) TRUSTED(NO))
```

-
  ```
  RDEFINE STARTED LOCKD.* DATA('RDZ - LOCK DAEMON')
  STDATA(USER(STCLOCK) GROUP(STCGROUP) TRUSTED(NO))
  ```
-
  ```
  SETROPTS RACLIST(STARTED) REFRESH
  ```

Note:

- Ensure that the started tasks user IDs are protected by specifying the NOPASSWORD keyword.
- Ensure that RSE server has a unique OMVS uid due to the z/OS UNIX related privileges granted to this uid.
- RSE daemon requires a large address space size (2GB) for proper operation. You should set this value in the ASSIZEMAX variable of the OMVS segment for user ID STCRSE. This to ensure that RSE daemon will get the required region size, regardless of changes to MAXASSIZE in SYS1.PARMLIB(BPXPRMxx).
- RSE also requires a large number of threads for proper operation. You can set the limit in the THREADSMAX variable of the OMVS segment for user ID STCRSE. This ensures that RSE will get the required thread limit, regardless of changes to MAXTHREADS or MAXTHREADTASKS in SYS1.PARMLIB(BPXPRMxx). Refer to "Tuning considerations" in the *Host Configuration Reference* (SC14-7290) to determine the correct value for the thread limit.
- User ID STCJMON is another good candidate for setting THREADSMAX in the OMVS segment, because JES Job Monitor uses a thread per client connection.

You might want to consider making the STCRSE user ID restricted. Users with the RESTRICTED attribute cannot access protected (MVS) resources they are not specifically authorized to access.

```
ALTUSER STCRSE RESTRICTED
```

To ensure that restricted users do not gain access to z/OS UNIX file system resources through the “other” permission bits, you must define the RESTRICTED.FILESYS.ACCESS profile in the UNIXPRIV class with UACC(NONE). Refer to *Security Server RACF Security Administrator's Guide* (SA22-7683) for more information about restricting user IDs.

Attention: If you use restricted user IDs, you must explicitly add the permission to access a resource with the TSO **PERMIT** or the z/OS UNIX **setfacl** commands. This includes resources where the Developer for System z documentation uses UACC (such as the ** profile in the PROGRAM class) or where it relies on common z/OS UNIX conventions (such as everyone having read and execute permission for Java libraries). Test this before activating it on a production system.

Define JES command security

JES Job Monitor issues all JES operator commands requested by a user through an extended MCS (EMCS) console, whose name is controlled with the CONSOLE_NAME directive, as documented in “FEJJCNFG, JES Job Monitor configuration file” on page 25.

The following sample RACF commands give Developer for System z users conditional access to a limited set of JES commands (Hold, Release, Cancel, and

Purge). Users only have execution permission if they issue the commands through JES Job monitor. Replace the #console placeholder with the actual console name.

-
  ```
  RDEFINE OPERCMDS MVS.MCSOPER.#console UACC(READ)
  DATA('RATIONAL DEVELOPER FOR SYSTEM Z')
  ```

-
  ```
  RDEFINE OPERCMDS JES%.** UACC(NONE)
  ```

-
  ```
  PERMIT JES%.** CLASS(OPERCMDS) ACCESS(UPDATE) WHEN(CONSOLE(JMON)) ID(*)
  ```

-
  ```
  SETROPTS RACLIST(OPERCMDS) REFRESH
  ```

Note:

- Usage of the console is permitted if no MVS.MCSOPER.#console profile is defined
- The CONSOLE class must be active for WHEN(CONSOLE(JMON)) to work, but there is no actual profile check in the CONSOLE class for EMCS consoles.
- Do not replace JMON with the actual console name in the WHEN(CONSOLE(JMON)) clause. The JMON keyword represents the point-of-entry application, not the console name.

Attention: Defining JES commands with universal access NONE in your security software might impact other applications and operations. Test this before activating it on a production system.

Table 20 and Table 21 show the operator commands issued for JES2 and JES3, and the discrete security profiles that can be used to protect them.

Table 20. JES2 Job Monitor operator commands

Action	Command	OPERCMDS profile	Required access
Hold	$Hx(jobid) with x = {J, S or T}	jesname.MODIFYHOLD.BAT jesname.MODIFYHOLD.STC jesname.MODIFYHOLD.TSU	UPDATE
Release	$Ax(jobid) with x = {J, S or T}	jesname.MODIFYRELEASE.BAT jesname.MODIFYRELEASE.STC jesname.MODIFYRELEASE.TSU	UPDATE
Cancel	$Cx(jobid) with x = {J, S or T}	jesname.CANCEL.BAT jesname.CANCEL.STC jesname.CANCEL.TSU	UPDATE
Purge	$Cx(jobid),P with x = {J, S or T}	jesname.CANCEL.BAT jesname.CANCEL.STC jesname.CANCEL.TSU	UPDATE

Table 21. JES3 Job Monitor operator commands

Action	Command	OPERCMDS profile	Required access
Hold	*F,J=jobid,H	jesname.MODIFY.JOB	UPDATE
Release	*F,J=jobid,R	jesname.MODIFY.JOB	UPDATE
Cancel	*F,J=jobid,C	jesname.MODIFY.JOB	UPDATE

Table 21. JES3 Job Monitor operator commands (continued)

Action	Command	OPERCMDS profile	Required access
Purge	`*F,J=jobid,C`	`jesname.MODIFY.JOB`	`UPDATE`

Note:

- The Hold, Release, Cancel, and Purge JES operator commands, and the Show JCL command, can only be executed against spool files owned by the client user ID, unless `LIMIT_COMMANDS=` with value `LIMITED` or `NOLIMIT` is specified in the JES Job Monitor configuration file. Refer to "Actions against jobs - target limitations" in the *Host Configuration Reference* (SC14-7290) for more information about this.
- Users can browse any spool file, unless `LIMIT_VIEW=USERID` is defined in the JES Job Monitor configuration file. Refer to "Access to spool files" in *Host Configuration Reference* (SC14-7290) for more information about this.
- Without being authorized for these operator commands, users will still be able to submit jobs and read job output through JES Job Monitor, if they have sufficient authority to possible profiles that protect these resources (such as those in the `JESINPUT`, `JESJOBS` and `JESSPOOL` classes).

Assuming the identity of the JES Job Monitor server by creating a JMON console from a TSO session is prevented by your security software. Even though the console can be created, the point of entry is different (JES Job Monitor versus TSO). JES commands issued from this console will fail the security check, if your security is set up as documented in this publication and the user does not have authority to the JES commands through other means.

Define RSE as a secure z/OS UNIX server

RSE requires UPDATE access to the `BPX.SERVER` profile to create/delete the security environment for the client's thread. If this profile is not defined, `UID(0)` is required for RSE.

- `RDEFINE FACILITY BPX.SERVER UACC(NONE)`
- `PERMIT BPX.SERVER CLASS(FACILITY) ACCESS(UPDATE) ID(STCRSE)`
- `SETROPTS RACLIST(FACILITY) REFRESH`

> **Attention:** Defining the `BPX.SERVER` profile makes z/OS UNIX as a whole switch from UNIX level security to z/OS UNIX level security, which is more secure. This might impact other z/OS UNIX applications and operations. Test this before activating it on a production system. Refer to *UNIX System Services Planning* (GA22-7800) for more information about the different security levels.

Define MVS program controlled libraries for RSE

Servers with authority to `BPX.SERVER` must run in a clean, program-controlled environment. This implies that all programs called by RSE must also be program controlled. For MVS load libraries, program control is managed by your security software.

RSE uses system (`SYS1.LINKLIB`), Language Environment's runtime (`CEE.SCEERUN*`) and ISPF's TSO/ISPF Client Gateway (`ISP.SISPLOAD`) load library.

- `RALTER PROGRAM ** UACC(READ) ADDMEM('SYS1.LINKLIB'//NOPADCHK)`
- `RALTER PROGRAM ** UACC(READ) ADDMEM('CEE.SCEERUN'//NOPADCHK)`

- `RALTER PROGRAM ** UACC(READ) ADDMEM('CEE.SCEERUN2'//NOPADCHK)`
- `RALTER PROGRAM ** UACC(READ) ADDMEM('ISP.SISPLOAD'//NOPADCHK)`
- `SETROPTS WHEN(PROGRAM) REFRESH`

Note: Do not use the `**` profile if you already have a `*` profile in the `PROGRAM` class. It obscures and complicates the search path used by your security software. In this case, you must merge the existing `*` and the new `**` definitions. IBM recommends using the ** profile, as documented in *Security Server RACF Security Administrator's Guide* (SA22-7683).

The following additional (prerequisite) libraries must be made program controlled to support the use of optional services. This list does not include data sets that are specific to a product that Developer for System z interacts with, such as IBM Debug Tool.

- Alternate REXX runtime library (for SCLM Developer Toolkit)
  - `REXX.*.SEAGALT`
- System load library (for SSL encryption)
  - `SYS1.SIEALNKE`
- File Manager listener load library (for File Manager integration)
  - `FMN.SFMNMODA`

Note: Libraries that are designed for LPA placement also require program control authorizations if they are accessed through LINKLIST or STEPLIB. This publication documents the usage of the following LPA libraries:

- ISPF (for TSO/ISPF Client Gateway)
  - `ISP.SISPLPA`
- REXX runtime library (for SCLM Developer Toolkit)
  - `REXX.*.SEAGLPA`
- Developer for System z (for CARMA)
  - `FEK.SFEKLPA`

Define application protection for RSE

During client logon, RSE daemon verifies that a user is allowed to use the application.

-
  ```
  RDEFINE APPL FEKAPPL UACC(READ) DATA('RATIONAL DEVELOPER FOR SYSTEM Z')
  ```
-
  ```
  SETROPTS RACLIST(APPL) REFRESH
  ```

Note:

- As described in more detail in "Define PassTicket support for RSE" on page 139, RSE supports the usage of an application ID other than `FEKAPPL`. The `APPL` class definition must match the actual application ID used by RSE.
- The client connection request will succeed if the application ID is not defined in the `APPL` class.
- The client connection request will only fail if the application ID is defined and the user lacks READ access to the profile.

Define PassTicket support for RSE

The client's password (or other means of identification, such as an X.509 certificate) is only used to verify his identity upon connection. Afterwards, PassTickets are used to maintain thread security.

PassTickets are system-generated passwords with a lifespan of about 10 minutes. The generated PassTickets are based upon a secret key. This key is a 64 bit number (16 hex characters). Replace in the sample RACF commands below the key16 placeholder with a user-supplied 16 character hex string (characters 0-9 and A-F).

-
```
RDEFINE PTKTDATA FEKAPPL UACC(NONE) SSIGNON(KEYMASKED(key16))
APPLDATA('NO REPLAY PROTECTION – DO NOT CHANGE')
DATA('RATIONAL DEVELOPER FOR SYSTEM Z')
```
-
```
RDEFINE PTKTDATA IRRPTAUTH.FEKAPPL.* UACC(NONE)
DATA('RATIONAL DEVELOPER FOR SYSTEM Z')
```
-
```
PERMIT IRRPTAUTH.FEKAPPL.* CLASS(PTKTDATA) ACCESS(UPDATE) ID(STCRSE)
```
-
```
SETROPTS RACLIST(PTKTDATA) REFRESH
```

RSE supports the usage of an application ID other than FEKAPPL. Uncomment and customize the "APPLID=FEKAPPL" option in rsed.envvars to activate this, as documented in “Defining extra Java startup parameters with _RSE_JAVAOPTS” on page 40. The PTKTDATA class definitions must match the actual application ID used by RSE.

You should not use OMVSAPPL as application ID, because it will open the secret key to most z/OS UNIX applications. You should also not use the default MVS application ID, which is MVS followed by the system's SMF ID, because this will open the secret key to most MVS applications (including user batch jobs).

Note:

- If the PTKTDATA class is already defined, verify that it is defined as a generic class before creating the profiles listed above. The support for generic characters in the PTKTDATA class is new since z/OS release 1.7, with the introduction of a Java interface to PassTickets.
- Substitute the wildcard (*) in the IRRPTAUTH.FEKAPPL.* definition with a valid user ID mask to limit the user IDs for which RSE can generate a PassTicket.
- Depending on your RACF settings, the user defining a profile might also be on the access list of the profile. It is advised that you remove this permission for the PTKTDATA profiles.
- JES Job Monitor and RSE must have the same application ID to allow JES Job Monitor to evaluate the PassTickets presented by RSE.
- If the system has a cryptographic product installed and available, you can encrypt the secured signon application key for added protection. In order to do so, use the KEYENCRYPTED keyword instead of KEYMASKED. Refer to *Security Server RACF Security Administrator's Guide* (SA22-7683) for more information about this.

Attention: The client connection request will fail if PassTickets are not set up correctly.

Define z/OS UNIX program controlled files for RSE

Servers with authority to BPX.SERVER must run in a clean, program-controlled environment. This implies that all programs called by RSE must also be program controlled. For z/OS UNIX files, program control is managed by the **extattr** command. To execute this command, you need READ access to BPX.FILEATTR.PROGCTL in the FACILITY class, or be UID(0).

RSE server uses RACF's Java shared library (/usr/lib/libIRRRacf*.so).

- extattr +p /usr/lib/libIRRRacf*.so

Note:

- Since z/OS 1.9, /usr/lib/libIRRRacf*.so is installed program controlled during SMP/E RACF install.
- Since z/OS 1.10, /usr/lib/libIRRRacf*.so is part of SAF, which ships with base z/OS, so it is available also to non-RACF customers.
- The setup might be different if you use a security product other than RACF. Consult the documentation of your security product for more information.
- The SMP/E install of Developer for System z sets the program control bit for internal RSE programs.
- Use the **ls -Eog** z/OS UNIX command to display the current status of the program control bit (the file is program controlled if the letter **p** shows in the second string).

```
$ ls -Eog /usr/lib/libIRRRacf*.so
-rwxr-xr-x  aps-  2     69632 Oct  5  2007 /usr/lib/libIRRRacf.so
-rwxr-xr-x  aps-  2     69632 Oct  5  2007 /usr/lib/libIRRRacf64.so
```

Verify security settings

Use the following sample commands to display the results of your security-related customizations.

- Security settings and classes
  - SETROPTS LIST
- OMVS segment for users
  - LISTUSER #userid NORACF OMVS
  - LISTGRP #group-name NORACF OMVS
- Data set profiles
  - LISTGRP FEK
  - LISTDSD PREFIX(FEK) ALL
- Started tasks
  - LISTGRP STCGROUP OMVS
  - LISTUSER STCJMON OMVS
  - LISTUSER STCRSE   OMVS
  - LISTUSER STCLOCK OMVS
  - RLIST STARTED JMON.* ALL STDATA
  - RLIST STARTED RSED.* ALL STDATA
  - RLIST STARTED LOCKD.* ALL STDATA
- JES command security
  - RLIST CONSOLE JMON ALL
  - RLIST OPERCMDS MVS.MCSOPER.JMON ALL

  - `RLIST OPERCMDS JES%.** ALL`
- RSE as a secure z/OS UNIX server
  - `RLIST FACILITY BPX.SERVER ALL`
- MVS program controlled libraries for RSE
  - `RLIST PROGRAM ** ALL`
- Application protection for RSE
  - `RLIST APPL FEKAPPL ALL`
- PassTicket support for RSE
  - `RLIST PTKTDATA FEKAPPL ALL SSIGNON`
  - `RLIST PTKTDATA IRRPTAUTH.FEKAPPL.* ALL`
- z/OS UNIX program controlled files for RSE
  - `ls -E /usr/lib/libIRRRacf*.so`

Chapter 9. Migration guide

Migration considerations

This section highlights installation and configuration changes compared to previous releases of the product. It also gives some general guidelines to migrate to this release. Refer to the related sections in this manual for more information.

- If you are a previous user of IBM Rational Developer for System z, IBM WebSphere Developer for System z, IBM WebSphere Developer for zSeries or IBM WebSphere® Studio Enterprise Developer, it is recommended that you save the related customized files BEFORE installing the upgrade to this version of IBM Rational Developer for System z Version 8.5.
- Read "Running multiple instances" in the *Host Configuration Reference* (SC14-7290) if you plan on running multiple instances of Developer for System z.

Note: The migration information listed here is for Developer for System z versions that are still supported at the time of publication.

Backing up previously configured files

If you are a previous user of Developer for System z, it is recommended that you save the related customized files before installing this version of IBM Developer for System z.

Customizable Developer for system z files can be found at the following locations:

- Version 7.6 and 8.0.1
  - `FEK.#CUST.RDZ*.**`, Configuration Utility work files
  - `FEK.SFEKSAMP`, some members are copied to `FEK.#CUST.*` by the `FEKSETUP` sample job, where * equals `PARMLIB`, `PROCLIB`, `JCL`, `CNTL`, `ASM` and `COBOL`
  - `FEK.SFEKSAMV`
  - `/usr/lpp/rdz/samples/`, some files are copied to `/etc/rdz/` and `/var/rdz/sclmdt/*` by the `FEKSETUP` sample job, where * equals `CONFIG/`, `CONFIG/PROJECT/` and `CONFIG/script/`
- Version 7.5
  - `FEK.SFEKSAMP`, some members are copied to `FEK.#CUST.*` by the `FEKSETUP` sample job, where * equals `PARMLIB, PROCLIB, JCL, CNTL, ASM` and `COBOL`
  - `FEK.SFEKSAMV`
  - `/usr/lpp/rdz/samples/`, some files are copied to `/etc/rdz/` and `/etc/rdz/sclmdt/*` by the `FEKSETUP` sample job, where * equals `CONFIG/,` `CONFIG/PROJECT/` and `CONFIG/script/`
- Version 7.1
  - `FEK.SFEKSAMP`
  - `CRA.SCRASAMP`
  - `/usr/lpp/wd4z/rse/lib/`, customizable files are advised to be copied to `/etc/wd4z/`

Previous Developer for system z setups also document changes to configuration files owned by other products.

- Version 8.0.1

  - SYS1.PARMLIB(BPXPRMxx)
    Set z/OS UNIX system defaults.
  - SYS1.PARMLIB(COMMNDxx)
    Start servers at IPL time.
  - SYS1.PARMLIB(LPALSTxx)
    Add FEK.SFEKLPA to LPA.
  - SYS1.PARMLIB(PROGxx)
    Add FEK.SFEKAUTH and FEK.SFEKLOAD to LINKLIST.
  - (WLM)
    Assign an application environment for a DB2 stored procedure.
- Version 7.5 and 7.6
  - SYS1.PARMLIB(ASCHPMxx)
    Define an APPC transaction class for the TSO Commands service.
  - SYS1.PARMLIB(BPXPRMxx)
    Set z/OS UNIX system defaults.
  - SYS1.PARMLIB(COMMNDxx)
    Start servers at IPL time.
  - SYS1.PARMLIB(LPALSTxx)
    Add FEK.SFEKLPA to LPA.
  - SYS1.PARMLIB(PROGxx)
    APF authorize FEK.SFEKAUTH.
    Add FEK.SFEKAUTH and FEK.SFEKLOAD to LINKLIST.
  - (APPC)
    Define an APPC transaction for the TSO Commands service.
  - (WLM)
    Associate an APPC transaction program with a TSO performance group.
  - (WLM)
    Assign an application environment for a DB2 stored procedure.
- Version 7.1
  - SYS1.PARMLIB(ASCHPMxx)
    Define an APPC transaction class for the TSO Commands service.
  - SYS1.PARMLIB(BPXPRMxx)
    Set z/OS UNIX system defaults.
  - SYS1.PARMLIB(PROGxx)
    APF authorize FEK.SFEKLOAD.
  - /etc/services
    Define the RSE daemon port.
  - /etc/inetd.conf
    Define the RSE daemon service.
  - /etc/SCLMDT/COMFIG/ISPF.conf
    Define the location of the TSO Commands server.
  - (APPC)
    Define an APPC transaction for the TSO Commands service.
  - (WLM)
    Associate an APPC transaction program with a TSO performance group.

- (WLM)

  Assign an application environment for a DB2 stored procedure.

Migrate from version 8.0.1 to version 8.5

These notes are for a migration from a base version 8.0.1 to version 8.5. It includes changes that are already documented as part of version 8.0.1 maintenance. The changes that are part of the maintenance stream (and thus possibly already implemented) are marked with the release where they were introduced.

IBM Rational Developer for System z, FMID HHOP850

- The default SMP/E install location for MVS and z/OS UNIX components did not change and thus remain `FEK.*` and `/usr/lpp/rdz/*`.
- CARMA - The CRASTART load module, which resides in LPA, has been updated, requiring an LPA update (since version 8.0.3.2).
- CARMA - The CRAMSG VSAM must be updated (since version 8.0.3 and 8.5).
- CARMA - The CRADEF and CRASTRS VSAM filess for the CA Endevor® SCM RAM must be updated to use the new support for CA Endevor® SCM batch-actions (since version 8.0.3) and CA Endevor® SCM packages (since version 8.0.3).
- CARMA - New CRADEF and CRASTRS VSAM input has been added to allow restoring CA Endevor® SCM package actions from CA Endevor® SCM element menus.
  - CRA0VPKD - to be merged in CRADEF.
  - CRA0VPKS - to be merged in CRASTRS.
- CARMA - New sample members have been added (since version 8.0.3):
  - CRABCFG - configuration file for CA Endevor® SCM batch-actions.
  - CRABATCA - sample job for CA Endevor® SCM batch-actions.
- CARMA - The following customizable members have changed (since version 8.0.3, 8.0.3.1, and 8.5):
  - CRANDVRA
  - CRASHOW
  - CRASRV.properties
  - CRABCFG
- CARMA - Additional DD statements are added for the CA Endevor® SCM RAM (since version 8.0.3):
  - CRABCFG
  - CRABSKEL
  - PKGSCLS (allocated by CRANDVRA)
- Enterprise Service Tools - IRZ load modules and message modules moved to a new library (since version 8.5):
  - FEK.SFEKLMOD(IRZ* IIRZ*)
- File Manager Integration is removed (since version 8.5). Some functions, such as unformatted QSAM editing, are now part of regular data set handling by Developer for System z. More advanced functions, such as formatted data editing using copybooks or include files, require the IBM File Manager plug-in for Eclipse.
- Include pre-processor - New sample members have been added (since version 8.0.3.1):

  - FEKRNPLI
- Host Configuration Utility - Added migration option (since version 8.0.2)
- JES Job Monitor - New operator commands have been added to the JMON started task (since version 8.0.3.2):
  - MODIFY STORAGE
- JES Job Monitor - New optional directives have been added to FEJJCNFG (since version 8.0.3.1 and 8.0.3.2):
  - LIMIT_CONSOLE
  - SEARCHALL
  - TRACE_STORAGE
- PROCLIB - The following PROCLIB members have changed (since version 8.0.3):
  - ELAXFUOP
- RSE - The option to specify TMPDIR as startup argument for the RSED and LOCKD started tasks has been removed. It is replaced by a non-customizable function that defines the home directory of the started task user ID to TMPDIR if `/tmp` is not available for write actions (since version 8.0.3.1).
- RSE - New operator commands have been added to the LOCKD started task (since version 8.0.2):
  - MODIFY DISPLAY TABLE
- RSE - New operator commands have been added to the RSED started task (since version 8.0.2, 8.0.3, and 8.0.3.2):
  - MODIFY IVP ISPF,userid
  - MODIFY IVP PASSTICKET,userid
  - MODIFY DEBUG HEAPDUMP,PID=pid
  - MODIFY DEBUG JAVACORE,PID=pid
- RSE - RSED started task operator commands have been enhanced (since version 8.0.2 and 8.0.3.1):
  - MODIFY DISPLAY CLIENT[{,LOGON | ,ID | ,USER}]
  - MODIFY DISPLAY PROCESS,CPU[,PID=pid]
- RSE - The following console messages are new (since version 8.0.3 and 8.0.3.1):
  - FEK910I = {0} IVP Exit code = {1}
  - FEK211W User, {0}, not logged on
- RSE - New non-customizable directives have been added to rsed.envvars (since version 8.0.3):
  - (_RSE_JAVAOPTS) -Dldap.server.address
  - (_RSE_JAVAOPTS) -Dldap.server.port
  - (_RSE_JAVAOPTS) -Dldap.ptc.group.name.suffix
  - _RSE_PTC
- RSE - New optional directives have been added to rsed.envvars (since version 8.0.3, 8.0.3.1, and 8.5):
  - (_RSE_JAVAOPTS) -Daudit.action
  - (_RSE_JAVAOPTS) -Daudit.action.id
  - (_RSE_JAVAOPTS) -Dlogon.action
  - (_RSE_JAVAOPTS) -Dlogon.action.id
  - (_RSE_JAVAOPTS) -Dreject.logon.threshold
  - (_RSE_JAVAOPTS) -Dinclude.c
  - (_RSE_JAVAOPTS) -Dinclude.cpp

  - (_RSE_JAVAOPTS) -DCPP_CLEANUP_INTERVAL
  - (_RSE_JAVAOPTS) -DRIS_BUFFER
  - (_RSE_JAVAOPTS) -DDSTORE_TCP_NO_DELAY
  - _RSE_FEK_SAF_CLASS
  - _RSE_LDAP_SERVER
  - _RSE_LDAP_PORT
  - _RSE_LDAP_PTC_GROUP_SUFFIX
  - CGI_ISPPREF
- RSE - Existing required directives have been renamed (since version 8.5):
  - _CMDSERV_BASE_HOME -> CGI_ISPHOME
  - _CMDSERV_CONF_HOME -> CGI_ISPCONF
  - _CMDSERV_WORK_HOME -> CGI_ISPWORK
  - _RSE_CMDSERV_OPTS -> _RSE_ISPF_OPTS
- RSE - Existing optional directives have been extended with more values (since version 8.5):
  - STEPLIB
- RSE - Interpretation of the following optional directives in rsed.envvars has changed (since version 8.0.3):
  - (_RSE_JAVAOPTS) -Dprocess.cleanup.interval
- RSE - The following optional configuration files are new (since version 8.5):
  - include.conf
- RSE - New optional directives have been added to pushtoclient.properties (since version 8.0.3):
  - accept.product.license
- RSE - Interpretation of the following optional directives in pushtoclient.properties has changed (since version 8.0.3):
  - config.enabled
  - product.enabled
  - reject.config.updates
  - reject.product.updates
- RSE - New z/OS UNIX samples have been added (since version 8.0.3 and 8.0.3.1):
  - process_audit.rex
  - process_logon.sh
- zUnit - New optional PROCLIB member has been added (since version 8.5):
  - AZUZUNIT
- New publication, *Rational Developer for System z Messages and Codes* (SC14-7497).
- New publication, *Rational Developer for System z Answers to common host configuration and maintenance issues* (SC14-7373).

Configurable files

Table 22 on page 148 shows an overview of files that are customized in version 8.5. Note that the Developer for System z sample libraries, `FEK.SFEKSAMP`, `FEK.SFEKSAMV` and `/usr/lpp/rdz/samples/`, come with more customizable members than listed here, such as sample CARMA source code and jobs to compile them.

The following members and files are no longer customizable or no longer used.

- FMIEXT.properties (no longer used)

Note: Sample job FEKSETUP copies all listed members to different data sets and directories, default FEK.#CUST.* and /etc/rdz/*.

Table 22. Version 8.5 customizations

Member/File	Default location	Purpose	Migration notes
FEKSETUP	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to create data sets and directories, and populate them with customizable files	Updated to include new customizable members, create new directory structure, and remove actions for files that are no longer used
JMON	FEK.SFEKSAMP(FEJJJCL) [FEK.#CUST.PROCLIB]	JCL for JES Job Monitor	None
FEJJJCL	FEK.SFEKSAMP [FEK.#CUST.PROCLIB(JMON)]	Shipping name for JMON member	See JMON member
RSED	FEK.SFEKSAMP(FEKRSED) [FEK.#CUST.PROCLIB]	JCL for RSE daemon	Changed TMPDIR support
FEKRSED	FEK.SFEKSAMP [FEK.#CUST.PROCLIB(RSED)]	Shipping name for RSED member	See RSED member
LOCKD	FEK.SFEKSAMP(FEKLOCKD) [FEK.#CUST.PROCLIB]	JCL for lock daemon	Changed TMPDIR support
FEKLOCKD	FEK.SFEKSAMP [FEK.#CUST.PROCLIB(LOCKD)]	Shipping name for LOCKD member	See LOCKD member
ELAXF*	FEK.SFEKSAMP [FEK.#CUST.PROCLIB]	JCL for remote project builds, and so on	Member ELAXFUOP has changed
FEKRACF	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL for security definitions	None
FEJJCNFG	FEK.SFEKSAMP [FEK.#CUST.PARMLIB]	JES Job Monitor configuration file	New optional directives have been added
FEJTSO	FEK.SFEKSAMP [FEK.#CUST.CNTL]	JCL for TSO submits	None
CRA$VMSG	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to create the CARMA message VSAM	VSAM input has changed
CRA$VDEF	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to create the CARMA configuration VSAM	None
CRA$VSTR	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to create the CARMA custom information VSAM	None

Table 22. Version 8.5 customizations (continued)

Member/File	Default location	Purpose	Migration notes
CRA$VCAD	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to create the CARMA configuration VSAM for CA Endevor® SCM RAM	VSAM input has changed
CRA$VCAS	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to create the CARMA custom information VSAM for CA Endevor® SCM RAM	VSAM input has changed
CRASUBMT	FEK.SFEKSAMP [FEK.#CUST.CNTL]	CARMA batch startup CLIST	None
CRASUBCA	FEK.SFEKSAMP [FEK.#CUST.CNTL]	CARMA batch startup CLIST for CA Endevor® SCM RAM	Additional DD statements added
CRABCFG	FEK.SFEKSAMP [FEK.#CUST.PARMLIB]	CARMA batch actions configuration for CA Endevor® SCM RAM	NEW, customization is optional
CRABATCA	FEK.SFEKSAMP [FEK.#CUST.CNTL]	CARMA batch action JCL for CA Endevor® SCM RAM	NEW, customization is optional
CRASHOW	FEK.SFEKSAMP [FEK.#CUST.PARMLIB]	CARMA configuration for CA Endevor® SCM RAM	New filters are added
CRATMAP	FEK.SFEKSAMP [FEK.#CUST.PARMLIB]	CARMA configuration for CA Endevor® SCM RAM	None
CRANDVRA	FEK.SFEKPROC	CARMA allocation REXX for CA Endevor® SCM RAM	Additional DD statements added
CRA#VSLM	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to create the SCLM RAM's message VSAM	None
CRA#ASLM	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to create the SCLM RAM's data sets	None
CRA#VPDS	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to create the PDS RAM's message VSAM	None
CRA#UADD	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to merge RAM definitions	None
CRA#UQRY	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to extract RAM definitions	None
CRAXJCL	FEK.SFEKSAMP [FEK.#CUST.ASM]	Sample source code for IRXJCL replacement	None
CRA#CIRX	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to compile CRAXJCL	None
ADNCSDRS	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to define the RESTful CRD server to primary CICS region	None

Table 22. Version 8.5 customizations (continued)

Member/File	Default location	Purpose	Migration notes
ADNCSDTX	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to define alternate transaction IDs to CICS region	None
ADNTXNC	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to create alternate transaction IDs	None
ADNMSGHC	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to compile ADNMSGHS	None
ADNMSGHS	FEK.SFEKSAMP [FEK.#CUST.COBOL]	Sample source code for the Pipeline Message Handler	None
ADNVCRD	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to create the CRD repository	None
ADNCSDWS	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to define the Web Service CRD server to primary CICS region	None
ADNCSDAR	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to define the CRD server to non-primary CICS regions	None
ADNJSPAU	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to update the CRD defaults	None
ADNVMFST	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to create and define the Manifest repository	None
ELAXMSAM	FEK.SFEKSAMP [FEK.#CUST.PROCLIB]	JCL procedure of the WLM address space	None
ELAXMJCL	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to define the PL/I and COBOL Stored Procedure Builder to DB2	None
AZUZUNIT	FEK.SFEKSAMP [FEK.#CUST.PROCLIB]	JCL procedure for zUnit	NEW, customization is optional
FEKRNPLI	FEK.SFEKSAMP [FEK.#CUST.CNTL]	Rexx to invoke the PL/I compiler from within the preprocessor framework	NEW, customization is optional
FEKLOGS	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to collect log files	Added additional checks (customization must be redone)
rsed.envvars	/usr/lpp/rdz/samples/ [/etc/rdz/]	RSE environment variables	Older copies must be replaced by this one (customizations must be redone).
ISPF.conf	/usr/lpp/rdz/samples/ [/etc/rdz/]	TSO/ISPF Client Gateway configuration file	None

Table 22. Version 8.5 customizations (continued)

Member/File	Default location	Purpose	Migration notes
CRASRV.properties	/usr/lpp/rdz/samples/ [/etc/rdz/]	CARMA configuration file	Added support for ephemeral ports
crastart.conf	/usr/lpp/rdz/samples/ [/etc/rdz/]	CARMA configuration file for CRASTART usage	None
crastart.endevor.conf	/usr/lpp/rdz/samples/ [/etc/rdz/]	CARMA configuration file for CRASTART usage for CA Endevor® SCM RAM	Additional DD statements added
include.conf	/usr/lpp/rdz/samples/ [/etc/rdz/]	Forced includes for C/C++ content assist	NEW, customization is optional
ssl.properties	/usr/lpp/rdz/samples/ [/etc/rdz/]	RSE SSL configuration file	None
rsecomm.properties	/usr/lpp/rdz/samples/ [/etc/rdz/]	RSE trace configuration file	Some directives became optional
pushtoclient.properties	/usr/lpp/rdz/samples/ [/etc/rdz/]	Push information to the client configuration file	Additional directives added and existing directives enhanced

Version 8.0.x migration notes

The following migration notes are version 8.0.x-specific. They are valid for migration from IBM Rational Developer for System z version 8.0.1 and version 8.0.2 to version 8.0.3, and are additions to the existing version 8.0.1 migration notes.

- CARMA - The CRAMSG VSAM must be updated (since version 8.0.3).
- CARMA - The CRADEF and CRASTRS VSAM filess for the CA Endevor® SCM RAM must be updated to use the new support for CA Endevor® SCM batch-actions (since version 8.0.3) and CA Endevor® SCM packages (since version 8.0.3).
- CARMA - New sample members have been added (since version 8.0.3):
  - CRABCFG - configuration file for CA Endevor® SCM batch-actions.
  - CRABATCA - sample job for CA Endevor® SCM batch-actions.
- CARMA - The following customizable members have changed (since version 8.0.3):
  - CRANDVRA
- CARMA - Additional DD statements are added for the CA Endevor® SCM RAM (since version 8.0.3):
  - CRABCFG
  - CRABSKEL
  - PKGSCLS (allocated by CRANDVRA)
- File Manager Integration is deprecated.
- Host Configuration Utility - Added migration option (since version 8.0.2)
- PROCLIB - The following PROCLIB members have changed (since version 8.0.3):

  - ELAXFUOP
- RSE - New operator commands have been added to the LOCKD started task (since version 8.0.2):
  - MODIFY DISPLAY TABLE
- RSE - New operator commands have been added to the RSED started task (since version 8.0.2 and 8.0.3):
  - MODIFY IVP ISPF,userid
  - MODIFY IVP PASSTICKET,userid
- RSE - RSED started task operator commands have been enhanced (since version 8.0.2):
  - MODIFY DISPLAY CLIENT [{,LOGON | ,ID | ,USER}]
- RSE - The following console messages are new (since version 8.0.3):
  - FEK910I
- RSE - New non-customizable directives have been added to rsed.envvars (since version 8.0.3):
  - (_RSE_JAVAOPTS) -Dldap.server.address
  - (_RSE_JAVAOPTS) -Dldap.server.port
  - (_RSE_JAVAOPTS) -Dldap.ptc.group.name.suffix
  - _RSE_PTC
- RSE - New optional directives have been added to rsed.envvars (since version 8.0.3):
  - (_RSE_JAVAOPTS) -Daudit.action
  - (_RSE_JAVAOPTS) -Daudit.action.id
  - (_RSE_JAVAOPTS) -DCPP_CLEANUP_INTERVAL
  - (_RSE_JAVAOPTS) -DDSTORE_TCP_NO_DELAY
  - _RSE_FEK_SAF_CLASS
  - _RSE_LDAP_SERVER
  - _RSE_LDAP_PORT
  - _RSE_LDAP_PTC_GROUP_SUFFIX
- RSE - Interpretation of the following optional directives in rsed.envvars has changed (since version 8.0.3):
  - (_RSE_JAVAOPTS) -Dprocess.cleanup.interval
- RSE - New optional directives have been added to pushtoclient.properties (since version 8.0.3):
  - accept.product.license
- RSE - Interpretation of the following optional directives in pushtoclient.properties has changed (since version 8.0.3):
  - config.enabled
  - product.enabled
  - reject.config.updates
  - reject.product.updates
- RSE - New z/OS UNIX samples have been added (since version 8.0.3):
  - process_audit.rex
- New publication, *Rational Developer for System z Messages and Codes* (SC14-7497).

Migrate from version 7.6 to version 8.0.1

These notes are for a migration from a base version 7.6 to version 8.0.1. It includes changes that are already documented as part of version 7.6 maintenance. The changes that are part of the maintenance stream (and thus possibly already implemented) are marked with the release where they were introduced.

IBM Rational Developer for System z, FMID HHOP801

- The default SMP/E install location for MVS and z/OS UNIX components did not change and thus remain `FEK.*` and `/usr/lpp/rdz/*`.
- Application Deployment Manager - Existing `ADN*` modules in the CICS RPL concatenation must be updated (since version 7.6.1).
- Application Deployment Manager - The following sample members have been updated to add URIMAP support in the Administrative utility (since version 7.6.1):
  - ADNJSPAU
  - ADNVCRD
- Application Deployment Manager - An existing CRD repository VSAM must be replaced to enable URIMAP support (since version 7.6.1).
- CARMA - Added support for an universal unique RAM ID stored in the CARMA definition VSAM data set, CRADEF (since version 7.6.1). An existing CRADEF VSAM must be replaced to use it.
- CARMA - Added support for a variable-length layout for the CARMA custom information VSAM data set, CRASTRS (since version 7.6.1). An existing CRASTRS VSAM must be replaced to use it.
- CARMA - New sample members have been added (since version 7.6.1):
  - CRA#VS2 - migrate CRASTRS to variable-length format.
- CARMA - Existing sample members have been renamed:
  - CRA#VCAD -> CRA$VCAD
  - CRA#VCAS -> CRA$VCAS
- CARMA - The CRADEF and CRASTRS VSAMs for the CA Endevor® SCM RAM must be updated.
- CARMA - Additional DD statements are added for the CA Endevor® SCM RAM (since version 7.6.1):
  - EXT2ELM
  - SPCLLIST (allocated by CRANDVRA)
- CARMA – The following customizable members have changed (since version 8.0.1).
  - CRANDVRA
  - CRASHOW
- JES Job Monitor - Usage of _CEE_ENVFILE_S in the started task JCL (since version 7.6.0.1).
- JES Job Monitor - The following `FEJJCNFG` directives became optional (since version 7.6.1):
  - HOST_CODEPAGE
- JES Job Monitor - New optional directives have been added to `FEJJCNFG` (since version 8.0.1):
  - SUBMIT_TIMEOUT

- JES Job Monitor – The following customizable members have changed (since version 8.0.1):
  - FEJTSO
- PROCLIB - New PROCLIB members have been added (since version 7.6.1):
  - ELAXFDCL
- PROCLIB - The following PROCLIB members have changed (since version 8.0.1):
  - LOCKD
  - RSED
- RSE - Usage of 64-bit Java is now supported (since version 7.6.1).
- RSE - Assume permission is granted when the application ID is not defined in the APPL security class (since version 8.0.1).
- RSE - The RSED and LOCKD started tasks now support TMPDIR (since version 8.0.1).
- RSE - The RSED started task now reads the RSED port number from `rsed.envvars` (since version 8.0.1).
- RSE - The LOCKD started task now reads the log level from `rsecomm.properties` (since version 8.0.1).
- RSE - New operator commands have been added (since version 7.6.1 and 8.0.1):
  - MODIFY DISPLAY PROCESS,DETAIL
  - MODIFY IVP DAEMON,userid
- RSE - The following non-customizable directives have changed or are new in `rsed.envvars` (since version 8.0.1):
  - (_RSE_JAVAOPTS) -DDSTORE_KEEPALIVE_RESPONSE_TIMEOUT
  - (_RSE_JAVAOPTS) -DDSTORE_IO_SOCKET_READ_TIMEOUT
  - (_RSE_JAVAOPTS) -Djob.monitor.port
  - CGI_ISPCONF
  - CGI_ISPWORK
  - _EDC_ADD_ERRNO2
- RSE - New optional directives have been added to `rsed.envvars` (since version 7.6.0.1 and 8.0.1):
  - (_RSE_JAVAOPTS) –Daudit.log.mode
  - (_RSE_JAVAOPTS) -Ddeny.nozero.port
  - (_RSE_JAVAOPTS) -Dsingle.logon
  - (_RSE_JAVAOPTS) -Dprocess.cleanup.interval
  - (_RSE_JAVAOPTS) -DDISABLE_DELETE_IN_SUBPROJECT
  - TMPDIR
- RSE - New required directives have been added to rsed.envvars (since version 8.0.1):
  - RSE_RSED_PORT
  - RSE_JMON_PORT
- RSE - The following configuration files are no longer used and should be removed from /etc/rdz (since version 8.0.1):
  - projectcfg.properties
  - propertiescfg.properties
  - uchars.settings
- RSE - The following configuration files are new (since version 8.0.1):
  - pushtoclient.properties

- RSE - The following directories have been renamed (since version 8.0.1):
  - /var/rdz/properties -> /var/rdz/pushtoclient
- RSE - The following directories have been moved (since version 8.0.1):
  - /var/rdz/projects -> /var/rdz/pushtoclient/projects
- RSE - The following console messages have changed or are new (since version 7.6.0.1, 7.6.1 and 8.0.1):
  - FEK001I
  - FEK012I
  - FEK210I
  - FEK900I
  - FEK901I
- RSE - documentation in the FEKSETUP job has exact details on what the job creates, and documents differences between Developer for System z versions (since version 8.0.1).
- RSE - FEKLOGS is enhanced to require less customization and collect more information (since version 8.0.1).

Configurable files

Table 23 gives an overview of files that are customized in version 8.0.1. Note that the Developer for System z sample libraries, FEK.SFEKSAMP, FEK.SFEKSAMV and /usr/lpp/rdz/samples/, come with more customizable members than listed here, such as sample CARMA source code and jobs to compile them.

The following members and files are no longer customizable or no longer used.

- CRA#CRAM (customization no longer required)
- CRAISPRX (deprecated)
- FEKAPPCC (deprecated)
- FEKAPPCL (deprecated)
- FEKAPPCX (deprecated)
- propertiescfg.properties (no longer used)
- projectcfg.properties (no longer used)
- uchars.settings (no longer customizable)

Note: Sample job FEKSETUP copies all listed members to different data sets and directories, default FEK.#CUST.* and /etc/rdz/*.

Table 23. Version 8.0.1 customizations

Member/File	Default location	Purpose	Migration notes
FEKSETUP	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to create data sets and directories, and populate them with customizable files	Updated to include new customizable members and create new directory structure
JMON	FEK.SFEKSAMP(FEJJJCL) [FEK.#CUST.PROCLIB]	JCL for JES Job Monitor	usage of _CEE_ENVFILE_S
FEJJJCL	FEK.SFEKSAMP [FEK.#CUST.PROCLIB(JMON)]	Shipping name for JMON member	See JMON member

Table 23. Version 8.0.1 customizations (continued)

Member/File	Default location	Purpose	Migration notes
RSED	FEK.SFEKSAMP(FEKRSED) [FEK.#CUST.PROCLIB]	JCL for RSE daemon	TMPDIR support
FEKRSED	FEK.SFEKSAMP [FEK.#CUST.PROCLIB(RSED)]	Shipping name for RSED member	See RSED member
LOCKD	FEK.SFEKSAMP(FEKLOCKD) [FEK.#CUST.PROCLIB]	JCL for lock daemon	TMPDIR support
FEKLOCKD	FEK.SFEKSAMP [FEK.#CUST.PROCLIB(LOCKD)]	Shipping name for LOCKD member	See LOCKD member
ELAXF*	FEK.SFEKSAMP [FEK.#CUST.PROCLIB]	JCL for remote project builds, and so on	New member ELAXFDCL
FEKRACF	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL for security definitions	None
FEJJCNFG	FEK.SFEKSAMP [FEK.#CUST.PARMLIB]	JES Job Monitor configuration file	Some directives became optional and new optional directives have been added
FEJTSO	FEK.SFEKSAMP [FEK.#CUST.CNTL]	JCL for TSO submits	Older copies must be replaced by this one (customizations must be redone)
CRA$VMSG	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to create the CARMA message VSAM	None
CRA$VDEF	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to create the CARMA configuration VSAM	None
CRA$VSTR	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to create the CARMA custom information VSAM	None
CRA$VCAD	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to create the CARMA configuration VSAM for CA Endevor® SCM RAM	Renamed, was CRA#VCAS
CRA$VCAS	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to create the CARMA custom information VSAM for CA Endevor® SCM RAM	Renamed, was CRA#VCAS
CRASUBMT	FEK.SFEKSAMP [FEK.#CUST.CNTL]	CARMA batch startup CLIST	Minor changes
CRASUBCA	FEK.SFEKSAMP [FEK.#CUST.CNTL]	CARMA batch startup CLIST for CA Endevor® SCM RAM	Additional DD statements added

Table 23. Version 8.0.1 customizations (continued)

Member/File	Default location	Purpose	Migration notes
CRASHOW	FEK.SFEKSAMP [FEK.#CUST.PARMLIB]	CARMA configuration for CA Endevor® SCM RAM	New filters are added
CRATMAP	FEK.SFEKSAMP [FEK.#CUST.PARMLIB]	CARMA configuration for CA Endevor® SCM RAM	None
CRANDVRA	FEK.SFEKPROC	CARMA allocation REXX for CA Endevor® SCM RAM	Additional DD statements added
CRA#VSLM	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to create the SCLM RAM's message VSAM	None
CRA#ASLM	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to create the SCLM RAM's data sets	None
CRA#VPDS	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to create the PDS RAM's message VSAM	None
CRA#UADD	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to merge RAM definitions	None
CRA#UQRY	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to extract RAM definitions	None
CRAXJCL	FEK.SFEKSAMP [FEK.#CUST.ASM]	Sample source code for IRXJCL replacement	None
CRA#CIRX	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to compile CRAXJCL	None
ADNCSDRS	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to define the RESTful CRD server to primary CICS region	Minor changes
ADNCSDTX	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to define alternate transaction IDs to CICS region	None
ADNTXNC	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to create alternate transaction IDs	None
ADNMSGHC	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to compile ADNMSGHS	None
ADNMSGHS	FEK.SFEKSAMP [FEK.#CUST.COBOL]	Sample source code for the Pipeline Message Handler	None
ADNVCRD	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to create the CRD repository	Added URIMAP support, customizations must be redone
ADNCSDWS	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to define the Web Service CRD server to primary CICS region	None

Table 23. Version 8.0.1 customizations (continued)

Member/File	Default location	Purpose	Migration notes
ADNCSDAR	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to define the CRD server to non-primary CICS regions	Minor changes
ADNJSPAU	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to update the CRD defaults	Added URIMAP support, customizations must be redone
ADNVMFST	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to create and define the Manifest repository	None
ELAXMSAM	FEK.SFEKSAMP [FEK.#CUST.PROCLIB]	JCL procedure of the WLM address space	None
ELAXMJCL	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to define the PL/I and COBOL Stored Procedure Builder to DB2	Minor changes
FEKLOGS	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to collect log files	Added additional checks (customization must be redone)
rsed.envvars	/usr/lpp/rdz/samples/ [/etc/rdz/]	RSE environment variables	Older copies must be replaced by this one (customizations must be redone).
ISPF.conf	/usr/lpp/rdz/samples/ [/etc/rdz/]	TSO/ISPF Client Gateway configuration file	Minor changes
CRASRV.properties	/usr/lpp/rdz/samples/ [/etc/rdz/]	CARMA configuration file	None
crastart.conf	/usr/lpp/rdz/samples/ [/etc/rdz/]	CARMA configuration file for CRASTART usage	Minor changes
crastart.endevor.conf	/usr/lpp/rdz/samples/ [/etc/rdz/]	CARMA configuration file for CRASTART usage for CA Endevor® SCM RAM	Additional DD statements added
ssl.properties	/usr/lpp/rdz/samples/ [/etc/rdz/]	RSE SSL configuration file	None
rsecomm.properties	/usr/lpp/rdz/samples/ [/etc/rdz/]	RSE trace configuration file	Some directives became optional
pushtoclient.properties	/usr/lpp/rdz/samples/ [/etc/rdz/]	Push information to the client configuration file	NEW, customization is optional
FMIEXT.properties	/usr/lpp/rdz/samples/ [/etc/rdz/]	File Manager Integration configuration file	None

Sample migration scenarios

The metadata for host-based projects moved in version 8.0.1 from /var/rdz/projects to /var/rdz/pushtoclient/projects. There are multiple options to migrate the existing metadata.

1. Create a symbolic link that makes /var/rdz/pushtoclient/projects point to /var/rdz/projects, as shown in the following sample z/OS UNIX command:

   ```
   ln -s /var/rdz/projects /var/rdz/pushtoclient/projects
   ```

2. The Developer for System z Client administrator can change the default location for the host-based projects (/var/rdz/pushtoclient/projects) to the actual location (/var/rdz/projects) when exporting the push-to-client metadata.
3. copy the host-based projects metadata from /var/rdz/projects to /var/rdz/pushtoclient/projects, as shown in the following sample z/OS UNIX command:

   ```
   cp -pr /var/rdz/projects /var/rdz/pushtoclient/projects
   ```

See *UNIX System Services Command Reference* (SA22-7802) for more information about the sample z/OS UNIX commands.

Migrate from version 7.5 to version 7.6

IBM Rational Developer for System z, FMID HHOP760

- The default SMP/E install location for MVS and z/OS UNIX components did not change and thus remain FEK.* and /usr/lpp/rdz/*.
- Application Deployment Manager - Existing ADN* modules in the CICS RPL concatenation must be updated.
- Application Deployment Manager - New load modules, which must be part of the CICS RPL concatenation, have been added to support the CICS RESTful interface.
  - ADNANAL
  - ADNCRD41
  - ADNREST
- Application Deployment Manager - New sample members have been added to support the CICS RESTful interface.
  - ADNCSDRS
  - ADNCSDTX
  - ADNTXNC
- Application Deployment Manager - Existing sample members are renamed.
  - ADNARCSD -> ADNCSDAR
  - ADNCMSGH -> ADNMSGHC
  - ADNMFEST -> ADNVMFST
  - ADNPCCSD -> ADNCSDWS
  - ADNSMSGH -> ADNMSGHS
  - ADNVSAM -> ADNVCRD
- A new, production type, RAM is provided to access CA Endevor® SCM.
  - CRARNDVR
- CARMA - New sample members have been provided to support the CA Endevor® SCM RAM.

  - FEK.#CUST.JCL(CRA#VCAD)
  - FEK.#CUST.JCL(CRA#VCAS)
  - FEK.#CUST.CNTL(CRASUBCA)
  - FEK.#CUST.PARMLIB(CRASHOW)
  - FEK.#CUST.PARMLIB(CRATMAP)
  - FEK.SFEKPROC(CRANDVRA)
  - /etc/rdz/crastart.endevor.conf
- CARMA - New sample members have been provided to support merging RAM definitions.
  - CRA#UADD
  - CRA#UQRY
- File Manager Integration - The batch interface to access File Manager is no longer supported.
- File Manager Integration - The `FMIEXT.properties` configuration file has changed completely and must be replaced.
- JES Job Monitor - LE options are embedded in the `FEJJMON` load module (since version 7.5.0.1), which might require changes to your started task definition. See the `FEK.SFEKSAMP(FEJJJCL)` sample JCL for more details.
- JES Job Monitor - New optional directives have been added to `FEJJCNFG` (in version 7.5.0.1 and 7.5.1.0).
  - APPLID
  - CONSOLE_NAME
  - GEN_CONSOLE_NAME
- JES Job Monitor - A new command, Show JCL, is supported (since version 7.5.1.0) which might require updates to your security software.
- Lock daemon – The lock daemon (`LOCKD`) is a new started task (since version 7.5.0.1). This started task can be queried to identify which Developer for z client is holding a data set lock. (System commands stop at address space level, which is the RSE thread pool.)
- SCLMDT - the default location for the SCLMDT project configuration files has changed.
  - /var/rdz/sclmdt
- RSE - New operator commands have been added.
  - MODIFY RSESTANDARDLOG
- RSE - New required directives have been added to rsed.envvars (in version 7.5.0.1 and 7.6.0.0).
  - _RSE_LOCKD_PORT
  - (_RSE_JAVAOPTS) -Dlock.daemon.port
  - (_RSE_JAVAOPTS) -Dlock.daemon.cleanup.interval
  - _RSE_LOCKD_CLASS
  - _RSE_HOST_CODEPAGE
  - (_RSE_JAVAOPTS) -Dfile.encoding
  - (_RSE_JAVAOPTS) -Dconsole.encoding
- RSE - New optional directives have been added to `rsed.envvars` (since version 7.5.0.1, 7.5.1.0 and 7.6.0.0).
  - (_RSE_JAVAOPTS) -Duser.log
  - (_RSE_JAVAOPTS) -Dkeep.last.log

- (_RSE_JAVAOPTS) -Denable.standard.log
- (_RSE_JAVAOPTS) -DDSTORE_LOG_DIRECTORY
- (_RSE_JAVAOPTS) -DHIDE_ZOS_UNIX
- (_RSE_JAVAOPTS) -Denable.certificate.mapping
- GSK_CRL_SECURITY_LEVEL
- GSK_LDAP_SERVER
- GSK_LDAP_PORT
- GSK_LDAP_USER
- GSK_LDAP_PASSWORD

- RSE - Some optional directives have changed in `rsed.envvars`.
  - (_RSE_JAVAOPTS) -Ddaemon.log
  - (_RSE_JAVAOPTS) -Xmx
  - SCLMDT_CONF_HOME
- RSE - New optional directives have been added to `ssl.properties` (since version 7.5.1.0 and 7.6.0.0).
  - server_keystore_label
  - server_keystore_type
- RSE - RSE daemon supports X.509 client certificate authentication (since version 7.5.1.0), which requires updates to your current certificate and security setup when used.
- RSE - Security has been tightened, failing connection requests upon PassTicket and FEKAPPL errors.
- RSE - The default location for all log files (daemon and user logs) has changed.
  - /var/rdz/logs
  - /var/rdz/logs/$LOGNAME
- RSE - A new sample JCL has been provided to gather Developer for System z logs and configuration information.
  - FEKLOGS

Configurable files

Table 24 gives an overview of files that are customized in version 7.6. Note that the Developer for System z sample libraries, `FEK.SFEKSAMP`, `FEK.SFEKSAMV` and `/usr/lpp/rdz/samples/`, come with more customizable members than listed here, such as sample CARMA source code and jobs to compile them.

Note: Sample job `FEKSETUP` copies all listed members to different data sets and directories, default `FEK.#CUST.*` and `/etc/rdz/*`.

Table 24. Version 7.6 customizations

Member/File	Default location	Purpose	Migration notes
FEKSETUP	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to create data sets and directories, and populate them with customizable files	Updated to include new customizable members
JMON	FEK.SFEKSAMP(FEJJJCL) [FEK.#CUST.PROCLIB]	JCL for JES Job Monitor	Added option to change LE options

Table 24. Version 7.6 customizations (continued)

Member/File	Default location	Purpose	Migration notes
FEJJJCL	FEK.SFEKSAMP [FEK.#CUST.PROCLIB(JMON)]	Shipping name for JMON member	See JMON member
RSED	FEK.SFEKSAMP(FEKRSED) [FEK.#CUST.PROCLIB]	JCL for RSE daemon	none
FEKRSED	FEK.SFEKSAMP [FEK.#CUST.PROCLIB(RSED)]	Shipping name for RSED member	See RSED member
LOCKD	FEK.SFEKSAMP(FEKLOCKD) [FEK.#CUST.PROCLIB]	JCL for lock daemon	NEW, customization is required
FEKLOCKD	FEK.SFEKSAMP [FEK.#CUST.PROCLIB(LOCKD)]	Shipping name for LOCKD member	See LOCKD member
ELAXF*	FEK.SFEKSAMP [FEK.#CUST.PROCLIB]	JCL for remote project builds, and so forth	none
FEKRACF	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL for security definitions	Minor updates
FEJJCNFG	FEK.SFEKSAMP [FEK.#CUST.PARMLIB]	JES Job Monitor configuration file	New optional directives have been added
FEJTSO	FEK.SFEKSAMP [FEK.#CUST.CNTL]	JCL for TSO submits	none
CRA$VMSG	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to create the CARMA message VSAM	none
CRA$VDEF	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to create the CARMA configuration VSAM	none
CRA$VSTR	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to create the CARMA custom information VSAM	none
CRASUBMT	FEK.SFEKSAMP [FEK.#CUST.CNTL]	CARMA batch startup CLIST	none
CRASUBCA	FEK.SFEKSAMP [FEK.#CUST.CNTL]	CARMA batch startup CLIST for CA Endevor® SCM RAM	NEW, customization is optional
CRASHOW	FEK.SFEKSAMP [FEK.#CUST.PARMLIB]	CARMA configuration for CA Endevor® SCM RAM	NEW, customization is optional
CRATMAP	FEK.SFEKSAMP [FEK.#CUST.PARMLIB]	CARMA configuration for CA Endevor® SCM RAM	NEW, customization is optional

Table 24. Version 7.6 customizations (continued)

Member/File	Default location	Purpose	Migration notes
CRANDVRA	FEK.SFEKPROC	CARMA allocation REXX for CA Endevor® SCM RAM	NEW, customization is optional
CRAISPRX	FEK.SFEKSAMP [FEK.#CUST.CNTL]	Sample DD allocation exec for CARMA using TSO/ISPF Client Gateway	none
CRA#VSLM	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to create the SCLM RAM's message VSAM	none
CRA#ASLM	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to create the SCLM RAM's data sets	none
CRA#VPDS	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to create the PDS RAM's message VSAM	none
CRA#CRAM	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to compile the skeleton RAM	none
CRA#VCAD	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to create the CARMA configuration VSAM for CA Endevor® SCM RAM	NEW, customization is optional
CRA#VCAS	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to create the CARMA custom information VSAM for CA Endevor® SCM RAM	NEW, customization is optional
CRA#UADD	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to merge RAM definitions	NEW, customization is optional
CRA#UQRY	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to extract RAM definitions	NEW, customization is optional
CRAXJCL	FEK.SFEKSAMP [FEK.#CUST.ASM]	Sample source code for IRXJCL replacement	none
CRA#CIRX	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to compile CRAXJCL	none
ADNCSDRS	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to define the RESTful CRD server to primary CICS region	NEW, customization is optional
ADNCSDTX	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to define alternate transaction IDs to CICS region	NEW, customization is optional
ADNTXNC	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to create alternate transaction IDs	NEW, customization is optional
ADNMSGHC	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to compile ADNMSGHS	Renamed, was ADNCMSGH

Table 24. Version 7.6 customizations (continued)

Member/File	Default location	Purpose	Migration notes
ADNMSGHS	FEK.SFEKSAMP [FEK.#CUST.COBOL]	Sample source code for the Pipeline Message Handler	Renamed, was ADNSMSGH
ADNVCRD	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to create the CRD repository	Renamed, was ADNVSAM
ADNCSDWS	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to define the Web Service CRD server to primary CICS region	Renamed, was ADNPCCSD
ADNCSDAR	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to define the CRD server to non-primary CICS regions	Renamed, was ADNARCSD
ADNJSPAU	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to update the CRD defaults	Definitions for the RESTful service are added, customizations must be redone
ADNVMFST	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to create and define the Manifest repository	Renamed, was ADNMFEST
ELAXMSAM	FEK.SFEKSAMP [FEK.#CUST.PROCLIB]	JCL procedure of the WLM address space for the PL/I and COBOL Stored Procedure Builder	none
ELAXMJCL	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to define the PL/I and COBOL Stored Procedure Builder to DB2	none
FEKAPPCC	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to create an APPC transaction	none
FEKAPPCL	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to display an APPC transaction	none
FEKAPPCX	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to delete an APPC transaction	none
FEKLOGS	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to collect log files	NEW, customization is optional
rsed.envvars	/usr/lpp/rdz/samples/ [/etc/rdz/]	RSE environment variables	Older copies must be replaced by this one (customizations must be redone).
ISPF.conf	/usr/lpp/rdz/samples/ [/etc/rdz/]	TSO/ISPF Client Gateway configuration file	ISP.SISPCLIB added to SYSPROC for SCLMDT

Table 24. Version 7.6 customizations (continued)

Member/File	Default location	Purpose	Migration notes
CRASRV.properties	/usr/lpp/rdz/samples/ [/etc/rdz/]	CARMA configuration file	none
crastart.conf	/usr/lpp/rdz/samples/ [/etc/rdz/]	CARMA configuration file for CRASTART usage	none
crastart.endevor.conf	/usr/lpp/rdz/samples/ [/etc/rdz/]	CARMA configuration file for CRASTART usage for CA Endevor® SCM RAM	NEW, customization is optional
ssl.properties	/usr/lpp/rdz/samples/ [/etc/rdz/]	RSE SSL configuration file	New optional directives have been added
rsecomm.properties	/usr/lpp/rdz/samples/ [/etc/rdz/]	RSE trace configuration file	none
propertiescfg.properties	/usr/lpp/rdz/samples/ [/etc/rdz/]	Host-based property groups configuration file	none
projectcfg.properties	/usr/lpp/rdz/samples/ [/etc/rdz/]	Host-based projects configuration file	none
FMIEXT.properties	/usr/lpp/rdz/samples/ [/etc/rdz/]	File Manager Integration configuration file	Older copies must be replaced by this one (customizations must be redone).
uchars.settings	/usr/lpp/rdz/samples/ [/etc/rdz/]	Uneditable characters configuration file	none

Migrate from version 7.1 to version 7.5

IBM Rational Developer for System z, FMID HHOP750

- The default SMP/E install location for MVS components did not change and thus remains `FEK.*`.
- The default SMP/E install location for z/OS UNIX components changed to `/usr/lpp/rdz/*`.
- Common Access Repository Manager (CARMA) has merged into Developer for System z Version 7.5, disabling the need to install it as a separate product.
- SCLM Developer Toolkit has merged into Developer for System z Version 7.5, disabling the need to install it as a separate product.
- In version 7.5 ISPF's TSO/ISPF Client Gateway service replaces the SCLM Developer Toolkit function used in version 7.1 to connect to the TSO Commands service. The APPC connection method is still supported.
- In version 7.5, RSE server is no longer an `INETD` managed process but a started task. RSE server now also uses a single server model whereas with previous versions, each client-host connection had a private RSE server.

- All modules requiring APF authorization (JES Job Monitor and SCLM Developer Toolkit) moved to FEK.SFEKAUTH in version 7.5, requiring an update to the existing APF definitions.
- JES Job Monitor load module moved to FEK.SFEKAUTH in version 7.5, requiring an update to the existing started task procedure.
- CARMA load modules moved to new libraries, requiring an update to the existing CRASUBMT server startup script.
- SCLM Developer Toolkit load modules moved to new libraries, requiring and update to the existing LINKLIST definitions.
- ELAXFTSO is a new sample build procedure since version 7.1.1, ELAXFCP1 and ELAXFPP1 are new in version 7.5.
- uchars.settings is a new configuration file for uneditable characters.
- propertiescfg.properties is a new configuration file for default property groups.
- FEJJCNFG, CRASRV.properties and FMIEXT.properties have new optional directives.
- rsed.envvars has changed in version 7.5 and must be replaced.
- The sample ISPF.conf file shipped with version 7.5 is similar to the one used by SCLM Developer Toolkit in version 7.1.
- Some of the existing Application Deployment manager customizations must be redone.
- Application Deployment Manager has new functions which require customization.
- The security settings for RSE server changed drastically in version 7.5.
- The MVS.MCSOPER.JMON security profile is new for JES Job monitor in version 7.5.
- The CARMA startup script changed name and moved to a new location, requiring an update to the existing CRASRV.properties configuration file.
- The FMI startup script changed name and moved to a new location, requiring an update to the existing FMIEXT.properties configuration file.
- A new load module has been added for bidirectional support, requiring an update to the existing CICS DFHRPL concatenation if you are not using the FEK.SFEKLOAD library.
- Changes to the MAXPROCUSER parameter of SYS1.PARMLIB(BPXPRMxx) are now also documented.

Configurable files

Table 25 gives an overview of files that are customized in version 7.5. Note that the Developer for System z sample libraries, FEK.SFEKSAMP, FEK.SFEKSAMV and /usr/lpp/rdz/samples/, come with more customizable members than listed here, such as sample CARMA source code and jobs to compile them.

Note: Sample job FEKSETUP copies all listed members to different data sets and directories, default FEK.#CUST.* and /etc/rdz/*.

Table 25. Version 7.5 customizations

Member/File	Default location	Purpose	Migration notes
FEKSETUP	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to create data sets and directories, and populate them with customizable files	NEW, customization is required

Table 25. Version 7.5 customizations (continued)

Member/File	Default location	Purpose	Migration notes
JMON	FEK.SFEKSAMP(FEJJJCL) [FEK.#CUST.PROCLIB]	JCL for JES Job Monitor	STEPLIB changed to SFEKAUTH
RSED	FEK.SFEKSAMP(FEKRSED) [FEK.#CUST.PROCLIB]	JCL for RSE daemon	NEW, customization is required
ELAXF*	FEK.SFEKSAMP [FEK.#CUST.PROCLIB]	JCL for remote project builds, and so on	ELAXFTSO, ELAXFCP1 and ELAXFPP1 are new
FEKRACF	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL for security definitions	NEW, required
FEJJCNFG	FEK.SFEKSAMP [FEK.#CUST.PARMLIB]	JES Job Monitor configuration file	• Some directives became optional • New optional directives have been added
FEJTSO	FEK.SFEKSAMP [FEK.#CUST.CNTL]	JCL for TSO submits	Job name can now be a variable
CRAISPRX	FEK.SFEKSAMP [FEK.#CUST.CNTL]	Sample DD allocation exec for CARMA using TSO/ISPF Client Gateway	NEW, customization is optional
CRAXJCL	FEK.SFEKSAMP [FEK.#CUST.ASM]	Sample source code for IRXJCL replacement	NEW, customization is optional
CRA#CIRX	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to compile CRAXJCL	NEW, customization is optional
ADNSMSGH	FEK.SFEKSAMP [FEK.#CUST.COBOL]	Sample source code for the Pipeline Message Handler	Older copies must be replaced by this one (customizations must be redone)
ADNPCCSD	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to define the CRD server to primary CICS region	Older copies must be replaced by this one (customizations must be redone)
ADNJSPAU	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to update the CRD defaults	NEW, customization is optional
ADNMFEST	FEK.SFEKSAMP [FEK.#CUST.JCL]	JCL to create and define the Manifest repository	NEW, customization is optional
rsed.envvars	/usr/lpp/rdz/samples/ [/etc/rdz/]	RSE environment variables	Older copies must be replaced by this one (customizations must be redone)

Table 25. Version 7.5 customizations (continued)

Member/File	Default location	Purpose	Migration notes
ISPF.conf	/usr/lpp/rdz/samples/ [/etc/rdz/]	TSO/ISPF Client Gateway configuration file	Identical to the ISPF.conf shipped with SCLMDT in v7.1
CRASRV.properties	/usr/lpp/rdz/samples/ [/etc/rdz/]	CARMA configuration file	• Startup script changed location and name • New optional directives have been added
crastart.conf	/usr/lpp/rdz/samples/ [/etc/rdz/]	CARMA configuration file for CRASTART usage	NEW, customization is optional
FMIEXT.properties	/usr/lpp/rdz/samples/ [/etc/rdz/]	File Manager Integration configuration file	• Startup script changed location and name • New optional directives have been added
uchars.settings	/usr/lpp/rdz/samples/ [/etc/rdz/]	Uneditable characters configuration file	NEW, customization is optional

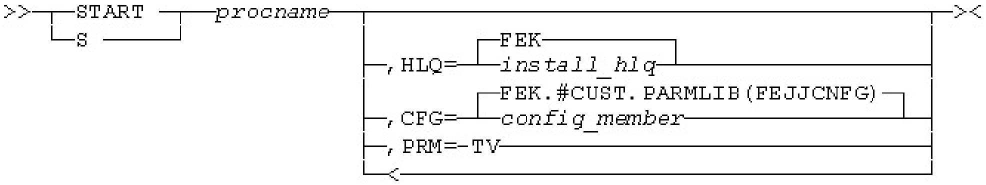
Chapter 10. Operator commands

This appendix summarizes the operator (or console) commands information in *Rational Developer for System z Host Configuration Guide* (SC23-7658). Refer to that publication for more details.

Start (S)

Use the **START** command to dynamically start a started task (STC). The abbreviated version of the command is the letter `S`.

JES Job Monitor



```
>>─┬─START ─┬─procname─┬──────────────────────────────────────┬─><
   └─S ─────┘          │        ┌─FEK──────────┐              │
                       ├─,HLQ=──┴─install_hlq──┴──────────────┤
                       │        ┌─FEK.#CUST.PARMLIB(FEJJCNFG)─┐│
                       ├─,CFG=──┴─config_member───────────────┴┤
                       ├─,PRM=-TV──────────────────────────────┤
                       └─<─────────────────────────────────────┘
```

Figure 35. START JMON operator command

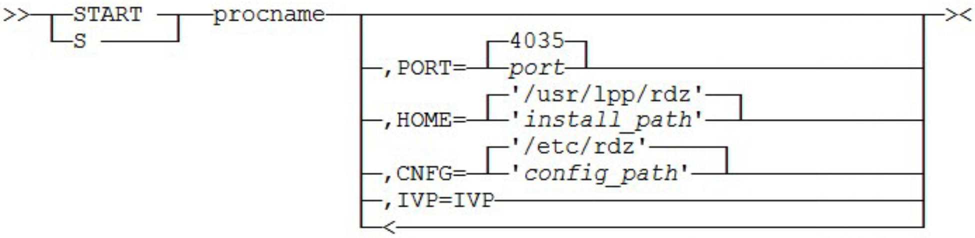
procname
: The name of the member in a procedure library that is used to start the server. The default name used during the host configuration is `JMON`.

HLQ=install_hlq
: High-level qualifier used to install Developer for System z. The default is `FEK`.

CFG=config_member
: Absolute data set and member name of the JES Job Monitor configuration file. The default is `FEK.#CUST.PARMLIB(FEJJCNFG)`.

PRM=-TV
: Enable verbose (trace) mode. Tracing will cause performance degradations and should only be done under the direction of the IBM support center.

RSE daemon



```
>>─┬─START ─┬─procname─┬──────────────────────────────────┬─><
   └─S ─────┘          │         ┌─4035─┐                 │
                       ├─,PORT=──┴─port─┴─────────────────┤
                       │         ┌─'/usr/lpp/rdz'───┐     │
                       ├─,HOME=──┴─'install_path'───┴─────┤
                       │         ┌─'/etc/rdz'─────┐       │
                       ├─,CNFG=──┴─'config_path'──┴───────┤
                       ├─,IVP=IVP─────────────────────────┤
                       └─<────────────────────────────────┘
```

Figure 36. START RSED operator command

procname
The name of the member in a procedure library that is used to start the server. The default name used during the host configuration is RSED.

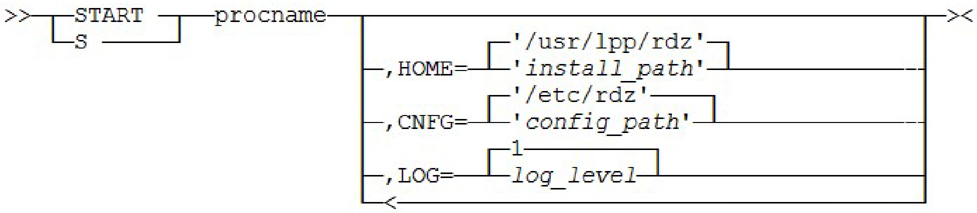
PORT=port
The port used by the RSE daemon for the clients to connect. If not specified, then the port defined in /etc/rdz/rsed.envvars is used (variable _RSE_RSED_PORT). The default is 4035.

IVP=IVP
Do not start the server but run the RSE daemon installation verification program (IVP).

CNFG='config_path'
Absolute location of the configuration files stored in z/OS UNIX. The default is '/etc/rdz'. Note that the z/OS UNIX path is case-sensitive and that it must be enclosed in single quotes (') to preserve lower case characters.

HOME='install_path'
Path prefix and the mandatory /usr/lpp/rdz used to install Developer for System z. The default is '/usr/lpp/rdz'. Note that the z/OS UNIX path is case-sensitive and that it must be enclosed in single quotes (') to preserve lower case characters.

Lock daemon



```
>>──┬─START─┬──procname──┬──────────────────────────────────────────┬──><
    └─S─────┘            │                                          │
                         │          ┌─'/usr/lpp/rdz'─┐              │
                         ├─,HOME=───┴─'install_path'─┴──────────────┤
                         │          ┌─'/etc/rdz'─────┐              │
                         ├─,CNFG=───┴─'config_path'──┴──────────────┤
                         │         ┌─1─────────┐                    │
                         ├─,LOG=───┴─log_level─┴────────────────────┤
                         └─<────────────────────────────────────────┘
```

Figure 37. START LOCKD operator command

procname
The name of the member in a procedure library that is used to start the server. The default name used during the host configuration is LOCKD.

LOG=log_level
The detail level of output in DD STDOUT.

- 0 : Log error messages only.
- 1 : Log error and warning messages (default).
- 2 : Log error, warning, and informational messages.

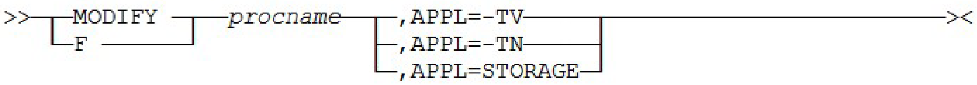
CNFG='config_path'
Absolute location of the configuration files stored in z/OS UNIX. The default is '/etc/rdz'. Note that the z/OS UNIX path is case-sensitive and that it must be enclosed in single quotes (') to preserve lower case characters.

HOME='install_path'
Path prefix and the mandatory /usr/lpp/rdz used to install Developer for System z. The default is '/usr/lpp/rdz'. Note that the z/OS UNIX path is case-sensitive and that it must be enclosed in single quotes (') to preserve lower case characters.

Modify (F)

The **MODIFY** command allows you to dynamically query and change the characteristics of an active task. The abbreviated version of the command is the letter F.

JES Job Monitor



```
>>─┬─MODIFY ─┬──procname──┬─,APPL=-TV──────┬──────────────────────────────><
   └─F ──────┘            ├─,APPL=-TN──────┤
                          └─,APPL=STORAGE──┘
```

Figure 38. MODIFY JMON operator command

procname
The name of the member in a procedure library that was used to start the server. The default name used during the host configuration is JMON.

-TV Enable verbose (trace) mode. Tracing will cause performance degradations and should only be done under the direction of the IBM support center. Message "Job Monitor TRACE_LEVEL_VERBOSE" is written to DD SYSOUT and to the console with message ID BPXM023I.

-TN Disable verbose (trace) mode. Message "Job Monitor TRACE_LEVEL_NONE" is written to DD SYSOUT and to the console with message ID BPXM023I.

STORAGE
Write a storage usage report to DD SYSOUT. Message "Job Monitor storage information written to SYSOUT" is written to the console with message ID BPXM023I. The storage usage report shows various storage related fields with sizes in bytes, kilobytes, and megabytes.

```
>>>STORAGE TRACE (console request)<<<
LDAREGRQ    00000000000 00000000K 00000M requested region size
  below 16M line
LDASIZA     00006266880 00006120K 00005M maximum region size
LDALIMIT    00006266880 00006120K 00005M limit
LDAVVRG     00006266880 00006120K 00005M getmain limit
LDALOAL     00000061440 00000060K 00000M in use
LDAHIAL     00000266240 00000260K 00000M LSQA/SWA/private subpools
_GAP        00000000000 00000000K 00000M gaps in allocation
_AVAIL      00005939200 00005800K 00005M available (including gaps)
_MAX        00006000640 00005860K 00005M current limit
  above 16M line
LDAESIZA    01905262592 01860608K 01817M maximum region size
LDAELIM     01905262592 01860608K 01817M limit
LDAEVVRG    01905262592 01860608K 01817M getmain limit
LDAELOAL    00000937984 00000916K 00000M in use
LDAEHIAL    00012754944 00012456K 00012M ELSQA/ESWA/private subpools
_EGAP       00000000000 00000000K 00000M gaps in allocation
_EAVAIL     01891569664 01847236K 01803M available (including gaps)
_EMAX       01892507648 01848152K 01804M current limit
```

RSE daemon

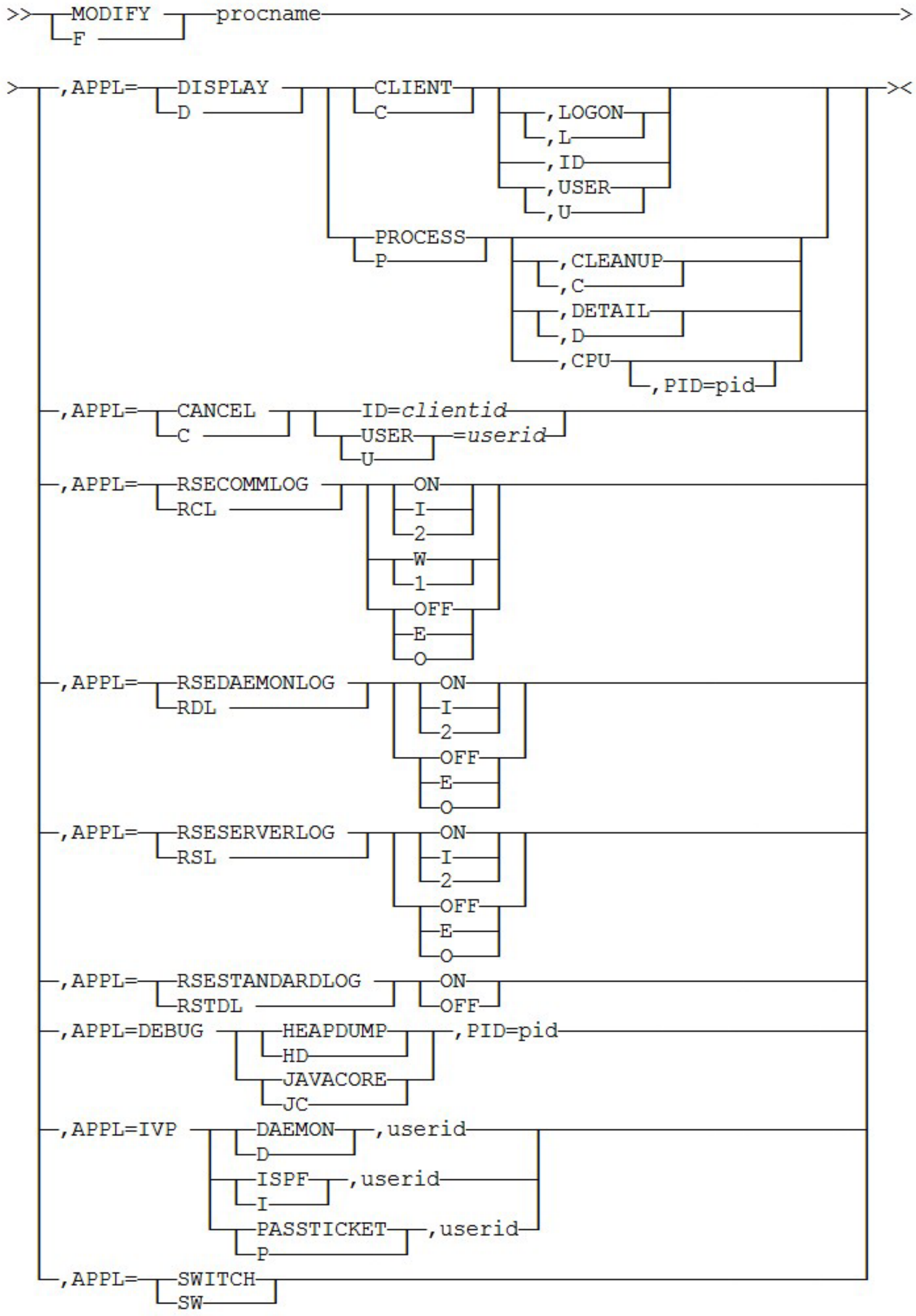


Figure 39. MODIFY RSED operator command

procname
The name of the member in a procedure library that was used to start the server. The default name used during the host configuration is RSED.

DISPLAY CLIENT[{,LOGON | ,ID | ,USER}]
Display the active clients in a single BPXM023I message. The result layout

depends on which command option was used. You can change the sorting order with the optional command arguments.

- No command option: Clients are grouped by the thread pool process that serves them.

  ```
  ProcessId(<processid>) ASId(<asid>) JobName(<jobname>)
  Clients(<local>/<total>) Order(<startup order>)
  <clientid><userid><connected since>
  ```

- LOGON command option: Clients are ordered by logon time.

  ```
  LOGON TIME------------------ ID----- USERID--
  <connected since>         <clientid> <userid>
  ```

- ID command option: Clients are ordered by client ID.

  ```
  ID----- USERID-- LOGON TIME------------------
  <clientid> <userid> <connected since>
  ```

- USER command option: Clients are ordered by user ID.

  ```
  USERID-- ID----- LOGON TIME------------------
  <userid> <clientid> <connected since>
  ```

DISPLAY PROCESS[{,CLEANUP | ,CPU [,PID=pid] | ,DETAIL}]

Display the RSE thread pool processes in one or more BPXM023I messages. There can be multiple processes, which are used for load balancing the connected users.

```
ProcessId(<processid>) Memory Usage(<java heap usage>%)
  Clients(<number of clients>) Order(<startup order>) <error status>
```

Note:

- `<processid>` can be used in process specific z/OS UNIX operator commands.
- Each process has its own Java heap, whose size can be set in `rsed.envvars`. Note that the reported Java heap usage will include storage released by Developer for System z, but which is not yet freed by Java's garbage collection process.
- `<startup order>` is a sequential number that indicates the order that the thread pools were started. The number corresponds to the number used in the filename of the `stderr.*.log` and `stdout.*.log` files.

In normal situations, `<error status>` is blank. Table 26 documents the possible non-blank values for `<error status>`.

Table 26. Thread pool error status

Status	Description
severe error	The thread pool process encountered an unrecoverable error and halted operations. The other status fields show the last known values. Use the CLEANUP option of the **DISPLAY PROCESS** modify command to remove this entry from the table.
killed process	The thread pool process was killed by Java, z/OS UNIX or an operator command. The other status fields show the last known values. Use the CLEANUP option of the **DISPLAY PROCESS** modify command to remove this entry from the table.

Table 26. Thread pool error status (continued)

Status	Description
timeout	The thread pool process did not respond in a timely manner to RSE daemon during a client connect request. The other status fields show the current values. The thread pool is excluded for future client connect requests. The *timeout* status is reset when a client served by this thread pool logs off.

More information is provided when the DETAIL option of the **DISPLAY PROCESS** modify command is used:

```
ProcessId(33555087) ASId(002E) JobName(RSED8) Order(1)
 PROCESS LIMITS:    CURRENT  HIGHWATER      LIMIT
  JAVA HEAP USAGE(%)     10         56        100
  CLIENTS                 0         25         60
  MAXFILEPROC            83        103      64000
  MAXPROCUSER            97         99        200
  MAXTHREADS              9         14       1500
  MAXTHREADTASKS          9         14       1500
```

The ASId field is the address space ID, in hexadecimal notation. The process limits table shows the current resource usage, the high-water mark for the resource usage, and the resource limit. Note that due to other limiting factors, the defined limit might never be reached.

The CPU option of the **DISPLAY PROCESS** modify command will show the accumulated CPU usage (in milliseconds) of each thread in a thread pool. There will be a BPXM023I message per thread pool. By default all thread pools will report the CPU usage, but you can limit the scope to a single thread pool by specifying `PID=pid` on the operator command, where `pid` is the process ID of the target thread pool.

```
ProcessId(421     ) ASId(007D) JobName(RSED8) Order(1)
USERID   THREAD-ID        TCB@     ACC_TIME TAG
STCRSE   0EDE540000000000 005E6B60      822 1/ThreadPoolProcess
STCRSE   0EDE870000000001 005E69C8      001
STCRSE   0EDE980000000002 005E6518     1814
STCRSE   0EDEBA0000000003 005E66B0     2305
STCRSE   0EDECB0000000004 005E62F8      001
STCRSE   0EDEDC0000000005 005E60D8      001
STCRSE   0EDF860000000006 005C2BF8      628 6/ThreadPoolMonitor$Memory
UsageMonitor
STCRSE   0EDF970000000007 005C2D90      003 7/ThreadPoolMonitor
STCRSE   0EDFDB0000000008 005C29D8      001
STCRSE   0EE22E000000000E 005C1BE0      070
IBMUSER  0EE0EB0000000011 005C22B8      276 20/ServerReceiver
IBMUSER  0EE2500000000012 005C19C0      137 16/ServerUpdateHandler
IBMUSER  0EE2610000000013 005C17A0      509 15/ServerCommandHandler
IBMUSER  0EE1840000000014 005C1E00      065 21/ZosSystemMiner
STCRSE   0EE1510000000016 005C2098      078
STCRSE   0EE1950000000017 005C1580      001
IBMUSER  0EE23F0000000018 005C1360      021 26/UniversalFileSystemMine
r
IBMUSER  0EE2A5000000001C 005C0CF0      003 27/EnvironmentMiner
IBMUSER  0EE283000000001D 005C1140      002 31/CommandMiner
IBMUSER  0EE272000000001E 005C0E88      081 32/MVSFileSystemMiner
IBMUSER  0EE294000000001F 005C0AD0      002 33/MVSByteStreamHandler$Op
enCloseThread
STCRSE   0EE2E90000000023 005C0470      001
IBMUSER  0EE2C70000000024 005C08B0      050 38/JESMiner
IBMUSER  0EE2B60000000026 005C0690      004 40/FAMiner
IBMUSER  0EE30B0000000027 005C0250      002 41/LuceneMiner
IBMUSER  0EE31C0000000028 005C0030      002 42/CDTParserMiner
IBMUSER  0EE32D0000000029 005BDE00      002 43/MVSLuceneMiner
IBMUSER  0EE33E000000002A 005BDBE0      002 44/CDTMVSParserMiner
```

If the output size exceeds the maximum number of lines for a console message, the output will be split over multiple BPXM023I messages. These additional messages will have the same header as the first message, but with the `CONTINUATION` keyword added to the first line.

```
ProcessId(421     ) ASId(007D) JobName(RSED8) Order(1) CONTINUATION
USERID   THREAD-ID        TCB@     ACC_TIME TAG
```

CANCEL ID=clientid

Cancel a client connection based upon the client ID, which is shown in the **DISPLAY CLIENT** modify command.

Note that when a client connection is cancelled, the host threads will go through normal termination processing to clean up resources used by them. This implies that some threads can take a few minutes before they end. (For example, because they are waiting on the keep-alive mechanism to time out.)

CANCEL USER=userid

Cancel a client connection based upon the client's user ID, which is shown in the **DISPLAY CLIENT** modify command.

Note that when a client connection is cancelled, the host threads will go through normal termination processing to clean up resources used by them. This implies that some threads can take a few minutes before they end. (For example, because they are waiting on the keep-alive mechanism to time out.)

RSECOMMLOG {ON | OFF | I | W | E | 2 | 1 | 0}

Control the trace detail level for RSE server (`rsecomm.log`) and the MVS data set services (`lock.log` and `ffs*.log`). The startup default is defined in `rsecomm.properties`. There are three detail levels available:

E or 0 or OFF	Error messages only.
W or 1	Error and Warning messages. This is the default setting in `rsecomm.properties`.
I or 2 or ON	Error, Warning and Informational messages.

Detailed tracing will cause performance degradations and should only be done under the direction of the IBM support center.

RSEDAEMONLOG {ON | OFF | I | E | 2 | 0}

Control the trace detail level for RSE daemon (`rsedaemon.log`). The startup default is defined in `rsecomm.properties`. There are two detail levels available:

E or 0 or OFF	Error messages only.
I or 2 or ON	Error, Warning, and Informational messages.

Detailed tracing will cause performance degradations and should only be done under the direction of the IBM support center.

RSESERVERLOG {ON | OFF | I | E | 2 | 0}

Control the trace detail level for RSE thread pools (rseserver.log). The startup default is defined in rsecomm.properties. There are two detail levels available:

E or 0 or OFF	Error messages only.
I or 2 or ON	Error, Warning, and Informational messages.

Detailed tracing will cause performance degradations and should only be done under the direction of the IBM support center.

RSESTANDARDLOG {ON |, OFF}

Disable (OFF) or enable (ON) updating the log files holding the stdout and stderr streams of the thread pools (stdout.*.log and stderr.*.log). The startup default is defined by the enable.standard.log directive in rsed.envvars.

Detailed tracing will cause performance degradations and should only be done under the direction of the IBM support center.

DEBUG HEAPDUMP,PID=pid

Request a Java Heap dump for a specified thread pool (where pid is the process ID of an RSE thread pool). The dump is written to the directory specified by _CEE_DUMPTARG in rsed.envvars (default value is /tmp). Results are shown in a single BPXM023I console message.

```
BPXM023I (STCRSE)
JVMDUMP034I User requested Heap dump using '/tmp/heapdump.20120223.211'
430.16777590.0001.phd' through JVMRI
```

DEBUG JAVACORE,PID=pid

Request a Java Core dump for a specified thread pool (where pid is the process ID of an RSE thread pool). The dump is written to the directory specified by _CEE_DUMPTARG in rsed.envvars (default value is /tmp). Results are shown in a single BPXM023I console message.

```
BPXM023I (STCRSE)
JVMDUMP034I User requested Java dump using '/tmp/javacore.20120223.214
244.16777590.0002.phd' through JVMRI
```

IVP DAEMON,userid

Log user ID userid on to RSE daemon to do a connection test. Results are shown with one ore more FEK900I console messages. The return code is shown with console message FEK901I.

```
+FEK900I DAEMON IVP: SSL is disabled
+FEK900I DAEMON IVP: connected
+FEK900I DAEMON IVP: 1977
+FEK900I DAEMON IVP: 6902918
+FEK900I DAEMON IVP: Success
+FEK901I DAEMON IVP  Exit code = 0
```

Note:

- The function is similar to what the fekfivpd IVP (Installation Verification Program) does.
- RSE daemon will generate a PassTicket which is used as password for the IVP, so there will be no WTOR (Write To Operator with Reply) requesting a password.

IVP ISPF,userid

Invoke ISPF's Client Gateway as user ID `userid`. Results are shown with one ore more FEK900I console messages. The return code is shown with console message FEK901I.

```
+FEK900I ISPF IVP: executed on CDFMVS08 -- Tue Sep 13 22:29:28 EDT 2011
+FEK900I ISPF IVP: executed by uid=1(IBMUSER) gid=0(SYS1)
+FEK900I ISPF IVP: using /etc/rdz/rsed.envvars
+FEK900I ISPF IVP: current address space size limit is 2147483647
(2048.0 MB)
+FEK900I ISPF IVP: maximum address space size limit is 2147483647
(2048.0 MB)
+FEK900I ISPF IVP: -----------------------------------------------------
--------
+FEK900I ISPF IVP: /etc/rdz/ISPF.conf content:
+FEK900I ISPF IVP: -----------------------------------------------------
--------
+FEK900I ISPF IVP: ispllib=ISP.SISPLOAD
+FEK900I ISPF IVP: ispmlib=ISP.SISPMENU
+FEK900I ISPF IVP: isptlib=ISP.SISPTENU
+FEK900I ISPF IVP: ispplib=ISP.SISPPENU
+FEK900I ISPF IVP: ispslib=ISP.SISPSLIB
+FEK900I ISPF IVP: sysproc=ISP.SISPCLIB,FEK.SFEKPROC
+FEK900I ISPF IVP: -----------------------------------------------------
--------
+FEK900I ISPF IVP: Host install verification for RSE
+FEK900I ISPF IVP: Review IVP log messages from HOST below :
+FEK900I ISPF IVP: -----------------------------------------------------
--------
+FEK900I ISPF IVP: Service level 22Feb2011
+FEK900I ISPF IVP: RSE connection and base TSO/ISPF session initializati
on check only
+FEK900I ISPF IVP: *** CHECK : ENVIRONMENT VARIABLES - key variables
displayed below :
+FEK900I ISPF IVP: Server PATH         = .:/usr/lpp/java/J6.0/bin:/usr/l
pp/rdz/bin:/usr/lpp/ispf/bin:/bin:/usr/sbin
+FEK900I ISPF IVP: STEPLIB             = NONE
+FEK900I ISPF IVP: Temporary directory = /tmp
+FEK900I ISPF IVP: CGI_ISPHOME  = /usr/lpp/ispf
+FEK900I ISPF IVP: CGI_ISPCONF  = /etc/rdz
+FEK900I ISPF IVP: CGI_ISPWORK  = /var/rdz
+FEK900I ISPF IVP: -----------------------------------------------------
--------
+FEK900I ISPF IVP: *** CHECK : USS MODULES
+FEK900I ISPF IVP: Checking ISPF Directory : /usr/lpp/ispf
+FEK900I ISPF IVP: Checking modules in /usr/lpp/ispf/bin directory
+FEK900I ISPF IVP: Checking for ISPF configuration file ISPF.conf
+FEK900I ISPF IVP: RC=0
+FEK900I ISPF IVP: MSG: SUCCESSFUL
+FEK900I ISPF IVP: -----------------------------------------------------
--------
+FEK900I ISPF IVP: *** CHECK : TSO/ISPF INITIALIZATION
+FEK900I ISPF IVP: ( TSO/ISPF session will be initialized )
+FEK900I ISPF IVP: RC=0
+FEK900I ISPF IVP: MSG: SUCCESSFUL
+FEK900I ISPF IVP: -----------------------------------------------------
--------
+FEK900I ISPF IVP: *** CHECK: Shutting down TSO/ISPF IVP session
+FEK900I ISPF IVP: RC=0
+FEK900I ISPF IVP: MSG: SUCCESSFUL
+FEK900I ISPF IVP: -----------------------------------------------------
--------
+FEK900I ISPF IVP: Host installation verification completed successfully
+FEK900I ISPF IVP: -----------------------------------------------------
--------
+FEK901I ISPF IVP  Exit code = 0
```

Note:

- The function is similar to what the `fekfivpi` IVP (Installation Verification Program) does.
- RSE daemon will generate a PassTicket which is used as password for the IVP, so there will be no WTOR (Write To Operator with Reply) requesting a password.

IVP PASSTICKET,userid

Test the reusability of a PassTicket generated for user ID `userid`. Results are shown with one ore more FEK900I console messages. The return code is shown with console message FEK901I.

```
+FEK900I PASSTICKET IVP: the default applid=FEKAPPL
+FEK900I PASSTICKET IVP: Success, PassTicket IVP finished normally
+FEK901I PASSTICKET IVP  Exit code = 0
```

Note:

- When using RACF as security product, reusable PassTickets require the “`NO REPLAY PROTECTION`” keyword in the security definitions.
- There is no equivalent IVP (Installation Verification Program) for this test. Starting RSE daemon with the IVP=IVP argument will invoke a PassTicket IVP that tests PassTicket generation, but it cannot test PassTicket reusability.
- RSE daemon will generate a PassTicket which is used as password for the IVP, so there will be no WTOR (Write To Operator with Reply) requesting a password.

SWITCH

Switch to a new audit log file.

Lock daemon

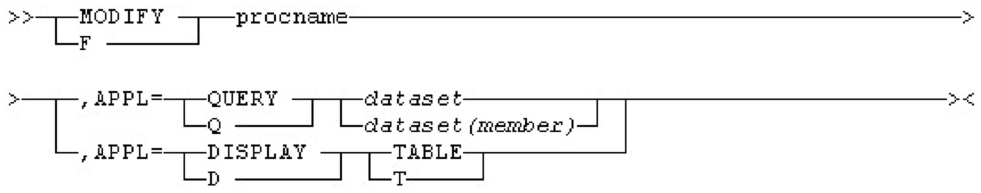


Figure 40. MODIFY LOCKD operator command

procname

The name of the member in a procedure library that was used to start the server. The default name used during the host configuration is `LOCKD`.

QUERY dataset[(member)]

Query the lock status of the listed data set or member. The server will reply with one of the following messages:

```
BPXM023I (stclock) dataset[(member)] NOT LOCKED
BPXM023I (stclock) dataset[(member)] LOCKED BY userid
```

Note:

- The server will also report locks held by other products, such as ISPF.
- Locks held by Developer for System z clients who were unable to register with the lock daemon will result in the thread pool server address space (RSEDx) being reported as lock owner.

Console message FEK513W is generated when RSE server is unable to register the client with the lock daemon. The ASID and TCB values mentioned in this message can be compared against the output of the **D GRS,RES=(*,dataset[(member)])** operator command in order to find the actual user holding the lock.

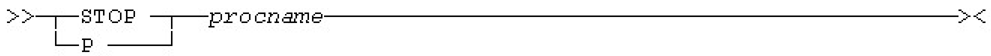
DISPLAY TABLE

Display the lock daemon mapping table in a single BPXM023I message. The lock daemon uses this mapping table to determine which Developer for System z user holds a certain data set lock (GRS reports only the ASID/TCB pair).

```
PID------- ASID TCB----- USERID--
      350 001A 00123ABC IBMUSER
```

Stop (P)

Use the **STOP** command to stop an active task. The abbreviated version of the command is the letter P.



```
>>─┬─STOP ─┬──procname──────────────────────────────────────────><
   └─P ────┘
```

Figure 41. STOP operator command

procname

The name of the member in a procedure library that was used to start the server. The default names used during the host configuration are JMON, RSED, and LOCKD for JES Job Monitor, RSE daemon, and the lock daemon, respectively.

Console messages

JES Job Monitor

JES Job Monitor does not have product-specific console messages. The server relies on z/OS and JES to generate console messages for actions done by Developer for System z clients.

RSE daemon, RSE thread pool server, and lock daemon

Table 27 lists the product-specific console messages generated by RSE daemon, RSE thread pool server, and the lock daemon.

Table 27. RSE console messages

Message ID	Message text
FEK001I	RseDaemon being initialized in {0} bit mode
FEK002I	RseDaemon started. (port={0})
FEK003I	Stop command being processed
FEK004I	RseDaemon: Max Heap Size={0}MB and private AS Size={1}MB
FEK005I	Server process started. (processId={0})
FEK009I	RseDaemon is waiting for the server process to start.
FEK010I	(rsed.envvars location = {0})
FEK011I	(log directory = {0})

Table 27. RSE console messages (continued)

Message ID	Message text
FEK012I	(RSE home directory = {0})
FEK100E	Daemon port/timeout value must be digits
FEK101E	JRE {0} or higher required
FEK102E	Invalid arguments received: {0}
FEK103E	Almost Disk-Full in {0}
FEK104E	Maximum number of processes has been reached
FEK105E	Error in sending audit data (rc={0})
FEK110E	socket() failed. reason=({0})
FEK111E	setsockopt() failed. reason=({0})
FEK112E	bind() failed. reason=({0})
FEK113E	listen() failed. reason=({0})
FEK114E	accept() failed. reason=({0})
FEK115E	write() failed. reason=({0})
FEK116E	pipe() failed. reason=({0})
FEK117E	socketpair() failed. reason=({0})
FEK118E	select() failed. reason=({0})
FEK119E	_console() failed. reason=({0})
FEK130E	gsk_environment_open() failed. reason=({0})
FEK131E	gsk_attribute_set_enum(GSK_PROTOCOL_SSLV2) failed. reason=({0})
FEK132E	gsk_attribute_set_enum(GSK_PROTOCOL_SSLV3) failed. reason=({0})
FEK133E	gsk_attribute_set_enum(GSK_PROTOCOL_TLSV1) failed. reason=({0})
FEK134E	gsk_attribute_set_buffer(GSK_KEYRING_FILE) failed. reason=({0})
FEK135E	gsk_attribute_set_buffer(GSK_KEYRING_PW) failed. reason=({0})
FEK136E	gsk_environment_init() failed. reason=({0})
FEK137E	gsk_secure_socket_open() failed. reason=({0})
FEK138E	gsk_attribute_set_numeric_value(GSK_FD) failed. reason=({0})
FEK139E	gsk_attribute_set_buffer(GSK_KEYRING_LABEL) failed. reason=({0})
FEK140E	gsk_attribute_set_enum(GSK_SESSION_TYPE) failed. reason=({0})
FEK141E	gsk_attribute_set_callback(GSK_IO_CALLBACK) failed. reason=({0})
FEK142E	gsk_secure_socket_init() failed. reason=({0})
FEK143E	gsk_attribute_set_enum(GSK_CLIENT_AUTH_TYPE) failed. reason=({0})
FEK144E	gsk_get_cert_info failed. reason=({0})
FEK145E	gsk_secure_socket_read() failed. reason=({0})
FEK146E	gsk_secure_socket_write() failed. reason=({0})
FEK150E	RseDaemon abnormally terminated; {0}
FEK201I	{0} Command has been processed
FEK202E	Invalid Command Entered
FEK203E	Invalid Display Command: Display Process|Client
FEK204E	Invalid Cancel Command: Cancel ID=|User=

Table 27. RSE console messages (continued)

Message ID	Message text
FEK205E	Command was not processed owing to consecutive SWITCHs
FEK206E	Audit Log facility is not active
FEK207I	No Client to be displayed
FEK208I	{0} canceled
FEK209I	No Process to be displayed
FEK210I	{0} canceled owing to duplicate logon
FEK211W	User, {0}, not logged on
FEK501I	Lock daemon started, port={0}, cleanup interval={1}, log level={2}
FEK502I	Lock daemon terminating
FEK510E	Lock daemon, missing port
FEK511E	Lock daemon, wrong port, port={0}
FEK512E	Lock daemon, socket error, port={0}
FEK513W	Lock daemon, registration failed, ASID={0}, TCB={1}, USER={2}
FEK514W	Lock daemon, wrong log level, log level={0}
FEK900I	{0} IVP: {1}
FEK901I	{0} IVP Exit code = {1}
FEK910I	{0} EXIT : {1}
FEK999E	{0}
BPXM023I	(stclock) dataset[(member)] NOT LOCKED
BPXM023I	(stclock) dataset[(member)] LOCKED BY userid
BPXM023I	(stclock) command, WRONG COMMAND
BPXM023I	(stclock) command, MISSING ARGUMENT
BPXM023I	(stclock) argument, WRONG ARGUMENT

How to read a syntax diagram

The syntax diagram shows you how to specify a command so that the operating system can correctly interpret what you type. Read the syntax diagram from left to right and from top to bottom, following the horizontal line (the main path).

Symbols

The following symbols are used in syntax diagrams:

Symbol	Description
>>	Marks the beginning of the syntax diagram.
>	Indicates that the syntax diagram is continued.
|	Marks the beginning and end of a fragment or part of the syntax diagram.
><	Marks the end of the syntax diagram.

Operands

The following types of operands are used in syntax diagrams:

- Required operands are displayed on the main path line:

```
>>──REQUIRED_OPERAND──><
```

- Optional operands are displayed below the main path line:

```
>>─┬──────────────────┬─><
   └─OPTIONAL_OPERAND─┘
```

- Default operands are displayed above the main path line:

```
   ┌─DEFAULT_OPERAND─┐
>>─┴─────────────────┴─><
```

Operands are classified as keywords or variables:

- Keywords are constants that must be provided. If the keyword appears in the syntax diagram in both uppercase and lowercase, the uppercase portion is the abbreviation for the keyword (for example, KEYword). Keywords are not case-sensitive. You can code them in uppercase or lowercase.
- Variables are italicized, appear in lowercase letters, and represent names or values you supply. For example, a data set name is a variable. Variables can be case sensitive.

Syntax example

In the following example, the `USER` command is a keyword. The required variable parameter is `user_id`, and the optional variable parameter is `password`. Replace the variable parameters with your own values:

```
>>──USER──user_id─┬──────────┬──────────────────────────────────><
                  └─password─┘
```

Nonalphanumeric characters and blank spaces

If a diagram shows a character that is not alphanumeric (such as parentheses, periods, commas, equal signs, and blank spaces), you must code the character as part of the syntax. In this example, you must code `OPERAND=(001 0.001)`:

```
>>──OPERAND──=──(──001── ──0.001──)──────────────────────><
```

Selecting more than one operand

An arrow returning to the left in a group of operands means that more than one can be selected, or that a single one can be repeated:

```
>>───┬──────────────────────┬─────────────────────────────────><
     ├─REPEATABLE_OPERAND_1─┤
     ├─REPEATABLE_OPERAND_2─┤
     └─<────────────────────┘
```

Longer than one line

If a diagram is longer than one line, the first line ends with a single arrowhead and the second line begins with a single arrowhead:

```
>>──| The first line of a syntax diagram that is longer than one line |──>
>──| The continuation of the subcommands, parameters, or both |─────────><
```

Syntax fragments

Some diagrams might contain syntax fragments, which serve to break up diagrams that are too long, too complex, or too repetitious. Syntax fragment names are in mixed case and are shown in the diagram and in the heading of the fragment. The fragment is placed below the main diagram:

```
 >>──┤ Syntax fragment ├──────────────────────────────────────><

Syntax fragment:
├──1ST_OPERAND──,──2ND_OPERAND──,──3RD_OPERAND──┤
```

Appendix. Host Configuration Reference

This section summarizes the information in *Rational Developer for System z Host Configuration Reference* (SC14-7290). Refer to that publication for more details.

Understanding Developer for System z

The Developer for System z host consists of several components that interact to give the client access to the host services and data. Understanding the design of these components can help you make the correct configuration decisions.

Security considerations

Developer for System z provides mainframe access to users on a non-mainframe workstation. Validating connection requests, providing secure communication between the host and the workstation, and authorizing and auditing activity are therefore important aspects of the product configuration.

TCP/IP considerations

Developer for System z uses TCP/IP to provide mainframe access to users on a non-mainframe workstation. It also uses TCP/IP for communication between various components and other products.

WLM considerations

Unlike traditional z/OS applications, Developer for System z is not a monolithic application that can be identified easily to Workload Manager (WLM). Developer for System z consists of several components that interact to give the client access to the host services and data. Some of these services are active in different address spaces, resulting in different WLM classifications.

Tuning considerations

RSE (Remote Systems Explorer) is the core of Developer for System z. To manage the connections and workloads from the clients, RSE is composed of a daemon address space, which controls thread pooling address spaces. The daemon acts as a focal point for connection and management purposes, while the thread pools process the client workloads.

This makes RSE a prime target for tuning the Developer for System z setup. However, maintaining hundreds of users, each using 16 or more threads, a certain amount of storage, and possibly one or more address spaces requires proper configuration of both Developer for System z and z/OS.

Performance considerations

z/OS is a highly customizable operating system, and (sometimes small) system changes can have a huge impact on the overall performance. This chapter highlights some of the changes that can be made to improve the performance of Developer for System z.

Push-to-client considerations

Push-to-client, or host-based client control, supports central management of the following:

- Client configuration files
- Client product version
- Project definitions

CICSTS considerations

This chapter contains information useful for a CICS Transaction Server administrator.

User exit considerations

This chapter assists you with enhancing Developer for System z by writing exit routines.

Customizing the TSO environment

This chapter assists you with mimicking a TSO logon procedure by adding DD statements and data sets to the TSO environment in Developer for System z.

Running multiple instances

There are times that you want multiple instances of Developer for System z active on the same system, for example, when testing an upgrade. However, some resources such as TCP/IP ports cannot be shared, so the defaults are not always applicable. Use the information in this chapter to plan the coexistence of the different instances of Developer for System z, after which you can use this configuration guide to customize them.

Troubleshooting configuration problems

This chapter is provided to assist you with some common problems that you might encounter during your configuration of Developer for System z, and has the following sections:

- Log and setup analysis using FEKLOGS
- Log files
- Dump files
- Tracing
- z/OS UNIX permission bits
- Reserved TCP/IP ports
- Address Space size
- APPC transaction and TSO Commands service
- Miscellaneous information

Setting up SSL and X.509 authentication

This appendix is provided to assist you with some common problems that you might encounter when setting up Secure Socket Layer (SSL), or during checking or modifying an existing setup. This appendix also provides a sample setup to support users authenticating themselves with an X.509 certificate.

Setting up TCP/IP

This appendix is provided to assist you with some common problems that you might encounter when setting up TCP/IP, or during checking or modifying an existing setup.

Bibliography

Referenced publications

The following publications are referenced in this document:

Table 28. Referenced publications

Publication title	Order number	Reference	Reference Web site
Program Directory for IBM Rational Developer for System z	GI11-8298	Developer for System z	http://www.ibm.com/software/rational/products/developer/systemz/library/index.html
Rational Developer for System z Prerequisites	SC23-7659	Developer for System z	http://www.ibm.com/software/rational/products/developer/systemz/library/index.html
Rational Developer for System z Host Configuration Quick Start	GI11-9201	Developer for System z	http://www.ibm.com/software/rational/products/developer/systemz/library/index.html
Rational Developer for System z Host Configuration Guide	SC23-7658	Developer for System z	http://www.ibm.com/software/rational/products/developer/systemz/library/index.html
Rational Developer for System z Host Configuration Reference	SC14-7290	Developer for System z	http://www.ibm.com/software/rational/products/developer/systemz/library/index.html
Rational Developer for System z Host Configuration Utility Guide	SC14-7282	Developer for System z	http://www.ibm.com/software/rational/products/developer/systemz/library/index.html
Rational Developer for System z Messages and Codes	SC14-7497	Developer for System z	http://www.ibm.com/software/rational/products/developer/systemz/library/index.html
Rational Developer for System z Answers to common host configuration and maintenance issues	SC14-7373	Developer for System z	http://www.ibm.com/software/rational/products/developer/systemz/library/index.html
Rational Developer for System z Common Access Repository Manager Developer's Guide	SC23-7660	Developer for System z	http://www.ibm.com/software/rational/products/developer/systemz/library/index.html
Rational Developer for System z Prerequisites	SC23-7659	Developer for System z	http://www.ibm.com/software/rational/products/developer/systemz/library/index.html
Rational Developer for System z Host Configuration Quick Start	GI11-9201	Developer for System z	http://www.ibm.com/software/rational/products/developer/systemz/library/index.html
SCLM Developer Toolkit Administrator's Guide	SC23-9801	Developer for System z	http://www.ibm.com/software/rational/products/developer/systemz/library/index.html
Using APPC to provide TSO command services	SC14-7291	White paper	http://www-306.ibm.com/software/awdtools/rdz/library/
Using ISPF Client Gateway to provide CARMA services	SC14-7292	White paper	http://www-306.ibm.com/software/awdtools/rdz/library/
Communications Server IP Configuration Guide	SC31-8775	z/OS 1.11	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/

Table 28. Referenced publications (continued)

Publication title	Order number	Reference	Reference Web site
Communications Server IP Configuration Reference	SC31-8776	z/OS 1.11	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
Communications Server IP Diagnosis Guide	GC31-8782	z/OS 1.11	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
Communications Server IP System Administrator's Commands	SC31-8781	z/OS 1.11	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
Communications Server SNA Network Implementation Guide	SC31-8777	z/OS 1.11	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
Communications Server SNA Operations	SC31-8779	z/OS 1.11	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
Cryptographic Services System SSL Programming	SC24-5901	z/OS 1.11	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
DFSMS Macro Instructions for Data Sets	SC26-7408	z/OS 1.11	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
DFSMS Using data sets	SC26-7410	z/OS 1.11	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
Language Environment Customization	SA22-7564	z/OS 1.11	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
Language Environment Debugging Guide	GA22-7560	z/OS 1.11	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
MVS Initialization and Tuning Guide	SA22-7591	z/OS 1.11	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
MVS Initialization and Tuning Reference	SA22-7592	z/OS 1.11	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
MVS JCL Reference	SA22-7597	z/OS 1.11	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
MVS Planning APPC/MVS Management	SA22-7599	z/OS 1.11	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
MVS Planning Workload Management	SA22-7602	z/OS 1.11	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
MVS System Commands	SA22-7627	z/OS 1.11	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
Security Server RACF Command Language Reference	SA22-7687	z/OS 1.11	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
Security Server RACF Security Administrator's Guide	SA22-7683	z/OS 1.11	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
TSO/E Customization	SA22-7783	z/OS 1.11	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
TSO/E REXX Reference	SA22-7790	z/OS 1.11	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
UNIX System Services Command Reference	SA22-7802	z/OS 1.11	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/

Table 28. Referenced publications (continued)

Publication title	Order number	Reference	Reference Web site
UNIX System Services Planning	GA22-7800	z/OS 1.11	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
UNIX System Services User's Guide	SA22-7801	z/OS 1.11	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
Using REXX and z/OS UNIX System Services	SA22-7806	z/OS 1.11	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
Java™ Diagnostic Guide	SC34-6650	Java 5.0	http://www.ibm.com/developerworks/java/jdk/diagnosis/
Java SDK and Runtime Environment User Guide	/	Java 5.0	http://www-03.ibm.com/servers/eserver/zseries/software/java/
Resource Definition Guide	SC34-6430	CICSTS 3.1	http://www-03.ibm.com/systems/z/os/zos/bkserv/zapplsbooks.html
Resource Definition Guide	SC34-6815	CICSTS 3.2	http://www-03.ibm.com/systems/z/os/zos/bkserv/zapplsbooks.html
Resource Definition Guide	SC34-7000	CICSTS 4.1	https://publib.boulder.ibm.com/infocenter/cicsts/v4r1/index.jsp?topic=/com.ibm.cics.ts.home.doc/library/library_html.html
Resource Definition Guide	SC34-7181	CICSTS 4.2	https://publib.boulder.ibm.com/infocenter/cicsts/v4r2/index.jsp?topic=/com.ibm.cics.ts.home.doc/library/library_html.html
RACF Security Guide	SC34-6454	CICSTS 3.1	http://www-03.ibm.com/systems/z/os/zos/bkserv/zapplsbooks.html
RACF Security Guide	SC34-6835	CICSTS 3.2	http://www-03.ibm.com/systems/z/os/zos/bkserv/zapplsbooks.html
RACF Security Guide	SC34-7003	CICSTS 4.1	https://publib.boulder.ibm.com/infocenter/cicsts/v4r1/index.jsp?topic=/com.ibm.cics.ts.home.doc/library/library_html.html
RACF Security Guide	SC34-7179	CICSTS 4.2	https://publib.boulder.ibm.com/infocenter/cicsts/v4r2/index.jsp?topic=/com.ibm.cics.ts.home.doc/library/library_html.html
Language Reference	SC27-1408	Enterprise COBOL for z/OS	http://www-03.ibm.com/systems/z/os/zos/bkserv/zapplsbooks.html

The following Web sites are referenced in this document:

Table 29. Referenced Web sites

Description	Reference Web site
Developer for System z Information Center	http://publib.boulder.ibm.com/infocenter/ratdevz/v8r5/index.jsp
Developer for System z Support	http://www-306.ibm.com/software/awdtools/rdz/support/
Developer for System z Library	http://www-306.ibm.com/software/awdtools/rdz/library/
Developer for System z home page	http://www-306.ibm.com/software/awdtools/rdz/
Developer for System z Recommended service	http://www-01.ibm.com/support/docview.wss?rs=2294&context=SS2QJ2&uid=swg27006335
Developer for System z enhancement request	https://www.ibm.com/developerworks/support/rational/rfe/
z/OS internet library	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/

Table 29. Referenced Web sites (continued)

Description	Reference Web site
CICSTS Information Center	https://publib.boulder.ibm.com/infocenter/cicsts/v4r1/index.jsp
IBM Tivoli® Directory Server	http://www-01.ibm.com/software/tivoli/products/directory-server/
Problem Determination Tools Plug-ins	http://www-01.ibm.com/software/awdtools/deployment/pdtplugins/
Download Apache Ant	http://ant.apache.org/
Java keytool documentation	http://java.sun.com/j2se/1.5.0/docs/tooldocs/solaris/keytool.html
CA support home page	https://support.ca.com/

Informational publications

The following publications can be helpful in understanding setup issues for requisite host components:

Table 30. Informational publications

Publication title	Order number	Reference	Reference Web site
ABCs of z/OS System Programming Volume 9 (z/OS UNIX)	SG24-6989	Redbook	http://www.redbooks.ibm.com/
System Programmer's Guide to: Workload Manager	SG24-6472	Redbook	http://www.redbooks.ibm.com/
TCPIP Implementation Volume 1: Base Functions, Connectivity, and Routing	SG24-7532	Redbook	http://www.redbooks.ibm.com/
TCPIP Implementation Volume 3: High Availability, Scalability, and Performance	SG24-7534	Redbook	http://www.redbooks.ibm.com/
TCP/IP Implementation Volume 4: Security and Policy-Based Networking	SG24-7535	Redbook	http://www.redbooks.ibm.com/
Tivoli Directory Server for z/OS	SG24-7849	Redbook	http://www.redbooks.ibm.com/

Glossary

Action ID
A numeric identifier for an action between 0 and 999

Application Server

1. A program that handles all application operations between browser-based computers and an organization's back-end business applications or databases. There is a special class of Java-based appservers that conform to the J2EE standard. J2EE code can be easily ported between these appservers. They can support JSPs and servlets for dynamic Web content and EJBs for transactions and database access.
2. The target of a request from a remote application. In the DB2 environment, the application server function is provided by the distributed data facility and is used to access DB2 data from remote applications.
3. A server program in a distributed network that provides the execution environment for an application program.
4. The target of a request from an application requester. The database management system (DBMS) at the application server site provides the requested data.
5. Software that handles communication with the client requesting an asset and queries of the Content Manager.

Bidirectional (bi-di)
Pertaining to scripts such as Arabic and Hebrew that generally run from right to left, except for numbers, which run from left to right. This definition is from the Localization Industry Standards Association (LISA) Glossary.

Bidirectional Attribute
Text type, text orientation, numeric swapping, and symmetric swapping.

Build Request
A request from the client to perform a build transaction.

Build Transaction
A job started on MVS to perform builds after a build request has been received from the client.

Compile

1. In Integrated Language Environment (ILE) languages, to translate source statements into modules that then can be bound into programs or service programs.
2. To translate all or part of a program expressed in a high-level language into a computer program expressed in an intermediate language, an assembly language, or a machine language.

Container

1. In CoOperative Development Environment/400, a system object that contains and organizes source files. An i5/OS® library or an MVS-partitioned data set are examples of a container.
2. In J2EE, an entity that provides life-cycle management, security, deployment, and runtime services to components. (Sun) Each type of container (EJB, Web, JSP, servlet, applet, and application client) also provides component-specific services
3. In Backup Recovery and Media Services, the physical object used to store and move media such as a box, a case, or a rack.
4. In a virtual tape server (VTS), a receptacle in which one or more exported logical volumes (LVOLs) can be stored. A stacked volume containing one or more LVOLs and residing outside a VTS library is considered to be the container for those volumes.
5. A physical storage location of the data. For example, a file, directory, or device.

6. A column or row that is used to arrange the layout of a portlet or other container on a page.
7. An element of the user interface that holds objects. In the folder manager, an object that can contain other folders or documents.

Database
A collection of interrelated or independent data items that are stored together to serve one or more applications.

Data Definition View
Contains a local representation of databases and their objects and provides features to manipulate these objects and export them to a remote database

Data Set
The major unit of data storage and retrieval, consisting of a collection of data in one of several prescribed arrangements and described by control information to which the system has access.

Debug
To detect, diagnose, and eliminate errors in programs.

Debugging Session
The debugging activities that occur between the time that a developer starts a debugger and the time that the developer exits from it.

Error Buffer
A portion of storage used to hold error output information temporarily.

Gateway

1. A middleware component that bridges Internet and intranet environments during Web service invocations.
2. Software that provides services between the endpoints and the rest of the Tivoli environment.
3. A component of a Voice over Internet Protocol that provides a bridge between VoIP and circuit-switched environments.
4. A device or program used to connect networks or systems with different network architectures. The systems might have different characteristics, such as different communication protocols, different network architecture, or different security policies, in which case the gateway performs a translation role as well as a connection role.

Interactive System Productivity Facility (ISPF)
An IBM licensed program that serves as a full-screen editor and dialog manager. Used for writing application programs, it provides a means of generating standard screen panels and interactive dialogs between the application programmer and terminal user. ISPF consists of four major components: DM, PDF, SCLM, and C/S. The DM component is the Dialog Manager, which provides services to dialogs and end-users. The PDF component is the Program Development Facility, which provides services to assist the dialog or application developer. The SCLM component is the Software Configuration Library Manager, which provides services to application developers to manage their application development libraries. The C/S component is the Client/Server, which allows you to run ISPF on programmable workstation, to display the panels using the display function of your workstation operating system, and to integrate workstation tools and data with host tools and data.

Interpreter
A program that translates and runs each instruction of a high-level programming language before it translates and runs the next instruction.

Isomorphic
Each composed element (in other words, an element containing other elements) of the XML instance document starting from

the root has one and only one corresponding COBOL group item whose nesting depth is identical to the nesting depth of its XML equivalent. Each non-composed element (in other words, an element that does not contain other elements) in the XML instance document starting from the top has one and only one corresponding COBOL elementary item whose nesting depth is identical to the nesting level of its XML equivalent and whose memory address at runtime can be uniquely identified.

Linkage Section
The section in the data division of an activated unit (a called program or an invoked method) that describes data items available from the activating unit (a program or a method). These data items can be referred to by both the activated unit and the activating unit.

Load Library
A library containing load modules.

Lock Action
Locks a member.

Navigator View
Provides a hierarchical view of the resources in the Workbench.

Non-Isomorphic
A simple mapping of COBOL items and XML elements belonging to XML documents and COBOL groups that are not identical in shape (non-isomorphic). Non-isomorphic mapping can also be created between non-isomorphic elements of isomorphic structures.

Output Console View
Displays the output of a process and allows you to provide keyboard input to a process.

Output View
Displays messages, parameters, and results that are related to the objects that you work with

Perspective
A group of views that show various aspects of the resources in the workbench. The workbench user can switch perspectives, depending on the task at hand, and customize the layout of views and editors within the perspective.

RAM Repository Access Manager

Remote File System
A file system residing on a separate server or operating system.

Remote System
Any other system in the network with which your system can communicate.

Remote Systems Perspective
Provides an interface for managing remote systems using conventions that are similar to ISPF.

Repository

1. A storage area for data. Every repository has a name and an associated business item type. By default, the name will be the same as the name of the business item. For example, a repository for invoices will be called Invoices. There are two types of information repositories: local (specific to the process) and global (reusable).
2. A VSAM data set on which the states of BTS processes are stored. When a process is not executing under the control of BTS, its state (and the states of its constituent activities) are preserved by being written to a repository data set. The states of all processes of a particular process-type (and of their activity instances) are

stored on the same repository data set. Records for multiple process-types can be written to the same repository.

3. A persistent storage area for source code and other application resources. In a team programming environment, a shared repository enables multiuser access to application resources.
4. A collection of information about the queue managers that are members of a cluster. This information includes queue manager names, their locations, their channels, what queues they host, and so on.

Repository Instance
A project or component that exists in an SCM.

Repositories View
Displays the CVS repository locations that have been added to your Workbench.

Response File

1. A file that contains a set of predefined answers to questions asked by a program and that is used instead of entering those values one at a time.
2. An ASCII file that can be customized with the setup and configuration data that automates an installation. The setup and configuration data would have to be entered during an interactive install, but with a response file, the installation can proceed without any intervention.

Servers View
Displays a list of all your servers and the configurations that are associated with them.

Shell A software interface between users and the operating system that interprets commands and user interactions and communicates them to the operating system. A computer might have several layers of shells for various levels of user interaction.

Shell Name
The name of the shell interface.

Shell Script
A file containing commands that can be interpreted by the shell. The user types the name of the script file at the shell command prompt to make the shell execute the script commands.

Sidedeck
A library that publishes the functions of a DLL program. The entry names and module names are stored in the library after the source code is compiled.

Silent Installation
An installation that does not send messages to the console but instead stores messages and errors in log files. Also, a silent installation can use response files for data input.

Silent Uninstallation
An uninstallation process that does not send messages to the console but instead stores messages and errors in log files after the uninstall command has been invoked.

Task List
A list of procedures that can be executed by a single flow of control.

URL Uniform Resource Locator

Documentation notices for IBM Rational Developer for System z

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

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

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

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

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
1623-14, Shimotsuruma, Yamato-shi
Kanagawa 242-8502 Japan

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

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

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

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

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:

Intellectual Property Dept. for Rational Software
IBM Corporation
5 Technology Park Drive
Westford, MA 01886
U.S.A.

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 IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

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

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

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 the names and addresses used by an actual business enterprise is entirely coincidental.

Copyright license

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

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

© (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. 2009, 2012.

If you are viewing this information in softcopy, the photographs and color illustrations may not appear.

Trademark acknowledgments

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

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.

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

Windows is a trademark of Microsoft Corporation in the United States, other countries, or both.

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

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

Other product and service names might be trademarks of IBM or other companies.

Copyright license

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

Trademark acknowledgments

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

CA Endevor is a registered trademark of CA Technologies.

Rational are trademarks of International Business Machines Corporation and Rational Software Corporation, in the United States, other countries, or both.

Intel and Pentium are trademarks of Intel Corporation in the United States, or other countries, or both.

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

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries.

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

Index

Readers' Comments — We'd Like to Hear from You

IBM Rational Developer for System z
Version 8.5
Configuration Guide

Publication No. SC23-7658-07

We appreciate your comments about this publication. Please comment on specific errors or omissions, accuracy, organization, subject matter, or completeness of this book. The comments you send should pertain to only the information in this manual or product and the way in which the information is presented.

For technical questions and information about products and prices, please contact your IBM branch office, your IBM business partner, or your authorized remarketer.

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you. IBM or any other organizations will only use the personal information that you supply to contact you about the issues that you state on this form.

Comments:

Thank you for your support.

Send your comments to the address on the reverse side of this form.

If you would like a response from IBM, please fill in the following information:

Name

Address

Company or Organization

Phone No.

Email address

Readers' Comments — We'd Like to Hear from You

SC23-7658-07

Cut or Fold
Along Line

Fold and Tape **Please do not staple** Fold and Tape

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST-CLASS MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

IBM
Corporation
Building 501
P.O Box 12195
Research Triangle Park, NC
USA 27709-2195

Fold and Tape **Please do not staple** Fold and Tape

Cut or Fold
Along Line

Printed in USA

SC23-7658-07