IBM Tivoli OMEGAMON XE and Tivoli Management Services
on z/OS
Version 6 Release 3

Common Planning and Configuration
Guide

IBM
IBM Tivoli OMEGAMON XE and Tivoli Management Services
on z/OS
Version 6 Release 3

Common Planning and Configuration Guide
Enabling SNMP V3 passwords for autonomous agents ........................................... 138
Complete configuration of the OMEGAMON enhanced 3270 user interface .......... 139
Create a CUASITE profile ......................................................... 139
Enable security ................................................................. 140
  Define an SAF general resource class ................................ 141
  Define LOGON profiles to control access to the interface .... 142
  Define Take Action profiles to control access to data actions .... 143
  Define Query profiles to control access to data sources ....... 144
  Permit access to profiles .................................................. 144
  Using Demo mode ............................................................ 144
  Create a UNIX ID for the address space ................................ 145
(Optional) Enable maintenance of the historical data store ....................... 145
(Optional) Enable historical data collection .......................................... 146
Run the ITMSUPER Tools (optional) ........................................ 146

Part 3. Replicating configured runtime environments .................................... 147

Chapter 7. Using the PARMGEN method to replicate a configured runtime environment ........................................... 149
PARMGEN transport scenarios ................................................... 149
Clone an existing runtime environment ............................................... 152
Clone an existing environment with system variables enabled to run on a different LPAR ............................................ 164

Chapter 8. Using the Configuration Tool batch mode to replicate a configured environment .................................... 179
Step 1. Create the batch mode job ............................................ 180
  Creating the batch mode job ............................................. 180
  Batch mode job submission options ................................... 181
  Batch mode configuration jobs ......................................... 182
  Batch mode job report ..................................................... 182
Step 2. Create and clone a batch mode parameter member ............................................. 183
Step 3. Adjust and validate the parameters ............................................. 186
Step 4. Generate sample transport JCL .................................... 188
Step 5. Create a new runtime environment ...................................... 190
  Scenario A: Defining a runtime environment on a z/OS image using shared DASD 191
  Scenario B: Transporting a runtime environment from one z/OS image to another 192
  Scenario C: Transporting runtime environment batch jobs from one z/OS image to another equipped with the Configuration Tool 193
  Scenario D: Transporting runtime environment batch mode parameters from one z/OS image to another ............................................. 195
Step 6. Complete the configuration of a new runtime environment ............... 196

Part 4. Appendixes ......................................................... 197

Appendix A. Enabling system variable support ........................................... 199
Using the PARMGEN Workflow user interface to set up system variables ....... 199
  Prerequisites ................................................................. 199
  Additional symbolics for OMEGAMON XE for IMS on z/OS sites .......... 200
  Enable system variable support ........................................ 200
  Specify user-defined symbols and variables ................................ 201
Using the Configuration Tool (ICAT) to set up system variables .................. 202
  Enable system variable support ........................................ 202
  Create the system variable parameter member ................................ 203
  Create the VTAM major node rename job ................................ 204
  Create one VTAM major node for all products in the runtime environment .... 204

Appendix B. Configuration Tool deployment scenarios .................................... 207
Scenario 1: Standard replication method ............................................. 207
Scenario 2: Using common RKANPARU and RKANCMDU libraries ............ 209
Scenario 3: Using batch mode to replicate a runtime environment to another LPAR .......... 211
Scenario 4: Using batch auto-submit to replicate a runtime environment to another LPAR .... 212
Scenario 5: Using batch mode to replicate a runtime environment on the same LPAR .......... 213
Scenario 6: Using interactive copy to replicate a runtime environment .......... 215
Scenario 7: Copying configuration values from one instance to another of the Configuration Tool ............................................. 215
Scenario 8: Configuring a remote monitoring server and monitoring agents in batch mode ............................................. 216
  Step 1. Create the batch mode job (CICATB) ................................ 217
  Step 2. Clone the runtime environment batch parameter member .......... 218
  Step 3. Review and update the parameter member for the new runtime environment ............................................. 221
  Step 4. Run the CICATB job to validate the changes ....................... 223
  Step 5. Create and (optionally) execute the batch jobs to create the runtime environment ............................................. 225
  Step 6. Customize and complete the configuration ................................ 229

Appendix C. Configuration Tool reference ............................................... 231
Configuration services and utilities ............................................... 231
Runtime environment utilities ............................................. 232
Batch mode utilities ............................................................ 236
  KCISETUP: Set up the environment for batch utilities ............... 236
  Invoke KCISETUP ............................................................ 236
  KCICFKKEY: Manage ISPF PF Keys .................................... 237
  KCICPGHP: Display help for configuration parameters .................. 238
Appendix D. PARMGEN Reference 243
KCIRPLBS online help macro 243
XF edit macro 243

Appendix E. Maintaining the persistent data store 245
Overview of the persistent data store on z/OS 245
Automatic maintenance process 246
The part of maintenance you control 246
Indicating data set backup to tape or to DASD 247
Names of exported data sets 247
Making archived data available 248
Data set naming conventions 248
Prerequisites 248
Finding background information 248
Connecting the data set to the monitoring server 249
Disconnecting the data set 249
Exporting and restoring persistent data 250
Exporting persistent data 250
Restoring exported data 251
Format of exported data records 251
Section 2 records 251
Dictionary header record 251
Table description record 252
Column description record 253
Section 1 record 253
Section 3 records 254
Extracting persistent data store data to flat files 255
Extracting persistent data store data to EBCDIC files 256
Format of extracted header and data file records 256
Components of the persistent data store 257
Operation of the persistent data store 258
Command interface 258
Maintenance commands 258
SWITCH command 258
BACKUP command 259
ADDFILE command 259
DELFILE command 260
EXTRACT command 260
INITIDS command 261
RECOVER command 261
RESUME command 261
Other useful commands 262
QUERY CONNECT command 262
QUERY DATASTORE command 262
COMMIT command 262

Appendix F. Product codes 265

Appendix G. Predefining and managing OMEGAMON started tasks 267
Workload Manager (WLM) settings for the OMEGAMON tasks 269

Documentation library 271
Shared OMEGAMON XE publications 271
IBM Tivoli Monitoring library 272
Documentation for the base agents 273
Other sources of documentation 273

Support information 275

Notices 277
Trademarks 278

Index 279
# Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Agent/server/client architecture</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>High-availability runtime environment</td>
<td>27</td>
</tr>
<tr>
<td>3</td>
<td>Full runtime environment on a single system</td>
<td>47</td>
</tr>
<tr>
<td>4</td>
<td>Full runtime environments on several systems</td>
<td>48</td>
</tr>
<tr>
<td>5</td>
<td>Base runtime environment</td>
<td>49</td>
</tr>
<tr>
<td>6</td>
<td>Sharing-with-base runtime environment</td>
<td>50</td>
</tr>
<tr>
<td>7</td>
<td>Sharing-with-full runtime environment</td>
<td>52</td>
</tr>
<tr>
<td>8</td>
<td>Sharing-with-SMP/E runtime environment</td>
<td>53</td>
</tr>
<tr>
<td>9</td>
<td>The PARMGEN Workflow main panel</td>
<td>64</td>
</tr>
<tr>
<td>10</td>
<td>The PARMGEN Workflow main panel</td>
<td>69</td>
</tr>
<tr>
<td>11</td>
<td>Set up the PARMGEN work environment for a runtime environment (1 of 3)</td>
<td>71</td>
</tr>
<tr>
<td>12</td>
<td>Set up the PARMGEN work environment for a runtime environment (2 of 3)</td>
<td>72</td>
</tr>
<tr>
<td>13</td>
<td>Set up the PARMGEN work environment for a runtime environment (3 of 3)</td>
<td>73</td>
</tr>
<tr>
<td>14</td>
<td>Display PARMGEN environment analysis</td>
<td>77</td>
</tr>
<tr>
<td>15</td>
<td>Exclude products from PARMGEN customization</td>
<td>78</td>
</tr>
<tr>
<td>16</td>
<td>Customize PARMGEN configuration profile members</td>
<td>86</td>
</tr>
<tr>
<td>17</td>
<td>Update the user prolog</td>
<td>88</td>
</tr>
<tr>
<td>18</td>
<td>Verify the list of products in the runtime environment</td>
<td>88</td>
</tr>
<tr>
<td>19</td>
<td>Update TCP/IP port values</td>
<td>89</td>
</tr>
<tr>
<td>20</td>
<td>Enable APF-authorization statements</td>
<td>90</td>
</tr>
<tr>
<td>21</td>
<td>Enable the self-describing agent feature</td>
<td>91</td>
</tr>
<tr>
<td>22</td>
<td>Configure an OMEGAMON enhanced 3270 user interface</td>
<td>92</td>
</tr>
<tr>
<td>23</td>
<td>Create the runtime environment members and jobs</td>
<td>97</td>
</tr>
<tr>
<td>24</td>
<td>Submit batch jobs to complete PARMGEN setup</td>
<td>98</td>
</tr>
<tr>
<td>25</td>
<td>Configuration Tool Main Menu</td>
<td>105</td>
</tr>
<tr>
<td>26</td>
<td>Set up Work Environment panel</td>
<td>106</td>
</tr>
<tr>
<td>27</td>
<td>Specify JCL Options panel</td>
<td>106</td>
</tr>
<tr>
<td>28</td>
<td>Set up Configuration Environment panel</td>
<td>108</td>
</tr>
<tr>
<td>29</td>
<td>Configure Products panel</td>
<td>110</td>
</tr>
<tr>
<td>30</td>
<td>Product Selection Menu</td>
<td>110</td>
</tr>
<tr>
<td>31</td>
<td>Runtime Environments (RTEs) panel</td>
<td>111</td>
</tr>
<tr>
<td>32</td>
<td>Add Runtime Environment panel for a base runtime environment</td>
<td>112</td>
</tr>
<tr>
<td>33</td>
<td>Add Runtime Environment (1 of 3) panel for a full or sharing runtime environment</td>
<td>113</td>
</tr>
<tr>
<td>34</td>
<td>Add Runtime Environment (2 of 3) panel for a full or sharing runtime environment</td>
<td>116</td>
</tr>
<tr>
<td>35</td>
<td>Add Runtime Environment (3 of 3) panel for a full or sharing runtime environment</td>
<td>119</td>
</tr>
<tr>
<td>36</td>
<td>Verify Configuration and Generate Parameter Map panel</td>
<td>122</td>
</tr>
<tr>
<td>37</td>
<td>Specify Nonstandard Parameters panel</td>
<td>125</td>
</tr>
<tr>
<td>38</td>
<td>Create/Clone Batch Mode Parameters panel</td>
<td>185</td>
</tr>
<tr>
<td>39</td>
<td>SYSTSPRT for successful scan</td>
<td>187</td>
</tr>
<tr>
<td>40</td>
<td>Pre-defined and user-defined symbolics</td>
<td>201</td>
</tr>
<tr>
<td>41</td>
<td>Batch scenario configuration: z/OS remote monitoring server reporting to z/OS hub</td>
<td>216</td>
</tr>
<tr>
<td>42</td>
<td>Runtime Environments (RTEs) panel</td>
<td>219</td>
</tr>
<tr>
<td>43</td>
<td>RTE Utility Menu panel</td>
<td>219</td>
</tr>
<tr>
<td>44</td>
<td>Create/Clone Parameter Deck Menu panel</td>
<td>219</td>
</tr>
<tr>
<td>45</td>
<td>Create/Clone Batch Mode Parameters panel</td>
<td>220</td>
</tr>
<tr>
<td>46</td>
<td>Batch parameter member for SYSB</td>
<td>221</td>
</tr>
<tr>
<td>47</td>
<td>SYSLOG/OPERLOG messages</td>
<td>229</td>
</tr>
<tr>
<td>48</td>
<td>Runtime Environments (RTEs) panel with new SYSB runtime environment</td>
<td>230</td>
</tr>
<tr>
<td>49</td>
<td>Configuration Services and Utilities panel</td>
<td>231</td>
</tr>
<tr>
<td>50</td>
<td>RTE Utility Menu panel</td>
<td>233</td>
</tr>
<tr>
<td>51</td>
<td>Runtime environments panel</td>
<td>239</td>
</tr>
<tr>
<td>52</td>
<td>README Table of Contents Display panel</td>
<td>240</td>
</tr>
<tr>
<td>53</td>
<td>Delete runtime environment panel</td>
<td>242</td>
</tr>
</tbody>
</table>
## Tables

1. Common components of OMEGAMON XE monitoring agents on z/OS .......................... 9
2. New runtime libraries and started task DDNAMEs .................................................. 13
3. Product packaging, Tivoli Management Services on z/OS ..................................... 22
4. Product packaging, OMEGAMON XE monitoring agents ..................................... 23
5. Types of libraries ...................................................................................................... 44
6. Types of runtime environments ............................................................................ 44
7. Runtime environment data set naming convention ............................................... 45
8. Validation of the runtime environment ................................................................. 135
9. Data validation on user interfaces .......................................................................... 137
10. CICATB job reports ............................................................................................. 183
11. Runtime environment transport methods ............................................................. 190
12. Actions required to update the runtime environment parameters ....................... 222
13. Batch parameter format ....................................................................................... 238
14. Actions required for configuration and maintenance tasks .................................. 240
15. Section 2 data record format, exported persistent data .......................................... 252
16. Section 2 table description record, exported persistent data .................................. 252
17. Section 2 column description record, exported persistent data .............................. 253
18. Section 1 data record format, exported persistent data .......................................... 254
19. Section 3 record format, exported persistent data .................................................. 254
20. Extracted data ....................................................................................................... 256
21. Two-character product codes ............................................................................... 265
22. OMEGAMON started tasks and automation dependencies .................................... 267
23. OMEGAMON started tasks by priority group ....................................................... 269
Part 1. Planning your monitoring environment

The information in this section is intended to help you plan the configuration and deployment of the OMEGAMON XE monitoring agents and the components of their common infrastructure.

- [Chapter 1, “Overview of the OMEGAMON XE products and IBM Tivoli Management Services on z/OS,” on page 3](#) describes the OMEGAMON XE monitoring agents and the commonly shared components of IBM® Tivoli® Management Services. This section also contains topics that describe changes and enhancements in this release that affect the configuration of monitoring agents and common components.
- [Chapter 2, “Prerequisites and packaging,” on page 21](#) provides information about hardware and software prerequisites, packaging, and special requirements for communications and security.
- [Chapter 3, “Planning your deployment,” on page 25](#) covers some basic decisions you need to make to help you plan for installing and configuring the common components and OMEGAMON® XE monitoring agents on z/OS®.
Chapter 1. Overview of the OMEGAMON XE products and IBM Tivoli Management Services on z/OS

The OMEGAMON XE products are a suite of products used to monitor and manage sophisticated mainframe applications and environments. The OMEGAMON XE products share a common technology, Tivoli Management Services, to provide security, data transfer and storage, notification, user interface presentation, and communication for a number of Tivoli monitoring products.

The topics in this guide describe the planning and configuration procedures common to the OMEGAMON XE products and the shared components that run on z/OS. For instructions on configuring a specific OMEGAMON XE product, see the configuration guide for that product after you have completed the steps described in this guide. For instructions on performing the SMP/E installation of the products and components, see IBM Tivoli Management Services on z/OS: Program Directory for IBM Tivoli Management Services on z/OS and see the Program Directory for each OMEGAMON XE product.

Note: In the documentation for the Tivoli Management Services on z/OS and OMEGAMON XE products, installation includes the SMP/E installation only, and configuration includes the remaining steps to set up the product to run successfully in your environment.

Tivoli Management Services and OMEGAMON XE components

The OMEGAMON XE suite of products uses the Tivoli Management Services infrastructure, which provides security, data transfer and storage, notification mechanisms, user interface presentation, and communication services for products in the IBM Tivoli Monitoring and IBM Tivoli OMEGAMON XE suites in an agent/server/client architecture.

Figure 1 on page 4 shows a simple topology of the various principal components that form the Tivoli Management Services and OMEGAMON XE monitoring infrastructure, and how they interact with each other.
Note: Not all the Tivoli Management Services and OMEGAMON XE components in Figure 1 are required for all customer environments. For example, you do not have to have a Tivoli Enterprise Portal Server installed in your environment to run OMEGAMON XE on z/OS and use the OMEGAMON enhanced 3270 user interface to view data. However, if you enable the self-describing agent feature provided by Tivoli Management Services, the Tivoli Enterprise Services user interface extension is required to control the administration of the self-describing agent feature. This component provides the tacmd command-line utility to interact with a monitoring server. This component is available on the IBM Tivoli Monitoring DVD image.

**Tivoli Enterprise Monitoring Server**

Tivoli Enterprise Monitoring Server (also called the monitoring server) is the nerve center of Tivoli Management Services. The monitoring server performs the following tasks:

- Consolidates the data collected by monitoring agents and distributes the data to the connected clients (OMEGAMON enhanced 3270 user interface and Tivoli Enterprise Portal Server).
- Manages the connection status of the monitoring agents.
- Receives commands from the connected clients and distributes them to the appropriate monitoring agents.
- Sends alerts to the connected clients when specified availability and performance problems are detected.
- Stores historical data and configuration prototypes.
The master monitoring server is called the hub monitoring server. The hub monitoring server acts as the focal point for data collection and distribution. It communicates with monitoring agents, with the OMEGAMON enhanced 3270 user interface, with the Tivoli Enterprise Portal Server, and with the Warehouse Proxy and Summarization and Pruning agents (see "Tivoli Data Warehouse" on page 6).

A remote monitoring server is remote only with respect to the hub monitoring server, not with respect to the monitoring agents. Remote monitoring servers communicate only with the monitoring agents that report to them and with the hub monitoring server to which the monitoring servers report.

Monitoring servers can run on z/OS, Windows, AIX®, Solaris, or Linux systems.

**Tivoli OMEGAMON enhanced 3270 user interface**

The OMEGAMON enhanced 3270 user interface component represents the latest generation of 3270 interfaces for the OMEGAMON family of monitoring products. The new interface preserves many of the valued features of the earlier 3270-based interfaces, but extends the functionality to include many new features including cross-system, cross-plex, and cross-product reporting.

The OMEGAMON enhanced 3270 user interface component enables you to monitor the performance of your z/OS systems, applications, and devices in your environment and helps you identify and troubleshoot problems with those monitored resources. In the interface, data is presented in workspaces and subpanels in which the collected data and relevant information is displayed. You can quickly and easily diagnose problems with monitored resources and take action to correct them. You can also customize the workspaces to suit your requirements, and even design and create your own workspaces and navigation.

For more information, see the IBM Tivoli OMEGAMON XE and Tivoli Management Services on z/OS: OMEGAMON Enhanced 3270 User Interface Guide.

**Tivoli Enterprise Portal Server and clients**

Tivoli Enterprise Portal (also called the portal or the portal client) is the user interface for products using Tivoli Management Services. The Tivoli Enterprise Portal is a thin Java™ client application. It has its own server, the Tivoli Enterprise Portal Server, that communicates with the hub monitoring server to send requests to and retrieve data from monitoring agents on managed systems. Tivoli Enterprise Portal Server (the portal server) builds and formats the portal workspaces that display real-time and historical data collected by the monitoring agents. The portal server can run on Windows, AIX, or Linux systems.

You can access the portal client in any of the following ways:

- Browser client (Internet Explorer or Mozilla Firefox on Windows, Linux, or AIX) connected to a web server embedded in the portal server.
- Desktop client installed on a Windows or Linux system.
- Desktop client downloaded and run by IBM Web Start for Java, and updated at every startup.

For setup information about the portal server and client, see the IBM Tivoli Monitoring: Installation and Setup Guide.

The portal server also includes the optional dashboard data provider which is used by Dashboard Application Services Hub to retrieve read-only monitoring data from the hub monitoring server and monitoring agents. Dashboard Application Services Hub can display this data in monitoring dashboard applications such as the IBM Infrastructure Management Dashboards for Servers or in custom dashboards.
Monitoring agents


OMEGAMON XE monitoring agents are located on monitored, or managed, systems. The agents pass the system or application data they collect to a Tivoli Enterprise Portal, and the data is passed to a connected client user interface (OMEGAMON enhanced 3270 user interface or the Tivoli Enterprise Portal). Monitoring agents can also sample data at specified intervals and store it for short-term historical collection, which can then be transferred to a Tivoli Data Warehouse. Finally, the monitoring agents can also compare the current values of monitored properties against a set of defined conditions and thresholds, and trigger alerts or actions when the current values match the defined conditions or surpass the thresholds.

OMEGAMON XE monitoring agents are available for the following products:
- IBM Tivoli OMEGAMON XE for CICS® on z/OS
- IBM Tivoli OMEGAMON XE for DB2® Performance Expert/Performance Monitor on z/OS
- IBM Tivoli OMEGAMON XE for IMS on z/OS
- IBM Tivoli OMEGAMON XE for Mainframe Networks on z/OS
- IBM Tivoli OMEGAMON XE for Storage on z/OS
- IBM Tivoli OMEGAMON XE for Messaging on z/OS
- IBM Tivoli OMEGAMON XE for z/VM® and Linux
- IBM Tivoli OMEGAMON XE on z/OS

Tivoli Data Warehouse

Tivoli Data Warehouse, an optional component of Tivoli Management Services, is a long-term data store for the performance and analysis data collected by the monitoring agents. The Warehouse Proxy agent periodically receives data from the hub monitoring server or from the monitoring agents and inserts the data into the Tivoli Data Warehouse. On a z/OS system, short-term history data for monitoring agents is maintained in data sets allocated and initialized during product configuration. The Warehouse Proxy agent receives the short-term history data and delivers it to the warehouse.

Two specialized agents interact with the Tivoli Data Warehouse:
- The Warehouse Proxy agent receives the short-term history data and delivers it to the Tivoli Data Warehouse.
- You can use the Summarization and Pruning agent to customize how long to save (pruning) and how often to aggregate (summarization) the data in the Tivoli Data Warehouse database.

The Tivoli Data Warehouse, the Warehouse Proxy agent, and the Summarization and Pruning agent can run on Windows, AIX, Solaris, or Linux systems. The Tivoli Data Warehouse can also run on z/OS. For more information, see IBM Tivoli Monitoring: Installation and Setup Guide and IBM Tivoli Monitoring: Administrator’s Guide.

TMS:Engine

TMS:Engine provides common functions such as communications, multithreaded runtime services, tracing, and logging for the monitoring server and monitoring agents on z/OS. This shared component enables common portable code to make platform-independent system calls. This allows Tivoli Enterprise Monitoring Server code to be compiled for and executed on z/OS, Windows, Linux, and UNIX platforms.
Event synchronization component

The event synchronization component sends updates to situation events that have been forwarded to a Tivoli Enterprise Console® event server or an IBM Tivoli Netcool®/OMNIbus ObjectServer back to the monitoring server. In the Tivoli Enterprise Portal, the Situation Event Console, the Common Event Console and the Tivoli Enterprise Console event views are synchronized with the updated status of the events.

For information about the various configurations of monitoring servers and event servers that you can have in your environment, see the IBM Tivoli Monitoring: Installation and Setup Guide.

You can set up stand-alone monitoring agents (those that are configured in their own address spaces) to run in autonomous mode (without communicating directly with a monitoring server). An autonomous agent can emit Simple Network Management Protocol (SNMP) traps and Event Integration Facility (EIF) events directly to a Netcool/OMNIbus ObjectServer for agent-specific situations (but not for enterprise situations).

The IBM Tivoli Monitoring: Installation and Setup Guide provides instructions for configuring Netcool/OMNIbus ObjectServers to receive the events. For information on specifying which situation events to forward, see the Tivoli Enterprise Portal online help and the IBM Tivoli Monitoring: Tivoli Enterprise Portal User's Guide. For detailed information about configuring and managing autonomous agents, see the "Agent autonomy” topics of the IBM Tivoli Monitoring: Administrator’s Guide.

Tivoli Enterprise Portal Server extended services

Tivoli Enterprise Portal Server extended services (TEPS/e) is an embedded, co-located extension of the Tivoli Enterprise Portal Server that provides J2EE-based Application Server integration facilities. TEPS/e provides support for a federated user repository. For more information about TEPS/e, see the IBM Tivoli Monitoring: Administrator’s Guide.

Tivoli Enterprise Monitoring Automation Server, Registry Services, and OSLC clients

Note: The automation server is not supported if the hub monitoring server is installed on a z/OS system.

The Tivoli Enterprise Monitoring Automation Server is an optional component that can be installed on the same system as the hub Tivoli Enterprise Monitoring Server.

The automation server includes the Open Services Lifecycle Collaboration Performance Monitoring (OSLC-PM) service provider. The Performance Monitoring service provider registers monitoring resources such as computer systems, software servers, and databases with the Registry Services. Registry Services is a Jazz™ for Service Management integration service that provides a shared data repository for products in an integrated service management environment. Products that discover and manage shared IT resources can register these IT resources and the services they offer with Registry Services. Other products acting as OSLC clients can use linked data interfaces to consume data by querying Registry Services for the managed resources or the associated service providers of interest. In the query responses, Registry Services return links to the service providers that can be queried to retrieve more information about the managed resources.

The Performance Monitoring service provider also supports the OSLC-PM RESTful API for retrieving linked data about monitored resources. It accommodates HTTP GET requests for the RDF/XML, compact XML and HTML content types. When RDF/XML and HTML content is requested, the API returns resource metrics defined by the OSLC-PM domain.

Each agent that supports registration of monitored resources includes an OSLC template in their monitoring server application support. This template is used by the Performance Monitoring service
provider to determine which resources to register and what metrics are available for the resources. The IBM Tivoli Monitoring distributed OS agents provide templates to register ComputerSystems and IPAddress resources. See the user guide for each of your agents to determine if they also provide an OSLC template.

Install and configure the Tivoli Monitoring Automation Server if your monitoring environment has agents that provide an OSLC template and you want to integrate IBM Tivoli Monitoring with other products using Registry Services and OSLC linked data interfaces. See the IBM Tivoli Monitoring Administrator’s Guide to find more information on the Performance Monitoring service provider and getting started with OSLC integration. The automation server can be installed on Windows, Linux, AIX and Solaris systems.

Registry Services can be installed on Windows, Linux, and AIX systems.

**IBM Dashboard Application Services Hub, dashboard applications, Tivoli Authorization Policy Server, and Tivoli Common Reporting**

The Dashboard Application Services Hub is a Jazz for Service Management component. The Hub is an optional component for your environment that provides a web-based console component for common task navigation for products, aggregation of data from multiple products into a single view, and message passing between views from different products.

The IBM Tivoli Monitoring dashboard data provider retrieves monitoring agent data for display in the Dashboard Application Services Hub. The dashboard data provider is optionally enabled during the Tivoli Enterprise Portal Server configuration. With the dashboard data provider enabled, Dashboard Application Services Hub users can retrieve read-only data from the hub monitoring server and monitoring agent for display in monitoring dashboards such as the IBM Infrastructure Management Dashboards for Servers or in custom dashboards. The IBM Tivoli Monitoring Authorization Policy Server can also be installed into the Dashboard Application Services Hub. The Authorization Policy Server allows you define roles and permissions, which control the access that dashboard users have to managed systems and managed system groups displayed in monitoring dashboards of the IBM Dashboard Application Services Hub. The tivcmd command line interface for Authorization Policy is used to connect to the Authorization Policy Server and create and work with the policies.

Dashboard Application Services Hub can also be used in conjunction with the Tivoli Common Reporting component of Jazz for Service Management to gather, analyze, and report important trends in your managed environment using historical data from the Tivoli Data Warehouse.

Install and configure IBM Dashboard Application Services Hub if you will be using monitoring dashboards in your environment. Check the user guides for the monitoring agents in your environment to determine if they provide a monitoring dashboard application. You can also use Dashboard Application Services Hub to create custom dashboard pages showing data from any monitoring agents in your environment. IBM Dashboard Application Services Hub is supported on 64-bit Windows, Linux, and AIX systems.

**OMEGAMON XE shared components**

OMEGAMON XE monitoring agents on z/OS share several common components, which are included on the product tape. If you install into an existing environment, with the components at the required level, you might have to delete the FMIDs for these components from the SMP/E installation jobs to avoid errors because the components are already installed. See the Program Directory for each product for more information.

Table 1 on page 9 lists the common z/OS components.
Table 1. Common components of OMEGAMON XE monitoring agents on z/OS

<table>
<thead>
<tr>
<th>Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OMEGAMON subsystem</td>
<td>A z/OS subsystem, running in its own address space, that enables OMEGAMON II components running in other address spaces to monitor dynamic device activity. A single OMEGAMON subsystem can support multiple copies of OMEGAMON II and multiple OMEGAMON II components on a single z/OS image. The OMEGAMON subsystem also collects coupling facility data for Tivoli Management Services on z/OS.</td>
</tr>
<tr>
<td>OMNIMON base (includes the IBM Tivoli OMEGAMON enhanced 3270 user interface)</td>
<td>A set of common code used by several OMEGAMON products to control initialization, security, and I/O for all sessions.</td>
</tr>
<tr>
<td>Shared probes</td>
<td>Data probes shared by several OMEGAMON XE products.</td>
</tr>
<tr>
<td>End-to-End Response Time collector</td>
<td>Provides response time data for several OMEGAMON XE products.</td>
</tr>
</tbody>
</table>

Interoperability and integration with other products

The OMEGAMON XE family of monitoring agents are designed to integrate with each other and with other products that use Tivoli Management Services. These products exploit the power of the Tivoli Enterprise Portal to integrate and correlate performance and availability information from a variety of sources.

For example, you can create context-sensitive links between product workspaces to obtain additional information about systems, subsystems, resources, or network components that are being monitored by other monitoring agents. You can also create links from Tivoli Enterprise Portal workspaces to TN3270-based applications. Additionally, if you have purchased IBM Tivoli OMEGAMON DE on z/OS, you can create custom Tivoli Enterprise Portal workspaces composed of data from a range of Tivoli monitoring solutions (IBM Tivoli Monitoring, IBM Tivoli Composite Application Management, IBM Tivoli NetView® for z/OS, and OMEGAMON XE products). You can use OMEGAMON DE to display metrics from operating systems, middleware, databases, storage, web application servers, and network sources for integrated performance views on a single screen.

OMEGAMON DE on z/OS adds the Status Data Manager bridge component of OMEGAVIEW II for the Enterprise to your Tivoli Management Services environment. After you install OMEGAMON DE on z/OS, you can set up Tivoli Enterprise Portal workspaces that display data collected by more than one OMEGAMON XE product or that integrate OMEGAMON XE data with data from other products that use the Tivoli Enterprise Portal. You can create a consolidated situation for a combination of products. You can also use OMEGAMON DE on z/OS to set up advanced automation policies for notification and resolution of complex problems. For further information about the components and capabilities of OMEGAMON DE on z/OS, see IBM Tivoli OMEGAMON DE on z/OS documentation.

Additionally, OMEGAMON XE products are being integrated with an increasing number of other IBM products. You can generate reports using Tivoli Common Reporting, which gathers data collected by OMEGAMON XE products and presents results in a format allowing the analysis of trends and key metrics in a consistent and integrated manner. Situation events reported by OMEGAMON XE monitoring agents can be forwarded to Tivoli Event Console or IBM Tivoli Netcool/OMNibus for event correlation and management. From the Tivoli Enterprise Portal, you can launch into other web-based or web-enabled Tivoli applications without having to re-enter user credentials, and you can launch in context into the Tivoli Enterprise Portal from applications like Tivoli Business Services Management.

For more information on product integration, see the user’s guides for your products.
New in this release

Enhancements to the common infrastructure and the configuration software may affect configuration of components and monitoring agents. Review the changes in the current release in preparation for configuring or upgrading your software.

New in PARMGEN

The enhancements discussed in this section were included in APARs OA40035 (PTF UA67172 4Q12) and OA40649 (PTF UA67787 1Q13A) for FMID HKCI310.

Launch of PARMGEN Workflow user interface

The PARMGEN Workflow interface has been enhanced to launch from the %GBL_TARGET_HILEV %.TKANCUS SMP/E library:
EXEC '%GBL_TARGET_HILEV%.TKANCUS'

You can continue to invoke PARMGEN using the original invocation of the KCIR@PG1 member:
===> EXEC '%GBL_TARGET_HILEV%.TKANCUS(KCIR@PG1)' 

New z/OS Installation and Configuration Tools welcome panel

This new panel provides access to preparation checklists, the JOBGEN installation tool, and the PARMGEN Workflow configuration tool. The panel also provides information about joining the Service Management Connect System z® community, and the capability to selectively print checklists.

Welcome to the z/OS Installation and Configuration Tools for IBM Tivoli Management Services (TMS) dependent products

Option ==> 

Read/Print Checklists prior to installation and configuration:

1. Checklist: Planning and Acquisition
2. Checklist: Prepare the system
3. Checklist: Review the product components & installation steps
4. SMP/E-install z/OS products with Install Job Generator (JOBGEN)
5. Configure z/OS products with Parameter Generator Workflow (PARMGEN)
6. Checklist: Complete the post-configuration steps
7. About joining the community: Service Management Connect (SMC)

P. Print checklist(s) selectively

The help for this panel provides useful information such as the current maintenance level for PARMGEN, JOBGEN, and the Configuration Tool (ICAT) framework, and important new information such as the minimum for the operational TSO region size.

New options on the Workflow main panel

To switch from one runtime environment to another, you can now place a question mark (?) in the RTE NAME field to display a list of runtime environment from which you can select.

The Workflow main menu contains five new options:
Option 3: KCIJPCCF Clone customized WCONFIG members (Conditional)

The KCIJPCCF job copies the customized members such as imbeds, $JOB CARD, and $GBL$USR from the WCONFIG library of one runtime environment to another. If you are using the same global system procedure libraries, VTAM lists, and the like for your runtime environments, or if you have made customizations to WCONFIG override imbeds, to the $JOB CARD member, or the global PARMGEN configuration profile members such as $GBL$IBM or $GBL$USR for a previous runtime environment that you want to include in a new runtime environment, you can clone those members for a new runtime environment. Cloning existing WCONFIG members means that you do not have to configure those files again in the new environment. You might also clone the WCONFIG members as part of cloning an entire runtime environment, including the LPAR configuration profile.

Option 5: KCIJPMC1 Merge profile from backup %RTE_NAME% and Option 6: KCIJPMC2 Merge profile from model RTE

The KCIJPMC1 and KCIJPMC2 jobs merge already customized values from an existing runtime environment into newly created profiles.

The KCIJPMC1 job is used when you back up an LPAR configuration profile to do one of the following:
- Create a new PARMGEN runtime environment based on values converted from an environment created using the Configuration Tool
- Upgrade an existing environment
- Apply maintenance to an existing environment

The KCIJPMC2 job is used when you are cloning a model runtime environment.

Utility Access PARMGEN utilities

Access PARMGEN generated files in ISPF edit mode.

R New RTE Reset RTE, Status and Date fields

Clears the parameter values identifying the previous runtime environment and the status and dates for the options invoked.

New utility jobs

There are four new stand-alone utility jobs in the %RTE_PLIB_HILEV%.%RTE_NAME%.WKANSAMU library:

KCIJPCPR
Backs up the RK* production user runtime libraries

KCIJPCPW
Backs up the WK* PARMGEN work libraries

These two jobs are intended to replace the KCIJPCY backup job which requires editing to specify which set of libraries to copy into the backup libraries.

KCIJB2R
KCIJB2R (KCIJVB2R if system variables are enabled) is a fallback IEBCOPY job that copies the %RTE_HILEV%.%RTE_NAME%.RK*.%CLONE% backed up RK* production user libraries (RKANCMDU, RKANPARU, RKANSAMU, RKD2PAR, RKD2PRF and RKD2SAM %CLONE% copies) back into the product execution user libraries (RK*). Used if the RK* user libraries need to be restored back to the versions prior to running the KCIJPW2R WK* to RK* user libraries deployment job.

KCIJVSRV
Applicable to generated jobs for runtime environments for which system variables are enabled. Resolves any PARMGEN jobs that contain symbolics. Specify the JB2RESLV value to resolve and send the job through the internal reader (INTRDR).
For example, to submit the standalone WKANSAMU(KC2GLBCPF) OMEGAMON XE for CICS copy KC2GLB* global job, enter the following command:

```
Command ==> C 'JB2RSLV' 'KC2GLBCP' ALL
```

**New WK* member**

The KCJJPPRF profile refresh job supports a more automated mechanism for refreshing the configuration profiles if values are modified after initial deployment. This job is submitted by the KCJJPCFG job if setup values are modified after initial deployment. The KCJJPPRF job runs the KCJJPUP1 and the KCJJPMC1 jobs.

**Simplified process for refreshing profiles**

It now takes only three steps to refresh a modified LPAR configuration profile.

Previously, to refresh a profile (for example, after enabling security or customizing new features after applying maintenance) required six steps:

1. Regenerate and submit KCJJPCFG.
2. Rename the current LPAR profile.
3. Run the KCJJUP1 job to generate a new profile for the runtime environment.
4. Run the KCJJPMC1 job to merge the customized values from the backed up profile into the newly generated profile.
5. Run the $PARSE or $PARSESV job to create the tailored runtime members and jobs.
6. Submit the batch jobs to complete the setup of the environment.

Now, if you regenerate the KCJJPCFG job (Step 1 on the Workflow main menu), you are prompted to provide a name to back up the current LPAR configuration profile. If you specify a name for the backup, the configuration software creates the backup profile in the WCONBAK library. The KCJJPCFG job runs the new KCJJPPRF profile refresh job, which in turn runs the KCJJUP1 and the KCJJPMC1 jobs. This means that to refresh an LPAR profile, you only have to create the runtime members and jobs ($PARSE/$SPARSESV) and submit the LOAD job (KCJJrLOD). For typical maintenance scenarios that involve refreshing profiles, see the SMPE scenarios and the “How to: Change PARMGEN parameters” topic in the *IBM Tivoli OMEGAMON XE and Tivoli Management Services on z/OS: PARMGEN Reference*.

**Global parameters added to setup work environment panel**

Sixteen global runtime environment parameters have been added to the third Set up PARMGEN Work Environment for an RTE panel (KCIP@PG3). The values specified on this panel are inherited by the corresponding parameters in the LPAR configuration profile. Specifying the values on this panel significantly reduces the effort required to customize the profile. For example, if you customize the RTE_STC_PREFIX parameter on this panel, all the PARMGEN-generated started tasks in the runtime environment will be generated using that prefix. If you customize the RTE_VTAM_APPLID_PREFIX, all the PARMGEN-generated applids are created using the specified prefix. Depending on the number of products configured in the environment, this can eliminate the need to customize scores of parameters.

Two of these global parameters (RTE_STC_PREFIX and RTE_VTAM_APPLID_PREFIX) can only be used for initial configuration. The values replace variables in product configuration and cannot be updated from the Set up PARMGEN panel. The $CFG$IBM (profile template) is updated, and used to prepopulate any LPAR profiles. If you need to modify the values for product-specific started tasks or product-specific VTAM major node members and VTAM applids, edit the LPAR profile directly.

**New parameters**

The following parameters have been added to the LPAR configuration profile:
**KDS_X_TEMS_KDSSTRT1_DELAY_INT**

Typically, a monitoring server issues start up commands to any agents configured to run in its address space, including OMEGAMON enhanced 3270 data retrieval agent (DRA). This parameter is used to control how much time the monitoring server delays after its own startup before issuing start up commands to the agents. The default is 180 seconds (3 minutes).

**KDS_KMS_SDA_NO_GRANULAR**

Used to disable and enable granular control of the SDA feature during a V6.2.3 to V6.3.0 upgrade. Specify Y to allow all product and versions to install application support data; specify N to implement granular control. The default is Y. If granular control is enabled, the tacmd **tacmd addSdaInstallOptions** command must be used to specify the products and versions for which the self-describing agent facility is allowed to install data. Otherwise, the monitoring server blocks all updates.

For information on installing the tacmd command line interface, see *IBM Tivoli OMEGAMON XE and Tivoli Management Services on z/OS: Configuring the Tivoli Enterprise Monitoring Server on z/OS.*

**KMS_SECURITY_COMPATIBILITY_MODE**

Beginning with IBM Tivoli Monitoring V6.3.0, all automation requests are required to include a security token. The token validates that the request originated from an authorized source. The **KMS_SECURITY_COMPATIBILITY_MODE** variable (default=Y) is intended to help with migration to Tivoli Monitoring V6.3.0 by allowing a component, such as the OMEGAMON enhanced 3270 user interface, that does not issue tokens to initiate an automation request.

If the Y value is set, the Tivoli Enterprise Monitoring Server generates a default security token so that the automation request can still be processed successfully at the monitoring agent endpoint. However, if the value is set to N, the Tivoli Enterprise Monitoring Server will not generate a default security token and the target monitoring agent, if it is running at IBM Tivoli Monitoring V, will reject the automation request with a security violation.

**New libraries**

Two new libraries have been added.

**RKDSKAS library**

This is a new VSAM library for a hub monitoring server. The library is included to accommodate future updates.

**UKANDATV library**

This user data set is being introduced for future exploitation. It introduces DDNAMEs for features so there will be no interruption to started tasks when the features are implemented.

<table>
<thead>
<tr>
<th>Table 2. New runtime libraries and started task DDNAMEs</th>
</tr>
</thead>
<tbody>
<tr>
<td>DDNAME</td>
</tr>
<tr>
<td>-----------</td>
</tr>
<tr>
<td>QAICKAS</td>
</tr>
<tr>
<td>UKANDATV</td>
</tr>
</tbody>
</table>
New imbed files

After the KCIJPUP1 job is rerun, the WCONFIG library will contain the following new product override imbeds. Review the members and customize them as appropriate.

**KCISXW2R**
A new imbed member, KCISXW2R, has been added to the WCONFIG library. This member allows non-PARMGEN controlled members to be preserved in the RK* user libraries. The imbed contains a user-controlled list of members to EXCLUDE when the KCISW2R sample job clears the RK* user libraries prior to copying refreshed runtime members from the WK* libraries to the RK* production user libraries. This eliminates the need to manually copy back non-PARMGEN-tailored (but required) user-customized members.

**KCISST1**
Imbedded in all generated product started tasks in WKANSAMU to serve as the PARMGEN accounting header.

**KQISXML**
For OMEGAMON XE for Messaging. This member is inserted into the RKANDATV(KQISXML) member. The imbed allows you to specify additional brokers for KQISXML usage. Using the imbed reduces manual post-configuration editing and preserves the customization during maintenance or upgrades.

Updated imbed file

The KAG$PENV imbed file now contains sample agent autonomy-related placeholder parameters as a reference so agent features can be exploited by all stand-alone agents. These parameters are commented out by default.

Kpp$* members are preserved when the KCIJPUP1 job is resubmitted. To regenerate the KAG$PENV member to pick this updates, rename the member before regenerating and resubmitting the KCIJPUP1 job.

Obsolete parameters

The following parameters are now obsolete.

**Kpp_CLASSIC_STC_NUM**
(Where pp = C2, I2.) These parameters are no longer required for OMEGAMON XE for CICS on z/OS or OMEGAMON XE for IMS™ on z/OS products in PARMGEN mode configuration. The subsystem-specific runtime members that are generated based on the number of Classic and CUA pairs or number of IMS subsystem rows are now determined based on reading the actual KC2_CCnn_CLASSIC_* and KI2_I1nn_CLASSIC_* table rows. If subsystems are added or deleted, user have to maintain only the nn table row parameters, rather than the table rows and the Kpp_CLASSIC_STC_NUM counter.

**Kpp_X_SECURITY_USER_EXIT** and **Kpp_X_SECURITY_RESOURCE_CLASS**
(Where pp = C2, D2, DF, DS, I2, M2, ON) These parameters were originally intended for the TMS:Engine KppINNAM Engine NAM internal security member. However, if RTE_SECURITY_USER_LOGON = ACF2 or TSS, EXIT= is automatically generated as EXIT=KLVA2NEV (ACF2) or EXIT=KLVTNEV (TSS), respectively so this is not a user-controlled parameter. Only a couple of CUA products specify a CLASSES= parameter in their respective NAM member and they use hardcoded default names based on product architecture. So the RESOURCE_CLASS parameter is not currently a user-controlled parameter. The other OMEGAMONs do not use the parameter specific to the KppINNAM member.

**KDS_TEMS_VTAM_APPL_PREFIX**, **Kpp_AGT_VTAM_APPL_PREFIX**, **Kpp_CUA_VTAM_APPL_PREFIX**
These parameters are no longer used in PARMGEN to construct the product-specific VTAM major nodes and VTAM applids. The global runtime environment equivalents are used instead.
These parameters were required for earlier versions of the TMS:Engine, monitoring server, and common agent infrastructure, but not for V6.2.0 and higher.

For OMEGAMON XE for IMS. This flag is no longer needed to determine if the IMS RESLIB data set (specified by the KI2_I1nn_CLASSIC_IMS_RESLIB parameter) needs to be concatenated in the OMEGAMON for IMS started task (KI2_I1nn_CLASSIC_STC parameter).

The KppSUPDI command table exits are now maintained in a common security exit library and you no longer have to edit the security exits outside of the configuration software.

The common security exit library is identified by the RTE_X_SECURITY_EXIT_LIB parameter and allocated by the configuration software. The composite KCIJPSEC security job, which creates security-related members (load modules, encryption key, and other elements) based on the product security requirements, presents the security exits for editing before displaying the job JCL.

The default values for the following parameters have changed.

Tip: As a best practice, compare your customized LPAR configuration profile with the IBM-supplied $CFG$IBM LPAR profile to determine if a new version has changed the value for product parameters.

The previous default for this parameter was the RKANSAM library. The new default is the RKANSAMU library.

The existing LPAR configuration profile value for this parameter is preserved during upgrades or maintenance. If you want to start referencing RKANSAMU as your global exit library, modify your profile and recreate the WKANSAMU(KCIJcSEC) composite security job by rerunning the $PARSESEM job.

The following are the steps required steps to put the new value into effect:

1. Rerun the KCIJcLOD Load job to initially populate the new security exit library with the default security exits now supplied by the OMEGAMON XE for CICS, DB2, IMS and z/OS as %GBL_TARGET_HILEV%.TKANSAM(KppSUPDI) (where pp = OC, O2, OI, OM).
2. Review the security exits in the security exit library to determine if you need to further customize the exits (for example, MODULE=, password, command LEVELs, resource class).
3. Rerun the KCIJcSEC composite security job to refresh the security modules in RKANMODU with any changes made to the exits.

These parameters have a new default of # (hash or pound character) instead of the asterisk ("*"). The # value generates a blank value for the TCP/IP_USERID="" parameter in the xKANPARU(KppINTCP) member of the runtime environment. The blank value allows TCP/IP to decide the IP stack associated with the address space for better load balancing.

The new default is 300000 (instead of 256000).

This value is used for the MINIMUM(%KMC_AGT_STORAGE_MINIMUM_EXTEND%,X) parameter in the xKANPARU(KMCYSIN) member of OMEGAMON XE for Messaging’s KMC (MQ Configuration) agent. The existing configuration profile value for this parameter is preserved; if you want to start using the new configuration default, modify the parameter in the LPAR configuration profile, then rerun the $PARSE or $PARSES job.
**KMQ_AGT_STORAGE_MINIMUM_EXTEND**

The new default is 300000 (instead of 256000).

This value is used for the MINIMUM(%KMQ_AGT_STORAGE_MINIMUM_EXTEND%,X) parameter in the xKANPARU(KMQSYSIN) member of OMEGAMON XE for Messaging’s KMQ (MQ Monitoring) agent. The existing configuration profile value for this parameter is preserved; if you want to start using the new configuration default, modify the parameter in the LPAR configuration profile, then rerun the $PARSE or $PARSESv job.

**KD5_AGT_STORAGE_MINIMUM_EXTEND**

The new default is 300000 (instead of 256000).

This value is used for the MINIMUM(%KD5_AGT_STORAGE_MINIMUM_EXTEND%,X) parameter in the xKANPARU(KD5SYSIN) member of OMEGAMON XE for DB2 Performance Monitoring and OMEGAMON XE for DB2 Performance Expert agents. The existing configuration profile value for this parameter is preserved; if you want to start using the new configuration default, modify the parameter in the LPAR configuration profile, then rerun the $PARSE or $PARSESv job.

**Note:** The KCIJPMC1 and the KCIJPMC2 merge jobs have been enhanced to merge the higher of the two Kpp_AGT_STORAGE_MINIMUM_EXTEND parameter values. When the merged LPAR profile is rebuilt, it is automatically using the new storage settings.

**Automatic submission of KCIJVUPV job**

Variable-named runtime members that include symbolics (*_STC started tasks, *_VTAM_NODE, etc.) are now created automatically in one job ($PARSESv).

Previously, if a runtime environment that had local overrides for system symbols was updated, the KCIJVUPV job had to be submitted after the $PARSESv job to resolve the symbolics. Now, the KCIJVUPV is automatically submitted when the $PARSESv job is rerun.

**Dependency on the SMP/E global CSI fields eliminated**

The CSI_DSN and TARGET_ZONE parameters are no longer required to set up the PARMGEN work environment. These parameters have been removed from the second set up panel (KICP@PG2).

**Easier conversion of ICAT RTEs to PARMGEN**

The KCIJPCNV conversion job now reads values like high-level qualifiers and SMS parameters for a sharing environment if they are set by a shared-to runtime environment.

**Changed PARMGEN WK* and WCONFIG members or jobs**

Implementation of these changes is optional. The members and jobs will be updated in the libraries the next time the $PARSE or $PARSESv job is rerun. If this is a new PARMGEN runtime environment, the members are part of the initial list of members populated into the libraries.

**WCONFIG(KCIJPCFG) and WCONFIG(KCIJUP1)**

- Added a maintenance-level eyecatcher for ease in identifying what level of PARMGEN maintenance the job is generated from.

**All generated STCs in WKANSAMU**

- Added PARMGEN accounting header that provides the level of PARMGEN maintenance or IF that generated the current version of the started task. It also provides the following:
  - location of the PARMGEN work libraries if their high-level qualifier (RTE_PLIB_HILEV) is different from the runtime environment high-level qualifier (RTE_HILEV)
• the user-supplied GBL_DSN_SYS1_PROCLIB, so users know which user-supplied system procedure library was tailored into the PARMGEN WKANSAMU(KCIJPSYS) system copy job to which the STCs were copied
• a commented-out copy of the user’s WCONFIG($JOBCARD))/SYSDATE to indicate which user or jobname created the STC that is running in production and when it was last modified by a PARMGEN job.

WCONFIG(SPARESV, SPAREPR, and SPARESES) members
These jobs now contain a UPVJPRRX step. This step submits the equivalent of the WKANSAMU(KCIJVUPV) system variable IEBUPDTE job if system variable overrides are enabled.

New in Tivoli Management Services V6.3.0
Tivoli Management Services V6.3.0 introduces granular control for self-describing agent feature. Granular control provides control over which application support files are installed on your monitoring server and portal server by the automatic self-describing agent process. V6.3.0 also introduces enhanced auditing of Take Action requests using security tokens. Both these enhancements have considerations for configuration on z/OS.

Granular control for self-describing agent feature (SDA)

In Tivoli Management Services (IBM Tivoli Monitoring) V6.2.3, if SDA support was enabled, application support data was installed for all products and versions that were installed in the SMP/E target libraries. In V6.3.0, on distributed systems, granular control blocks installation of all self-describing agent application data until the tacmd command line interface is used to either specify the agents and versions for which support data is to be installed, or to enable installation for all products and versions.

Because the tacmd command line interface is available only on distributed systems, on a new installation of a V6.3 z/OS hub monitoring server, the granular control feature is disabled by default: if SDA is enabled, all application data is installed. If you want to enable granular control, you must install the tacmd interface and use the tacmd addSdaInstallOptions command to restrict support to specific products and versions. If you are upgrading from Tivoli Management Services V6.2.3 to V6.3.0, and SDA was already enabled on the hub monitoring server, the KDS_KMS_SDA_NO_GRANULAR parameter can be set to Y to retain the current behavior (all products and versions updated) or to N to enable granular control. If you enable control, you must use the tacmd interface to specify what data to install.

After modifying your application data installation options, you can use the tacmd listSdaInstallOptions command to display the current installation configurations for the hub monitoring server. For detailed information and tacmd commands, see the “Self-describing monitoring agents” topics in the IBM Tivoli Monitoring: Administration Guide; the “Managing your self-describing agent installations” and “Dynamically controlling the hub monitoring server self-describing agent capability” topics in the IBM Tivoli Monitoring: Installation and Setup Guide; and the IBM Tivoli Monitoring: Command Reference.

The tacmd component is available on the IBM Tivoli Monitoring DVD image. The component is automatically installed if you install the Tivoli Enterprise Portal desktop client, the Tivoli Enterprise Portal Server, or the Tivoli Enterprise Monitoring Server. If you do not want to install Tivoli Enterprise Portal Server or Tivoli Enterprise Monitoring Server on a distributed platform but you do intend to run with the self-describing agent feature enabled, install the tacmd command line interface on at least one distributed platform. It does not need to be installed on a distributed server, but the system that it is installed on requires access to your hub monitoring server that uses a soap services connection. See “Whether to enable the self-describing agents capability” in the IBM Tivoli OMEGAMON XE and Tivoli Management Services on z/OS: Configuring the Tivoli Enterprise Monitoring Server on z/OS for more information.
**Enhanced auditing of Take Action commands**

Beginning with IBM Tivoli Monitoring V6.3, all automation requests are required to include a security token. The token validates that the request originated from an authorized source. To aid in a staged upgrade, the KMS_SECURITY_COMPATIBILITY_MODE parameter was introduced. This parameter allows a pre-V6.3.0 component that does not issue tokens to initiate an automation request. If the value of the parameter is set to Y, the Tivoli Enterprise Monitoring Server generates a default security token so that the automation request can still be processed successfully at the monitoring agent endpoint. However, if the value is set to N, the Tivoli Enterprise Monitoring Server does not generate a default security token and the target monitoring agent, if it is running IBM Tivoli Monitoring V6.3, will reject the automation request with a security violation.

Currently, the OMEGAMON enhanced 3270 user interface does not issue tokens, so for z/OS monitoring servers, the value KMS_SECURITY_COMPATIBILITY_MODE is set to Y by default.

**New in the previous release**

Recent enhancements affect the configuration of Tivoli Management Services on z/OS and OMEGAMON XE monitoring agents. You might already be familiar with some of the changes described here if you have upgraded your monitoring environment. Each OMEGAMON XE monitoring agent also provides a number of additional new features. See the product-specific Planning and Configuration guides for further details.

**IBM Tivoli OMEGAMON enhanced 3270 user interface**

The IBM Tivoli OMEGAMON enhanced 3270 user interface component is the latest generation of 3270 interfaces for the OMEGAMON family of monitoring products. The new interface preserves many of the valued features of the earlier 3270-based interfaces, but extends the functionality to include many new features including cross-system and cross-product reporting.

As it is a shared OMEGAMON XE component, only a single instance of the OMEGAMON enhanced 3270 user interface address space is required within the customer sysplex to provide a client for users of all supported OMEGAMON XE products. The products that support the OMEGAMON enhanced 3270 user interface provide predefined workspaces that enable you to quickly and easily diagnose problems with monitored resources and take action to correct them. You can customize the workspaces to suit your requirements, and even design and create your own workspaces and navigation.

**Note:** Not all OMEGAMON XE agents currently provide support for the OMEGAMON enhanced 3270 user interface. See the product-specific Planning and Configuration guides for confirmation of support.

For more information, see the *IBM Tivoli OMEGAMON XE and Tivoli Management Services on z/OS: OMEGAMON Enhanced 3270 User Interface Guide*.

**PARMGEN configuration method**

The PARMGEN configuration method is now the preferred method of product configuration. It is particularly suitable for new OMEGAMON XE customers or existing OMEGAMON XE customers in environments in which one person is responsible for configuring all OMEGAMON XE products and components.

By using the PARMGEN Workflow user interface, you edit a comprehensive list of parameters for configuring all installed products and components. You then submit a series of jobs to create a complete runtime environment with the parameter values you specified. If you have runtime environments that are already configured by the Configuration Tool (ICAT) method, a conversion tool is provided to enable you to use the existing parameter values in those runtime environments to set up initial values for new runtime environments to be configured by the PARMGEN method.
Note: The initial version of the PARMGEN configuration method provided in previous releases of Tivoli Management Services was known as PARMLIB.

For more information, see Chapter 4, "Configuring products using the PARMGEN method," on page 63, and the IBM Tivoli OMEGAMON XE and IBM Tivoli Management Services on z/OS: Common Parameter Reference.

Support for self-describing agents
In earlier releases, the initial installation and version upgrades required you to add application support for your monitoring agents (also known as seeding) to the Tivoli Enterprise Monitoring Server for the agents to function correctly. The Tivoli Enterprise Monitoring Server and OMEGAMON XE monitoring agents now provide self-describing agent support. After startup, self-describing agents automatically push application support updates to the self-describing hub monitoring server, the Tivoli Enterprise Portal Server, and the Tivoli Enterprise Portal browser client.

Self-describing monitoring agents eliminate the requirement to recycle the monitoring server after application support updates, and ensure that application support files are current. By default, the self-describing agent feature is enabled within any remote monitoring server and any monitoring agent that provides self-describing agent support but the feature is disabled at the hub monitoring server. If your enterprise includes self-describing agents, you can enable the self-describing agent feature by changing parameter values for the hub monitoring server.

For more information, see IBM Tivoli Management Services on z/OS: Configuring the Tivoli Enterprise Monitoring Server on z/OS.

High-availability hub
You can now configure a high-availability hub monitoring server in any sysplex environment with dynamic virtual IP addressing (DVIPA) and shared DASD. A high-availability hub is configured in its own runtime environment, without any monitoring agents, and configurable on the same LPAR with a remote monitoring server. This configuration allows the hub monitoring server to be relocated to any suitable LPAR in the sysplex with no changes, and with minimal disruption to the components connecting to the hub.

For instructions on configuring a high-availability hub on the z/OS operating system, see IBM Tivoli Management Services on z/OS: Configuring the Tivoli Enterprise Monitoring Server on z/OS. For detailed information about the high-availability hub on z/OS, including information about conversion from a static hub to a high-availability hub, see IBM Tivoli Management Services on z/OS: Configuring the Tivoli Enterprise Monitoring Server on z/OS. For a scenario, see IBM Tivoli OMEGAMON XE and IBM Tivoli Management Services on z/OS: PARMGEN Reference.

Automatic registration of products with the local monitoring server
The runtime environment load processing option now automatically copies the product attribute and catalog members from the thilev.TKANDATV target data set that was installed by SMP/E to the rhilev.rte.RKANDATV runtime environment. You do not have to regenerate and rerun the "Register with local TEMS" job for each product in the runtime environment if you apply maintenance that updates the product attribute and catalog members only. Instead, you can just reload the runtime environment to refresh the members.

Tivoli Enterprise Monitoring Server support for audit logging
Tivoli Management Services provides a new auditing function that you can use to capture significant events occurring in your site's IBM Tivoli Monitoring environment and record them in permanent storage for later retrieval and analysis. Each audit record fully describes some event that has changed the state of your Tivoli Monitoring system. Platforms covered include Windows, UNIX, Linux, i5/OS, and z/OS systems. On a z/OS system, the auditing facility creates and stores information in Systems Management
Facility (SMF) format records. These new auditing and logging records can be stored in the Tivoli Data
Warehouse. Standard reports are provided by using the Tivoli Common Reporting feature. Additionally, a
new Tivoli Enterprise Portal workspace enables you to view auditing and logging records online.

For more information, see the IBM Tivoli Monitoring: Tivoli Enterprise Portal User’s Guide and IBM Tivoli
Management Services on z/OS: Configuring the Tivoli Enterprise Monitoring Server on z/OS.

Web Services SOAP Server available on remote Tivoli Enterprise Monitoring Server
The SOAP server is now started by default on all monitoring servers. This new feature is to support
automation policies that send requests to the local SOAP server. However, all requests that are sent to a
remote monitoring server that do not originate from a policy are rejected. For more information, see IBM
Tivoli Management Services on z/OS: Configuring the Tivoli Enterprise Monitoring Server on z/OS.

Authorization of z/OS Take Action commands by NetView
This subject is now fully documented and unified in a single publication. For more information, see the
IBM Tivoli Management Services on z/OS: Configuring the Tivoli Enterprise Monitoring Server on z/OS.
Chapter 2. Prerequisites and packaging

The topics in this section help you prepare to install and configure the OMEGAMON XE products and the components of Tivoli Management Services on z/OS systems.

- "Software and hardware prerequisites" directs you to the documentation of requirements that must be completed before you begin installing and configuring the products and components.
- "Packaging" describes the contents of the packages for the Tivoli Management Services and OMEGAMON XE products.

Software and hardware prerequisites

Before you begin installation and configuration, be sure you have identified any required any software and hardware prerequisites.

The Preinstallation Requirements and Instructions Technote contains a checklist of requirements and instructions for preinstallation tasks common to OMEGAMON XE products and Tivoli Management Services components being installed on z/OS systems. You can find this Technote by selecting one of your OMEGAMON XE products on the Software Support website at [http://www.ibm.com/software/support](http://www.ibm.com/software/support) and searching for "preinstallation requirements".

The Program Directory for each OMEGAMON XE product contains a complete list of the software and hardware prerequisites for that product. The Program Directory for IBM Tivoli Management Services on z/OS provides instructions for the SMP/E installation of these components and information about their z/OS hardware and software prerequisites. The hardware and software prerequisites for the distributed components of Tivoli Management Services can be found in IBM Tivoli Monitoring: Installation and Setup Guide.

To make sure that you have the latest version of all components, check for any fix packs that might be available. See the Recommended Maintenance Service Levels for OMEGAMON XE products on ITM V6.x Technote. You can find this Technote by selecting one of your OMEGAMON XE products on the Software Support website at [http://www.ibm.com/software/support](http://www.ibm.com/software/support) and searching for recommended maintenance.

Packaging

If you are installing OMEGAMON XE products and Tivoli Management Services components for the first time, you will find familiar IBM fulfillment methods (such as Shop zSeries®), installation tools (such as SMP/E or InstallShield), and installation documentation, including a program directory. You should be aware of the contents of your installation packages.

Product tapes (or their electronic representation) are in the standard IBM relfile format that IBM software manufacturing uses to create the tape images for installation systems such as ServerPac and CBPDO. If you receive your product through CBPDO, maintenance is included on the media for you to install. If you receive your product as part of a ServerPac or SystemPac®, maintenance is preinstalled.

You can order a ServerPac that includes your mainframe Tivoli products with or without the z/OS operating system. For a list of the Tivoli products you can include in a ServerPac, see the Shopz product catalog [https://www14.software.ibm.com/webapp/ShopzSeries/ShopzSeries.jsp](https://www14.software.ibm.com/webapp/ShopzSeries/ShopzSeries.jsp).

Important: The shared components of the IBM Tivoli Management Services on z/OS product are not automatically included in the product package for a monitoring agent on z/OS systems. Be sure to include IBM Tivoli Management Services on z/OS in your order. If you have not ordered any monitoring agents on a z/OS system but you want to install a monitoring server on a z/OS system, you can order
IBM Tivoli Management Services on z/OS by itself. The IBM Tivoli Management Services on z/OS product includes both mainframe and distributed components of Tivoli Management Services (IBM Tivoli Monitoring).

Table 3. Product packaging, Tivoli Management Services on z/OS

<table>
<thead>
<tr>
<th>Media set 1 of 2: z/OS installation</th>
<th>Name and description</th>
<th>Target</th>
</tr>
</thead>
<tbody>
<tr>
<td>Media set 1 of 2: z/OS installation</td>
<td>A downloaded electronic tape image provides the installation software for the mainframe components: Tivoli Enterprise Monitoring Server on z/OS TMS:Engine Software for configuring the product components by either the PARMGEN method or the Configuration Tool (ICAT) method</td>
<td></td>
</tr>
<tr>
<td>The z/OS media set also includes hardcopy license information.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Media set 2 of 2: Distributed installation</th>
<th>Name and description</th>
<th>Target</th>
</tr>
</thead>
<tbody>
<tr>
<td>Media set 2 of 2: Distributed installation</td>
<td>The distributed package includes DVDs and CDs: IBM Tivoli Monitoring DVDs include subdirectories and installation procedures for components and tools on supported Windows, UNIX, and Linux operating systems. IBM DB2 Universal Database™ CDs provide database functions to the components on supported Windows, UNIX, and Linux operating systems. Language packs provide online help and presentation files in the supported national languages: IBM Tivoli Monitoring Language Packs DVD or CDs DB2 National Language Pack CDs</td>
<td></td>
</tr>
</tbody>
</table>

Table 3 shows the contents of a product package for Tivoli Management Services on z/OS.

IBM Tivoli Monitoring publications are not included as part of the product package. You can find the latest publications in the IBM Tivoli Monitoring and OMEGAMON XE Information Center at [http://pic.dhe.ibm.com/infocenter/tivihelp/v61r1/index.jsp](http://pic.dhe.ibm.com/infocenter/tivihelp/v61r1/index.jsp). For a list of IBM Tivoli Monitoring publications, see “IBM Tivoli Monitoring library” on page 272.

An OMEGAMON XE monitoring agent on z/OS product package includes the following items:
- z/OS FMIDs either on tape or in electronic format.
  Product tapes are in CBPDO or ServerPac format and contain product-specific FMIDs.
- Physical CDs or DVDs or electronic CD or DVD images that are required for some core product functions.
- Publications on CD or DVD as well as in hardcopy or electronic form.

Table 4 on page 23 summarizes the contents of the packages.
### Table 4. Product packaging, OMEGAMON XE monitoring agents

<table>
<thead>
<tr>
<th>Media set 1 of 2: z/OS installation</th>
<th>Name and description</th>
<th>Target</th>
</tr>
</thead>
</table>
| Product tapes or electronic tape images provide the installation software for the monitoring agent and its components: | • OMEGAMON XE monitoring agent  
• OMNIMON base (as appropriate)  
• Shared probes (as appropriate)  
• End to End Response Time collector (as appropriate) |        |
| The agent media set also includes hardcopy publications: | • OMEGAMON XE product-specific license information and the IBM International Program License Agreement (IPLA)  
• IBM Tivoli OMEGAMON XE Monitoring Agents on z/OS: Quick Start Guide |        |

<table>
<thead>
<tr>
<th>Media set 2 of 2: Distributed installation</th>
<th>Name and description</th>
<th>Target</th>
</tr>
</thead>
</table>
| The agent media set includes the following DVDs or CDs or electronic DVD or CD images: | • IBM Tivoli OMEGAMON XE Documentation DVD. The most current documentation is found at [http://publib.boulder.ibm.com/infocenter/tivihelp/v15r1/index.jsp?toc=/com.ibm.itm.doc/toc.xml](http://publib.boulder.ibm.com/infocenter/tivihelp/v15r1/index.jsp?toc=/com.ibm.itm.doc/toc.xml)  
• OMEGAMON XE application support CD or DVD.  
• OMEGAMON XE language support CD or DVD. | |
Chapter 3. Planning your deployment

Review the Preinstallation Requirements and Instructions Technote and the “Prerequisites and packaging section” and make sure all prerequisites have been fulfilled. Then make the planning decisions discussed in this section.

Other sources of information:
• If you are upgrading your monitoring environment, see the IBM Tivoli OMEGAMON XE and IBM Tivoli Management Services on z/OS: Upgrade Guide in the IBM Tivoli Monitoring and OMEGAMON XE Information Center at http://pic.dhe.ibm.com/infocenter/tivihelp/v61r1/index.jsp In the product list on the left side of the information center, double-click Shared OMEGAMON XE publications to find the IBM Tivoli OMEGAMON XE and IBM Tivoli Management Services on z/OS: Upgrade Guide and other shared publications.
• For information on installing, configuring, and upgrading the distributed components of Tivoli Management Services, see the IBM Tivoli Monitoring: Installation and Setup Guide in the IBM Tivoli Monitoring section of the IBM Tivoli Monitoring and OMEGAMON XE Information Center.
• For product-specific installation, configuration, and upgrade information, see the documentation for each product in the information center and see the Technotes for each product. See “Other sources of documentation” on page 273 for information on accessing Technotes.

Decision 1: Whether to install into a shared CSI

To manage a suite of monitoring agent products with a monitoring server on z/OS systems, install the products into a shared consolidated software inventory (CSI) whenever possible.

A shared CSI has two main advantages:
• A shared CSI eliminates duplication and can reduce space requirements by as much as 75%. The monitoring agents share components that must be duplicated if different target and distribution zones are used.
• In a shared CSI, SMP/E can automatically manage IFREQ situations across product components. Otherwise, the IFREQ situations must be managed by running SMP/E cross-zone reports. For information about cross-zone reports, see the IBM SMP/E for z/OS documentation.

Note: While most of the monitoring agents on a z/OS system can coexist in a shared CSI, IBM cannot guarantee that these products can coexist with products from other vendors.

If you install a product or component into an existing CSI that contains a previous version of the same product or component, SMP/E deletes the previous version during the installation process. To maintain more than one version of products or components, you must install them into a separate CSI from the CSI that contains the previous version.

If you decide to install into a shared CSI, follow these guidelines:
• Specify the same high-level qualifier for the target and distribution libraries.
• Make sure you have enough DASD. If you are installing into an existing shared CSI, pay particular attention to the DKAN* and TKAN* common libraries. The DKAN* and TKAN* libraries were originally allocated with secondary space allocation, to allow them to expand when you install additional products or apply maintenance. However, if these libraries have already gone into multiple extents, you might not have sufficient extents available for any necessary expansion. In that case, you might receive error messages during installation.
During normal SMP/E processing, VSAM control interval and control area splits can occur. The resulting fragmentation can degrade SMP/E performance and space utilization. To reorganize the CSI, use your site’s approved utility and method for managing VSAM files.

If you are not installing into a shared CSI, you must use a different high-level qualifier for the target and distribution libraries.

**Decision 2: Where to install your hub and remote monitoring servers**

The hub monitoring server is the focal point for the entire monitoring environment. This server is under a significant load. The hub is responsible for coordinating communication with remote monitoring servers, with the IBM Tivoli OMEGAMON enhanced 3270 user interface, with the Tivoli Enterprise Portal Server, and with local monitoring agents; authenticating users; consolidating and distributing data; storing and tracking situations and policies; and initiating and tracking all generated Take Action commands.

Place the hub monitoring server inside the data center on a high-performance network. Connectivity between the hub monitoring server and other directly connected components such as the remote monitoring servers must be fast and reliable.

Remote monitoring servers communicate only with the monitoring agents that report to them and with the hub monitoring server to which they report. Note that a remote monitoring server is remote with respect to the hub monitoring server, not necessarily with respect to the monitoring agents. If monitoring agents are installed on the same system as a remote monitoring server, that monitoring server is local to the monitoring agents but remote to the hub.

The load on remote monitoring servers is typically low. Load is driven higher if historical data collection is performed on the monitoring servers instead of on the monitoring agents.

You can install monitoring servers on z/OS, Windows, and some UNIX and Linux systems. See IBM Tivoli Monitoring: Installation and Setup Guide for a complete list of supported platforms.

**Location of hub monitoring server**

Many organizations prefer the reliability and availability characteristics of the z/OS platform for the hub monitoring server. If most of your monitoring agents are on a z/OS system, placing the hub monitoring server on a z/OS system can shorten the communications path.

Alternatively, if most of your monitoring agents are on distributed systems, you might prefer a distributed platform for your hub monitoring server. If you install the hub monitoring server on a distributed system, such as a Windows, Linux, or AIX system, you have the option of deploying the portal server on the same system to shorten that communications path.

**Best practices**

Configure your components so that only local monitoring agents and remote monitoring servers report to the hub monitoring server. Configure monitoring agents to report to a hub only if that hub is on the same system as the monitoring agents (that is, if the monitoring agents are local to the hub). If your hub is not z/OS system, then it may make sense to have a remote monitoring server on your z/OS LPARs.

**High-availability hub monitoring server**

An operational hub monitoring server is essential to a monitoring environment. If the hub monitoring server address space fails, or if the system on which the hub is installed has a planned or unplanned outage, the flow of monitoring data comes to a halt. Therefore, it is important to restart the hub or move it to another system as quickly as possible. You can ensure continuous availability by using a high-availability hub monitoring server.
You can configure a high-availability hub monitoring server in any sysplex environment with dynamic virtual IP addressing (DVIPA) and shared DASD. A high-availability hub is configured in its own runtime environment, without any monitoring agents, and can be configured on the same LPAR with a remote monitoring server. System variables are not enabled on an HA hub. This configuration allows the hub monitoring server to be relocated to any suitable LPAR in the sysplex with no changes, and with minimal disruption to the components connecting to the hub.

**Best practices:** It is best practice to create a high-availability hub runtime environment if the following conditions apply:

- You have installed and intend to run OMEGAMON XE on z/OS
- You intend to use the self-describing agent feature for application support installation
- You plan to run a hub monitoring server on z/OS (as opposed to a distributed platform)

Additionally, configure a remote monitoring server running an OMEGAMON on z/OS agent on all LPARs that you want to monitor. If you elect not to create a high-availability hub runtime environment, a recycle may be required if you install maintenance updates to applications, such as OMEGAMON on z/OS, on the hub monitoring server. A recycle might cause a temporary downtime within your monitoring environment.

Figure 2 shows a typical configuration with a high-availability hub runtime environment deployed.

![Figure 2. High-availability runtime environment](image)

In this monitoring environment, you have two LPARs: LPAR 1 and LPAR 2. In LPAR 1, a runtime environment (RTE A) has been created containing just the hub monitoring server, which has been defined as a high-availability hub monitoring server with DVIPA. Because you also want to monitor LPAR 1 in addition to the subsystems running on it (in this example, CICS), a second runtime environment (RTE B) is created with a remote monitoring server and any monitoring agents that are needed.

**Note:** The OMEGAMON on z/OS monitoring agent shares an address space with the remote monitoring server while the OMEGAMON for CICS monitoring agent runs within its own address space.
On the second LPAR, a runtime environment (RTE C) is created in the same fashion as RTE B to monitor the systems and subsystems on that LPAR. It connects to the high-availability hub monitoring server through the DVIPA address. The advantage of the high-availability configuration is that if anything happens to LPAR 1, either planned or unplanned, the hub can be restarted on LPAR 2 without the need for reconfiguring the existing runtime environments.

For detailed information about planning for and configuring a high-availability hub on z/OS systems, see IBM Tivoli OMEGAMON XE and Tivoli Management Services on z/OS: Configuring the Tivoli Enterprise Monitoring Server on z/OS. For detailed information about the high-availability hub on distributed systems, see IBM Tivoli Monitoring: High-Availability Guide for Distributed Systems.

**Self-describing agents**

Self-describing agents include all the necessary application support files for the agents. After startup, self-describing agents automatically push application support updates to the self-describing hub monitoring server, the Tivoli Enterprise Portal Server, and the Tivoli Enterprise Portal browser client, and the enhanced 3270 user interface. Self-describing monitoring agents eliminate the requirement to recycle the monitoring server after application support updates, and ensure that application support files are current.

By default, the self-describing agent feature is enabled within any remote monitoring server and any monitoring agent that provides self-describing support, but the feature is disabled at the hub monitoring server. If your enterprise includes self-describing agents, you can enable the self-describing agent feature by changing parameter values for the hub monitoring server.

If you are planning to configure a hub monitoring server on a z/OS system and enable the self-describing agent feature, you must create Hierarchical File System/zSeries File System (HFS/zFS) directories on z/OS UNIX System Services systems. In this way, the hub monitoring server can process the self-describing application support updates. The UNIX System Services system where you create these directories must have access to a Java runtime environment running under IBM's 31-bit or 64-bit Java SDK Version 5 (or higher) on an HFS or zFS file system.

Ensure that you complete the following steps before starting the configuration of the hub runtime environment:

1. Confirm that you have enough UNIX System Services directory space. The amount required depends on the amount of agents you have deployed in your environment.
2. Install a supported level of Java and note the USS path for your Java home directory.
3. Determine the data set name of your SBXEXEC library.
4. Determine the UNIX System Services path prefix where your runtime environment will store self-describing agent packages. These packages do not need to be created in advance because the configuration process will create them if needed.

For more information on enabling the self-describing agent capability, see IBM Tivoli OMEGAMON XE and Tivoli Management Services on z/OS: Configuring the Tivoli Enterprise Monitoring Server on z/OS.

**Location of remote monitoring servers**

It is a good idea to place a remote monitoring server on every z/OS system where you are installing monitoring agents. In fact, two monitoring agents (OMEGAMON XE on z/OS and OMEGAMON XE for Storage on z/OS) require that you configure them in the same address space as a hub or remote monitoring server.

For advice about placing remote monitoring servers on distributed systems, see the IBM Tivoli Monitoring: Installation and Setup Guide.

**Tips:**
A remote monitoring server can report to a high-availability hub on the same LPAR.

If the hub is not a high-availability hub, it cannot be on the same LPAR as any of the remote monitoring servers that report to it and that are configured to use any TCP/IP protocol.

If more than one remote monitoring server is configured in a z/OS image and if a TCP/IP protocol is being used for communication, the hub to which each remote monitoring server reports must have a unique port number. Otherwise, connectivity problems are likely.

If more than one hub is configured in a z/OS image, each hub must have a unique port number for any nonsecure TCP/IP protocols being used and a unique port number for any secure TCP/IP protocols being used. For information about port number allocation, see "Port number assignments" on page 40.

If a remote monitoring server is to communicate with any monitoring agents that require SNA, the remote monitoring server must be configured for SNA communications. Examples of such monitoring agents include OMEGAMON XE on z/OS (for the EPILOG facility of the OMEGAMON II component) and OMEGAMON XE for Messaging on z/OS (for the 3270 interface component). See the product-specific configuration guides for further information about SNA requirements.

**Monitoring server names**

Each monitoring server has a unique name used for internal processing. This name is known as the TEMS name or the CMS name. You specify the TEMS name for a monitoring server on a z/OS system when you define the runtime environment in which the monitoring server is to be configured. The default name is `rte:CMS` (where `rte` is the name of the runtime environment). The TEMS name is stored as the value of the CMS_NODEID environment variable in the KDSENV member of the `rhilev.rte.RKANPARU` data set.

In general, it is best to accept the default TEMS name. If you want to specify a different name, follow these guidelines:

- The TEMS name must be unique.
- The name is alphanumeric and must begin with an alphabetic character.
- The length of the name must be at least 2 characters and no more than 32 characters.
- The name cannot contain blanks or special characters ($#@). An underscore (_) is permitted and conforms to ISO 9660 standards. A period (.) is also valid.
- The TEMS name is case-sensitive on all platforms. If you use a mixed-case name, you must supply the same mixed-case name when you configure all components and monitoring agents that will connect to the monitoring server.

**SOAP server on z/OS systems**

The SOAP server is an application server plug-in that receives and sends XML data, and provides XML SOAP interfaces into the Tivoli Management Services components and the monitoring agents. The SOAP server is installed with each monitoring server and is enabled during configuration of the hub monitoring server. During configuration, you specify the list of non-local hub monitoring servers with which the local SOAP server can communicate.

The Simple Object Access Protocol (SOAP) is a communication method that uses the Hypertext Transfer Protocol (HTTP) and Extensible Markup Language (XML) as the mechanisms for information exchange. Because web protocols are installed and available for use by all major operating system platforms, HTTP and XML provide a ready solution to the problem of how programs running under different operating systems in a network can communicate with each other. SOAP specifies exactly how to encode an HTTP header and an XML file so that a program in one computer can call a program in another computer and pass it information. It also specifies how the called program can return a response.
An advantage of SOAP is that program calls are likely to get through firewall servers that screen out requests other than those for known applications (through the designated port mechanism). Because HTTP requests are usually allowed through firewalls, programs using SOAP to communicate can generally communicate with programs anywhere.

**SOAP server terminology**

To configure the SOAP server, you must understand certain terms that are used in the configuration.

**Hub monitoring server list**

- Aliasing mechanism for identifying which non-local hub monitoring servers can be accessed from the local SOAP server.

**KDHSTHUBS**

- Global table used by all SOAP servers enabled in the installation library.

**KSHXHUBS**

- Member stored in the `rhilev.rte.RKANPARU` library and containing the hub monitoring server list.

**User access list**

- List of user IDs associated with the hub monitoring server list.

**SOAP server configuration and security**

User access to a SOAP server can be secured in one of two ways: by enabling security and creating user accounts for the hub monitoring server, or by adding specific users to the SOAP server definition. If security is not enabled and no users are added to the server definition, the SOAP server honors all requests from any sender. If security is enabled on the hub monitoring server, the SOAP server honors requests only from users defined to the system authorization facility. However, if any users are added to the SOAP server definition, only those users have access to the server, regardless of whether security is enabled on the monitoring server.

SOAP server configuration creates a KSHXHUBS member in the `rhilev.rte.RKANPARU` library. The KSHXHUBS member contains the hub monitoring server list, an aliasing mechanism for identifying the hub monitoring servers with which the local SOAP server can communicate.

When enabling access to a non-local hub monitoring server from the SOAP server, you can choose one of these options:

- Enable global access to all user IDs that pass logon validation.
- Specify a hub monitoring server list and, for each monitoring server on the list, the user IDs that are allowed to query (read) or update (write to) that monitoring server.

You must still use external security to validate user IDs and passwords, after the user IDs pass validation with the KSHXHUBS member.

The hub monitoring server list in a runtime environment is maintained in the KDSTHUBS global table. KDSTHUBS is used by all SOAP servers that are enabled in the installation library. Any changes you make to the hub monitoring server entries in KDSTHUBS affect KSHXHUBS members used in the different RKANPARU libraries for the runtime environments, when you reconfigure those runtime environments.

**Important:** Do not edit the KSHXHUBS member directly. Its XML tags and values require a specific format and are case-sensitive. If you want to change the contents of the KSHXHUBS member, do so in the PARMGEN configuration profile or in the Configuration Tool.

**Communication protocols**

If you enable the SOAP server while configuring the hub, at least one of the communication protocols you specify must be either IP*:Pipe or IP*:UDP.
The communication protocols for the SOAP server are automatically initialized to the protocol values set for the runtime environment.

**Decision 3: Where to configure your monitoring agents**

Two monitoring agents (OMEGAMON XE on z/OS and OMEGAMON XE for Storage on z/OS) require that you configure them in the same address space as a hub or remote monitoring server. For the other monitoring agents, you have the option to configure each monitoring agent as *stand-alone* (in its own address space) or in the same address space with a monitoring server.

**Terminology tip:** The term *stand-alone* can be confusing, because it has one meaning when applied to a monitoring server and another meaning when applied to a monitoring agent. A *stand-alone monitoring server* is one configured in its own runtime environment, without any monitoring agents. A *stand-alone monitoring agent* is one configured in its own address space, rather than in the same address space with a monitoring server. A stand-alone monitoring agent can be in the same runtime environment with a monitoring server, but a stand-alone monitoring server is never in the same runtime environment with monitoring agents.

**Best practices**

- Unless you are planning to configure a high-availability hub, configure one runtime environment per logical partition (LPAR). In each runtime environment, configure a monitoring server (hub or remote) and all agents required for monitoring the various workloads on that system. Configure all monitoring agents to report to the monitoring server in their runtime environment. See *IBM Tivoli Management Services on z/OS: Configuring the Tivoli Enterprise Monitoring Server on z/OS*.

- Unless a monitoring agent is required to run in the same address space as a monitoring server, it is best to configure each monitoring agent stand-alone (in its own address space). Configuring the monitoring agents stand-alone has several advantages:
  - A stand-alone monitoring agent can be started and stopped independent of the monitoring server.
  - If a monitoring server fails or becomes unresponsive, stand-alone agents that report to it can switch to a secondary monitoring server specified during configuration.
  - You can apply maintenance to a stand-alone monitoring agent without interfering with the operation of other components.
  - Troubleshooting is easier if each monitoring agent is identified in trace logs by its own started task.

Even if you configure a monitoring agent stand-alone, it still might report to the same monitoring server as other monitoring agents. Therefore, make sure that the values you supply when configuring the monitoring server are compatible with the requirements of all the monitoring agents intended to report to that monitoring server. For example, some monitoring agents may require a SNA connection between the agent and the monitoring server.

**Self-describing agents**

Self-describing agents include all the necessary application support files for the agents. After startup, self-describing agents automatically push application support updates to the self-describing hub monitoring server, the Tivoli Enterprise Portal Server, and the Tivoli Enterprise Portal browser client, and the enhanced 3270 user interface. Self-describing monitoring agents eliminate the requirement to recycle the monitoring server after application support updates, and ensure that application support files are current.

By default, the self-describing agent feature is enabled within any remote monitoring server and any monitoring agent that provides self-describing support but the feature is disabled at the hub monitoring server. If your enterprise includes self-describing agents, you can enable the self-describing agent feature by changing parameter values for the hub monitoring server.
For more information, see IBM Tivoli Management Services on z/OS: Configuring the Tivoli Enterprise Monitoring Server on z/OS.

**Autonomous agents**

You can choose to configure stand-alone monitoring agents to run in **autonomous mode** (without communicating directly with a monitoring server). An autonomous agent can emit Simple Network Management Protocol (SNMP) traps and Event Integration Facility (EIF) events directly to a Netcool/OMNIbus ObjectServer for agent-specific situations (but not for enterprise situations). Autonomous agents can be advantageous in environments where disk space or transmission bandwidth is in short supply.

An autonomous agent requires the DSEVT DDNAME in the monitoring agent started task in the `rhilev.rte.RKANSAMU` data set. This DDNAME points to the `rhilev.rte.RKDSEVT` data set.

The IBM Tivoli Monitoring: Installation and Setup Guide provides instructions for configuring Netcool/OMNIbus ObjectServers to receive the events. For information on specifying which situation events to forward, see the Tivoli Enterprise Portal online help and the IBM Tivoli Monitoring: Tivoli Enterprise Portal User’s Guide.

For information about parameters to set for an autonomous agent, see the "Agent autonomy" chapter of the IBM Tivoli Monitoring: Administrator’s Guide. You can add these parameters for an autonomous agent on a z/OS system by editing the `Kpp$PENV` member of the PARMGEN WCONFIG library or by using the Specify Nonstandard Parameters panel in the Configuration Tool (ICAT) (see “Adding, changing, or deleting nonstandard parameters in a runtime environment” on page 124).

For instructions on enabling SNMP V3 passwords for autonomous agents, see “Enabling SNMP V3 passwords for autonomous agents” on page 138.

**Decision 4: Which user interface to use**

Most OMEGAMON XE monitoring products support z/OS-based 3270 “green screen” interfaces as well the graphical Tivoli Enterprise Portal interface, which is based on distributed platforms. The two types of interfaces differ in the information they offer, the infrastructure they require, and the security they provide. The capabilities of each interface, as well as the familiarity of a user community with a specific type of interface, should be considered when deciding which interface to provide.

For many products, it is possible to provide several types, allowing users to access the product in the manner more appropriate to their skills and familiarity. The Tivoli Enterprise Portal interface requires that you that install one or more Tivoli Enterprise Monitoring Servers and a Tivoli Enterprise Portal Server, which must be installed on a distributed platform. The IBM Tivoli OMEGAMON enhanced 3270 user interface requires that you install one or more Tivoli Enterprise Portal Servers, but it offers plex-level as well as system-level and cross-product data and enhanced functionality. The older 3270 user interfaces require less infrastructure, but these interfaces provide more limited information and customization options. Review the following sections to decide which type of interface to use.

**Tivoli OMEGAMON enhanced 3270 user interface**

The OMEGAMON enhanced 3270 user interface retains the virtues of the earlier 3270 interfaces, while offering many of the features of the Tivoli Enterprise Portal in a native 3270 interface.

- The ability to view plex-wide data, data from multiple systems, and data from multiple agents in the same interface or even the same workspace
- The ability to modify product-provided workspaces to create customized views and queries, and or create new workspaces
- The ability to set dynamic filters
- The ability to define or modify thresholds that trigger status indicators
One or more enhanced 3270 interface address spaces can be deployed on any given sysplex. A monitoring agent or a remote monitoring server must run in the same sysplex as the interface to enable discovery and communications with the associated hub monitoring server.

The OMEGAMON enhanced 3270 user interface requires that at least one hub monitoring server be configured. The scope of the data displayed by the interface is determined by the scope of the associated hub monitoring server. If a hub monitoring server configuration connects agents from systems that span multiple sysplexes, the enhanced 3270 interface displays data from multiple sysplexes.

**Important:** The enhanced 3270 user interface uses BPXxxxx services, which are part of UNIX System Services, for TCP/IP (connect, send, receive, and so forth). A user ID defined with an OMVS segment must be created for the enhanced 3270 user interface started task.

**OMEGAMON (3270) and OMEGAMON II (CUA) interfaces**

The older 3270 interfaces require the least amount of product-specific architecture to support. These interfaces perform well and are user-friendly to systems programmers and administrators that are familiar with an ISPF-based 3270 interface. Standard and custom PF Key settings, menu options, and command-line interface options allow for short cuts to commonly viewed screens. While basic customization options allow for highlights and other eye-catcher techniques to be added to the interface, the customization options are limited. The 3270 interfaces use dedicated connection features and avoid disk I/O. Documentation may be out-of-date or difficult to find.

The OMEGAMON (3270) and OMEGAMON II interfaces do not provide plex data, cross-LPAR monitoring, or cross-product views. However, you can get limited multisystem monitoring and cross-system views using OMEGAMON DE. The following OMEGAMON XE products provide older 3270 interfaces:

- IBM Tivoli OMEGAMON XE on z/OS
- IBM Tivoli OMEGAMON XE for CICS on z/OS
- IBM Tivoli OMEGAMON XE for DB2 on z/OS
- IBM Tivoli OMEGAMON XE for IMS on z/OS
- IBM Tivoli OMEGAMON XE for Messaging on z/OS
- IBM Tivoli OMEGAMON XE for Storage on z/OS
- IBM Tivoli OMEGAMON XE for Mainframe Networks on z/OS

OMEGAMON XE on z/OS and OMEGAMON XE for Storage require more infrastructure than the other agents: a Tivoli Enterprise Monitoring Server must be installed in the same runtime environment as each monitoring agent and the OMEGAMON XE component to be configured. (While the OMEGAMON XE on z/OS agent can be configured to run as a stand-alone system, that is, outside of a monitoring server, such a configuration does not support plex data.)

The other agents do not require either a portal server or a monitoring server, and require only configuration of the OMEGAMON II component.

**Tivoli Enterprise Portal interface**

This graphical user interface may be accessed through a Java-based desktop client or a supported browser. The Tivoli Enterprise Portal provides the greatest number of features, but requires the most infrastructure.

The Tivoli Enterprise Portal provides the following features:

- The ability to merge the data from multiple agents or multiple systems into a single workspace for business views and unit of work views
- The ability to define specific conditions that should be monitored (situations)
- The ability to define colored highlights and sounds to alert operators when events occur
• Proactive automation to respond to situations
• Highly customizable graph and table views
• Integral historical data collection, reporting, and warehousing
• Customizable historical reporting
• Integration with other Tivoli and IBM products, including single sign-on
• Event forwarding to event management systems LDAP authentication

The portal uses the standard browser interface and is user-friendly to most users and is often the preferred interface for operation centers, managers, and technical staff. Using the Tivoli Enterprise Portal interface always requires defining and running the Tivoli Enterprise Portal Server, a hub Tivoli Enterprise Monitoring Server, and potentially one or more remote monitoring servers. The agents that will provide data also have to be configured. For more information about the Tivoli Enterprise Portal, see the IBM Tivoli Monitoring: Tivoli Enterprise Portal User’s Guide.

Decision 5: Which configuration method to use

To configure the Tivoli Enterprise Monitoring Server on a z/OS system, the IBM Tivoli OMEGAMON enhanced 3270 user interface and any OMEGAMON XE monitoring agents with a runtime environment, you have the choice between two configuration methods. Both provide the ability to vary the parameters and settings specific to your runtime environment and generate the started tasks and libraries required to run your monitoring environment.

The following configuration methods are available:

PARMGEN Workflow user interface
This method is now the preferred method of product configuration.

Using the PARMGEN Workflow user interface, you edit a comprehensive list of parameters for configuring all installed products and components. You then submit a series of jobs to create a complete runtime environment with the parameter values you specified.

The PARMGEN method can be used for creating new runtime environments and for upgrading existing ones that were created from product versions for which the PARMGEN configuration method was enabled. To upgrade from a product version lower than the first PARMGEN-enabled version, you must use the Configuration Tool (ICAT). To determine the lowest PARMGEN-enabled version of each of your products, see the PARMGEN Technote at http://www.ibm.com/support/docview.wss?uid=swg21417935 For information about configuring your runtime environment using the PARMGEN Workflow user interface, see Chapter 4, “Configuring products using the PARMGEN method,” on page 63.

Configuration Tool method
The Configuration Tool is included with several releases of Tivoli Management Services and OMEGAMON XE. The tool provides a batch mode and an interactive mode to configure a runtime environment.

With batch mode, you can create a single batch job to build, configure, and load a runtime environment for the Tivoli Enterprise Monitoring Server, monitoring agents, or both. See the information in Part 3, “Replicating configured runtime environments,” on page 147 for more information about batch mode configuration.

In interactive mode, ISPF panels guide you through the steps of setting the parameter values and of tailoring and running the configuration jobs. For more information about configuring your runtime environment using the Configuration Tool (ICAT), see the information in Chapter 5, “Configuring products with the Configuration Tool,” on page 103.

The PARMGEN method and the Configuration Tool batch mode is useful for replicating a configured runtime environment to other z/OS systems. The PARMGEN method is preferable for configuring all the products in a runtime environment and then replicating the configuration to other z/OS systems.
**Best practices**

If you have runtime environments that are already configured by the Configuration Tool method, a conversion tool (the KCIIJPCNV job) is provided for using the existing parameter values in those runtime environments to set up initial values for new runtime environments to be configured by the PARMGEN method. After you use the PARMGEN method to configure runtime environments, you cannot use the Configuration Tool to edit or maintain them.

**Configuration parameters**

Whether you decide to use the PARMGEN or Configuration Tool (ICAT) configuration method, the work of configuring your products consists primarily of setting appropriate values for the configuration parameters of runtime environments and of the product components they contain. This section provides a basic explanation of the configuration parameters you will use in performing the tasks described in the following sections.

For reference information about the parameters used in configuring runtime environments and Tivoli Enterprise Monitoring Servers, see *IBM Tivoli OMEGAMON XE and IBM Tivoli Management Services on z/OS: Common Parameter Reference*. For reference information about the parameters used in configuring monitoring agents, see the Parameter Reference for each product.

**Location of stored parameters**

Most configuration parameters and their configured values are stored in the KppENV or KppSYSIN members of the rhilev.rte.RKANPARU data set for each runtime environment.

- **pp** is the 2-character product or component code. For a comprehensive list of the product codes, see Appendix F, “Product codes,” on page 265.
- **rhilev** is the high-level qualifier for the runtime environment data sets
- **rte** is the runtime environment name, which is used as the mid-level qualifier for the runtime environment data sets

The parameters stored in the KppENV member are *environment variables*, which determine the operating characteristics of the runtime environment in which products and components are configured. The parameters stored in the KppSYSIN member are *startup parameters*, which determine the default startup values for each product or component.

Some environment variables and startup parameters are stored in members other than KppENV and KppSYSIN, or in data sets other than RKANPARU. Additionally, some parameters that are neither environment variables nor startup parameters must be included in the runtime libraries for the products and components.

For information about runtime environments, see “Decision 7: What types of runtime environments to set up” on page 44.

**System variables**

System variables are elements that allow systems to share parameter definitions while retaining unique values in those definitions. System variables act like variables in a program; they can take on different values, based on the input to the program. When you specify a system variable in a shared parameter definition, the system variable acts as a placeholder. Each system that shares the definition replaces the system variable with a unique value during initialization.

If you use system variables, the components inherit the system values of the system on which they are started (the host z/OS system). These system-specific values are then automatically loaded into temporary data sets that exist only while the component runs. The result is that the software runs correctly by using the system-specific parameter values for the host z/OS system.
Using system variable support includes the following benefits:

- You can deploy the same software unit, consisting of any or all TMS:Engine-based products, on any system without modification. LPAR-specific values are automatically resolved and substituted at product startup.
- The number of unique runtime environments required is smaller (although unique physical data sets must still exist.) This feature saves storage space, CPU, and labor.
- The same started task JCL and the same VTAM® node can be used on any system without modification.
- You can choose to use a single VTAM major node in place of the individual product major nodes. When generated, a single VTAM major node contains all VTAM applids for all TMS:Engine-based products you have configured in the runtime environment.

Tips:

- You cannot use system variables in the runtime environment of a high-availability hub. (Because the high-availability hub can be started on any LPAR with DVIPA, parameters such as the monitoring server node ID, name, and VTAM major node must be set to static values. See IBM Tivoli Management Services on z/OS: Configuring the Tivoli Enterprise Monitoring Server on z/OS for details.)
- Product started tasks contain a preprocessing step that resolves all system variable specifications in the product parameter members.
- If you enable system variables, product parameter members contain many variables whose values are resolved during startup of the started task.

See the z/OS MVS™ Initialization and Tuning Reference (SA22-7592) for basic information on system variables.

Both the PARMGEN and the Configuration Tool configuration methods support system variables. See Appendix A, “Enabling system variable support,” on page 199 for more information.

Decision 6: How to set up communications between components

A number of communication protocols are available for connections between the hub and remote monitoring servers and between each monitoring agent and the monitoring server it reports to. You configure each monitoring server to use one or more of these protocols, and you assign a priority number to each protocol selected. When the monitoring server attempts to communicate with another component, it tries the highest-priority protocol first. In case of failure, it goes on to the second-priority protocol, and so on. Every component configured to communicate with a monitoring server must have at least one communication protocol in common with that monitoring server.

The communication protocols are of two basic types:

**SNA**

Because some monitoring agents on z/OS systems require SNA, it must be one of the protocols selected for a monitoring server communicating directly with those monitoring agents. However, SNA does not have to be the highest-priority protocol.

**TCP/IP**

The TCP protocols available are IP.PIPE, IP6.PIPE, IP.PIPE, IP6.PIPE, IP6.PIPE, IP6.PIPE, and IP6.PIPE. For the hub monitoring server, at least one of the protocols chosen must be a TCP protocol to support the SOAP server. If you plan to implement long-term historical data collection, communication with the Tivoli Data Warehouse also requires a TCP protocol.

IP.PIPE is the default protocol. All of the piped protocols (IP.PIPE, IP.PIPE, IP.PIPE, and IP.PIPE) enable the monitoring software to traverse most firewall configurations. If you choose piped protocols for the monitoring server and monitoring agents, be aware of the following limitations:

- The maximum number of piped processes per host is 16.
- The piped protocols use only one physical port per process.
If you are configuring a high-availability hub monitoring server, Dynamic Virtual IP Addressing (DVIPA) is required on the z/OS system. This allows the high-availability hub to be moved and restarted on, for example a different LPAR, without the need for reconfiguration should problems occur or if you have a planned maintenance outage on the LPAR where it was originally running. DVIPA requires a piped TCP/IP protocol to be used and the address of the monitoring server must be resolvable through the Domain Name Server (DNS).

Your communication protocol settings for a monitoring server on a z/OS system are saved in the KDE_TRANSPORT environment variable of the KDSENV member of the rhilev.rte.RKANPARU library for the runtime environment.

**Requirements for TCP/IP communication protocols**

The Tivoli Enterprise Monitoring Server and the OMEGAMON enhanced 3270 user interface have special TCP/IP requirements. You must also provide a TCP/IP started task name.

Review the following TCP-related requirements.

**Default OMVS segment**

To use the TCP/IP communication protocols, a Tivoli Enterprise Monitoring Server on a z/OS system requires a default OMVS segment. See the z/OS Communications Server IP Configuration Guide for an explanation of how to provide an OMVS segment. The OMVS segment is also required for the Tivoli OMEGAMON enhanced 3270 user interface.

**UID for the enhanced 3270 user interface started task**

The enhanced 3270 user interface uses BPXmmm services, which are part of UNIX System Services, for TCP/IP. So a user ID defined with an OMVS segment must be created for the enhanced 3270 user interface started task.

**TCP/IP stack name**

You are prompted for a TCP/IP started task name when you create a runtime environment and when you configure a monitoring server or monitoring agent. In a new runtime environment, the default value for the TCP/IP started task name is an asterisk (*). This default, which uses the first TCP/IP stack that was started, is suitable if the LPAR contains a single TCP/IP stack.

If the LPAR contains more than one TCP/IP stack, you can specify the started task name of the TCP/IP stack you want to use; or you can specify the number sign (#), which is translated to a blank and allows the TCP/IP environment to choose the stack to use, either through TCP/IP definitions or through the use of the SYSTCPD DD statement.

Whichever method is used to select a TCP/IP stack in a multi-stack environment, the Tivoli Management Services components continue to use that stack, even if a different stack becomes the primary stack. Therefore, in a multi-stack environment, it is best to specify the started task name of the TCP/IP stack to be used, rather than specifying a wildcard or a blank.

**Firewall support**

Tivoli Management Services supports most common firewall configurations, including those that use address translation (application proxy firewall is a notable exception). To enable this support, use the piped protocols, which open a single port on the firewall for communication by IBM products. If your environment includes a firewall between any components that must communicate with each other, you must specify at least one piped protocol during configuration.

During startup, the monitoring server registers its services and the IP address of these services with a location broker. Clients such as monitoring agents send queries to the location broker to request address information for a service, and receive a list of protocols and IP addresses at which these services are available. The client then sends a specific server request to one of the addresses in the list received from the location broker. Service registration with the location broker assumes address continuity.
If the published address of the monitoring server is identical and reachable for either side of the firewall, then nothing further has to be done to achieve communications in this firewall environment. If the same address cannot be reached from either side of the barrier firewall, then either ephemeral pipe support or broker partitioning is required.

For more information about configuring firewall support, see the following publications:

- Instructions for enabling firewall support for the z/OS components: IBM Tivoli Management Services on z/OS: Configuring the Tivoli Enterprise Monitoring Server on z/OS and the configuration documentation for each monitoring agent.
- Instructions for enabling firewall support for the distributed components: the Firewalls appendix to the IBM Tivoli Monitoring: Installation and Setup Guide.

**Ephemeral pipe support**

Ephemeral pipe support allows piped connections to cross a NAT (network address-translating) firewall without a broker partition file. Ephemeral pipe support is enabled when IP*.PIPE connections cross a NAT firewall. If this default does not suit your network, you can disable ephemeral pipe support and use location broker partitioning instead, or you can force ephemeral connections to be used even when piped communications do not cross a NAT firewall.

Ephemeral pipe support and location broker partitioning for address translation are mutually exclusive.

**Disabling ephemeral pipe support:**

To disable ephemeral pipe support, specify Y as the value of the KDS_TEMs_COMM_ADDRESS_XLAT parameter in the PARMGEN configuration profile, or as the value of the Address translation field of the Configuration ToolIP.PIPE configuration panels.

See "Broker partitioning" on page 39 for information about enabling address translation through a broker partition file.

**Forcing ephemeral connections:**

Under some circumstances, you might want to force ephemeral connections to be used for IP*.PIPE communications, even if they do not cross a NAT firewall. For example, if your configuration requires more than 16 piped processes on a host, ephemeral connections make it possible to exceed the TCP maximum. However, forcing ephemeral connections can have serious drawbacks:

- If an ephemeral connection breaks between a monitoring agent and the monitoring server to which it reports, the monitoring server might not be able to originate a connection back to the monitoring agent.
- The address of a monitoring agent configured for ephemeral connections is randomly assigned in the monitoring server logs. Tracing and troubleshooting are more difficult as a result.
- Extra configuration might be required for enabling communications between the monitoring agents and the Warehouse Proxy agent.

If you want to force ephemeral connections, use the EPHEMERAL option of the KDE_TRANSPORT environment variable.

- EPHEMERAL:Y or EPHEMERAL:OUTBOUND forces outbound connections to be ephemeral.

**Important:** You can specify EPHEMERAL:Y or EPHEMERAL:OUTBOUND for a monitoring agent or a remote monitoring server, but not for a hub.

- EPHEMERAL:INBOUND forces inbound connections to be ephemeral.
You can specify the EPHEMERAL option in the Kpp_X_KDE_TRANSPORT_OPTIONS parameter in your PARMGEN configuration file or by using the Specify Nonstandard Parameters panel in the Configuration Tool (ICAT) (see Chapter 4, “Configuring products using the PARMGEN method,” on page 63 and “Adding, changing, or deleting nonstandard parameters in a runtime environment” on page 124.

For information about ephemeral pipe support on distributed components, see IBM Tivoli Monitoring: Installation and Setup Guide.

Broker partitioning
Address translation is an enhanced security feature of some firewall configurations. With this feature, components that must be reached across the firewall have two unique but corresponding addresses: the external address (valid for components outside the firewall) and the internal address (valid for components inside the firewall). A component on either side of the firewall knows only about the address that is valid for its partition (its own side of the firewall).

You can configure broker partitioning during configuration of the monitoring server on a z/OS system. To do so, you specify Y as the value of the KDS_TEMS_COMM_ADDRESS_XLAT parameter in the PARMGEN configuration profile, or as the value of the Address translation field of the IP.PIPE configuration panels in the Configuration Tool. You also supply, as the value of the KDS_TEMS_PARTITION_NAME parameter in the PARMGEN configuration profile or in the Configuration Tool (ICAT) Partition name field, the label that identifies the location of the monitoring server relative to the firewalls used for address translation.

The partition name that you supply is added to the partition table, which contains labels and associated socket addresses provided by the firewall administrator. The label is used outside the firewall to establish monitoring server connections.

Additionally, you supply the IP address of the monitoring server in its own partition, and the partition name and address assigned to the monitoring server from a location on the other side of each firewall being used. These values are saved as the KDC_PARTITIONFILE environment variable in the KDSENV member of the rhilev.rte.RKANPARU library. KDC_PARTITIONFILE points to a new member, KDCPART, created in the rhilev.rte.RKANPARU library.

Then, when you configure a monitoring agent that reports to the monitoring server, you specify Y as the value of the address translation parameter, and you supply the partition label of the monitoring server. These values are saved as the KDC_PARTITION environment variable in the monitoring agent's KppENV member of the rhilev.rte.RKANPARU library.

The well-known port for the hub monitoring server must be authorized by the firewall administrator. For the IP*.PIPE protocols, no additional ports require authorization. For the IP*.UDP protocols, a range of UDP ports must be authorized.

Firewall gateway support
A firewall gateway provides end-to-end connectivity options for environments with specific TCP/IP connection management policies. The firewall gateway can negotiate numerous firewall hops and supports network address translation. You can use a firewall gateway to configure network traffic so that it is always initiated from the more secure network zone, if two communicating components are in zones with different security levels.

A firewall gateway can be the most advantageous firewall configuration if any of the following conditions apply:

• A single TCP connection cannot span between product components. Example: communication between components requires crossing more than one firewall in an environment with a policy that does not allow a single connection to traverse more than one firewall.
Connection requirements do not allow the default pattern of connections to the hub monitoring server. Example: agents fail to connect to a monitoring server in a zone with higher security than that of the agents; security policy allows a connection to be established from a more secure zone to a less secure zone, but not the other way around.

Open firewall ports must be reduced to a single port or connection. The gateway can consolidate the ports into one. Example: agent failover and monitoring server assignment must be managed symbolically at the hub monitoring server end of the connection. Because gateway connections are made between matching service names, an administrator can change the failover and monitoring server assignment of agents by changing the client proxy bindings at the hub monitoring server.

To configure the firewall gateway, you must perform two tasks:

1. Create an XML document that specifies a set of zones, each of which contains at least one server (upstream) interface with one or more embedded client (downstream) interfaces. The XML document must be stored as a member of the rhilev.rte.RKANPARU library, and the member name must conform to z/OS naming standards (no more than 8 characters).

Here is an example of a gateway XML document, stored as the ZOSPROXY member of the RKANPARU library:

```xml
000002  name="zOSproxy" threads="32">
000003  <zone name="trusted" maxconn="512" error="ignore">
000004  <interface name="zosproxy_upstream" role="proxy">
000005  <bind ipversion="4" localport="pool2K" service="tems_pipe">
000006  <connection remoteport="1920">127.0.0.1</connection>
000007  </bind>
000008  <interface name="zosproxy_downstream" role="listen">
000009  <bind ipversion="4" localport="60902"></bind>
000010  </interface>
000011  </zone>
000012  <portpool name="pool2K">20000-21023 21024-22047</portpool>
000013 </tep:gateway>
```

For reference information about the elements of the gateway XML document, see the “XML document structure” section of the Firewalls appendix to the IBM Tivoli Monitoring: Installation and Setup Guide.

2. In the KppENV member of the rhilev.rte.RKANPARU library, add a KDE_GATEWAY environment variable that references the XML document.

Example:

```
KDE_GATEWAY=ZOSPROXY
```

For information about adding an environment variable, see “Adding, changing, or deleting nonstandard parameters in a runtime environment” on page 124.

Port number assignments

Tivoli Management Services reserves a default well-known port for the first process started on a system (normally, the monitoring server). For monitoring agents and other components, an algorithm based on the port number assigned to the monitoring server calculates the listening port to reserve. A port number cannot be assigned to a component if it is already reserved for another component or application.

The default listening port number for the monitoring server is 1918 for IP.PIPE, IP.UDP, IP6.PIPE, and IP6.UDP. For the secure IP protocols (IP.SPIPE and IP6.SPIPE), the default port number is 3660. For SNA, the default is 135. It is generally best to accept the default setting. However, you might find it necessary to change the setting under some conditions; for example, when the port assigned to a component by the algorithm is already reserved by another application or component.
**Algorithm for allocating port numbers**

Tivoli Management Services uses an algorithm to allocate port numbers for monitoring agents and other components to use in communications with the local monitoring server, under any of the TCP protocols.

The algorithm is

\[
\text{allocated port} = \text{well-known port} + (N \times 4096)
\]

where:
- \( \text{well-known port} \) is the port number assigned to the monitoring server (for example, the default, 1918).
- \( N \) indicates the position of the monitoring agent or other component in the startup sequence.

For example, if there are a monitoring server and two monitoring agents on a system, and the monitoring server uses port 1918, the first monitoring agent in the startup sequence is assigned port 6014 (1918 + 1\( \times 4096 \)), and the second monitoring agent to start is assigned port 10110 (1918 + 2\( \times 4096 \)).

**Controlling port number assignments**

You can change the port number assignments in either of two ways.
- Changing a TCP port number assigned to the monitoring server.
- Using the SKIP and COUNT options on the KDE_TRANSPORT environment variable.

You can also use the POOL option on the KDE_TRANSPORT environment variable to limit connections to a specific range of port numbers.

**Tips:**
- If you decide to modify the KDE_TRANSPORT environment variable, it is best to do so under the guidance of IBM Software Support. See "Support information" on page 275.
- For instructions on changing parameters so that they are not overridden when products are reinstalled or reconfigured, see "Modifying configured parameter values" on page 124.

**Changing a TCP port number assigned to the monitoring server:**

If you change a TCP port number assigned to a monitoring server on a z/OS system, the port numbers allocated for the local monitoring agents also change, because the algorithm starts from the monitoring server's well-known port. You can use this method to avoid conflicts with ports reserved for other applications or components. If you specify a high number, conflicts are less likely than if you use the default or specify a low number for the monitoring server's well-known port.

**Tips:**
- Only one remote monitoring server in an LPAR can report to a given hub. If more than one remote monitoring server is configured in an LPAR, the hub to which each remote monitoring server reports must have a unique port number. Otherwise, connectivity problems might occur.
- If you change a port number for a hub monitoring server, you must reconfigure all the components that communicate with it. If you change a port number for a remote monitoring server, you must reconfigure all the agents that report to it. In a large enterprise, this can be a large task.

**Using the SKIP and COUNT options to control port number assignments:**

For piped protocols (IP*.PIPE, but not IP*.UDP or SNA), you can use the SKIP and COUNT options to control the way port numbers are assigned to components. These options are specified on the KDE_TRANSPORT environment variable in the KppENV member of the rhilev.rite.RKANPARU library (where pp is the component code).

Example:
KDE_TRANSPORT=\nIP.PIPE PORT:1918 COUNT:1 SKIP:2\nIP.UDP PORT:1918\nSNA.PIPE PORT:135

- The PORT option specifies the well-known port for the monitoring server.
- The COUNT:N option is the mechanism for reserving IP.PIPE ports for components that connect to the monitoring server. N is the number of IP.PIPE ports to reserve on the host system, in addition to the well-known port for the monitoring server.

For example, if the well-known port for the monitoring server is 1918, COUNT:3 starts the search at port 6014 (1918 + 1*4096). If the monitoring agent process cannot bind to port 6014, the algorithm tries port 10110 (1918 + 2*4096). If port 10110 is not available, the search goes to port 14206 (1918 + 3*4096). The monitoring agent is assigned to the first available port encountered in the search. The process fails to start if the search reaches the highest port number without a successful binding (port 14206 in this example).

Use the COUNT option to reserve ports for components that must be accessible from outside a firewall. Because these ports must be permitted at the firewall, the ports must be predictable.

- The SKIP:N option specifies the number of ports to skip when starting the search for an available port.

For example, if the well-known port for the monitoring server is 1918, SKIP:2 specifies to start the search at port 10110 (1918 + 2*4096), skipping ports 1918 and 6014 (1918 + 1*4096). The algorithm continues searching until it finds an available port.

Use the SKIP option for components that do not require access across a firewall.

**Using the POOL option to set a range of port numbers:**

After the port allocation algorithm assigns a well-known port to each process, all subsequent ports allocated for connections between components are opaque ports; that is, any available port can be allocated for a connection. You can limit opaque port allocations to a specific range of ports by coding the POOL option with any protocol specified on the KDE_TRANSPORT environment variable.

The POOL option must specify a range of ports no smaller than 2 and no larger than 1024. POOL:1000-2023 is valid; POOL:1000-2024 is not. If more than 1024 ports are required in a pool for a specific protocol, you can code more than one POOL option, as in POOL:1000-2023 POOL:3000-4023.

**Network interfaces**

If your site runs more than one TCP/IP interface or network adapter on the same z/OS image, you can specify network interfaces to be used by monitoring servers and monitoring agents on a z/OS system. You specify the network interfaces in the IP communication protocol parameters for each component.

Before you begin configuring the monitoring server, decide whether you require a network interface list and, if so, which of the following values to specify for it:

- The host name or IP address of the preferred interface.
- A list of host names or IP addresses, in descending order of preference. Use a blank space to separate the entries.
- An asterisk (*) to prefer the interface associated with the default host name for the z/OS image. To display this value, enter TSO HOMETEST at the command-line.
- An exclamation point followed by an asterisk (!*) to use only the interface associated with the default host name for the z/OS image.
- An exclamation point followed by a host name or IP address (!hostname) to use only the interface associated with hostname.
- A minus sign followed by a host name or IP address (-hostname) to use any interface except the one associated with hostname.
If you provide a value for the KDS_TEMSTCP_KDEB_INTERFACELIST parameter in the PARMGEN configuration profile or for the Network interface list field in the Configuration Tool, the KDEB_INTERFACELIST environment variable is added to the KDSENV member of the rhilev.rte.RKANPARU library.

Important:
- If you set the value of this parameter to "!* or 'hostname', you must specify the same value for every component and product configured in all runtime environments on the same z/OS image.
- In the default character set (language locale en_US.ibm-037), the code for an exclamation point is x'5A'. If you are using a character set other than the default, a different character might map to that code. To require a specific network interface, use the character that maps to x'5A' in your character set.

See IBM Tivoli Management Services on z/OS: Configuring the Tivoli Enterprise Monitoring Server on z/OS for special considerations in specifying the KDEB_INTERFACELIST environment variable for a high-availability hub.

Disabling the HTTPS or HTTP server

If your environment uses an HTTP server but not an HTTPS server, or vice versa, you can prevent unnecessary error logging by specifying configuration parameters to disable the unused server. For a monitoring server, the HTTP or HTTPS server is used both for the Service Console and for the SOAP server. For a stand-alone monitoring agent, the HTTP or HTTPS server is used for the Service Console only.

To disable an HTTPS server, the HTTPS option in the KDE_TRANSPORT environment variable must be set to 0 (zero) in the Kpp ENV member of the rhilev.rte.RKANPARU library. To disable an HTTP server, the HTTP option in the KDE_TRANSPORT environment variable must be set to 0 (zero).

You can disable the HTTPS server by setting the value of the Kpp_X_KDE_TRANSPORT_HTTP_OPTIONS parameter to HTTPS:0 in the PARMGEN configuration profile, or use the Specify Nonstandard Parameters panel in the Configuration Tool to add the HTTPS:0 option to the KDE_TRANSPORT environment variable. Example:

KDE_TRANSPORT
New Value: =HTTPS:0\________________________________________
Old Value (if replacing): __________________________________________
Low-level data set qualifier: RKANPARU Member: KDSENV

Similarly, you can disable the HTTP server by setting the value of the Kpp_X_KDE_TRANSPORT_HTTP_OPTIONS parameter to HTTP:0 in the PARMGEN configuration profile, or use the Specify Nonstandard Parameters panel in the Configuration Tool to add the HTTP:0 option to the KDE_TRANSPORT environment variable.

Important: If you disable both the HTTP server and the HTTPS server in the KDSENV member (which contains environment variable definitions for the monitoring server), you disable the SOAP server. The SOAP server is required if you want to use the tacmd command-line utility to administer the self-describing agent feature.

For instructions on adding or changing parameter options so that they are not overridden when products are reinstalled or reconfigured, see “Adding, changing, or deleting nonstandard parameters in a runtime environment” on page 124.
**Decision 7: What types of runtime environments to set up**

A *runtime environment* is a logical grouping of runtime libraries that are referenced by started tasks running on a z/OS image. Before you configure monitoring servers and monitoring agents, you define a runtime environment of a certain type. The type of runtime environment determines the number and types of runtime libraries required.

Table 5 summarizes the types of libraries created during installation and configuration of monitoring servers and monitoring agents on z/OS systems.

**Table 5. Types of libraries**

<table>
<thead>
<tr>
<th>Type of library</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Runtime libraries</td>
<td>General term for libraries referenced by started task procedures.</td>
</tr>
<tr>
<td>Target libraries</td>
<td>SMP/E-maintained target libraries.</td>
</tr>
<tr>
<td>Base libraries</td>
<td>Read-only runtime libraries that the configuration process does not alter and that are shareable between systems. These libraries physically exist in a full or base runtime environment, or as SMP/E target libraries (if a runtime environment shares with SMP/E). The base libraries can contain the actual data sets maintained by SMP/E, or a copy of them. Use a clone or copy of the SMP/E installation libraries for a production environment.</td>
</tr>
<tr>
<td>LPAR-specific libraries</td>
<td>Runtime libraries that are built during configuration to run on a specific logical partition (LPAR). These libraries contain the unique elements required for a particular LPAR and cannot be shared among z/OS images.</td>
</tr>
</tbody>
</table>

The distinction among library types helps you to optimize your product environment. For example, by allocating common base libraries to a single runtime environment that can be shared by other runtime environments, you can substantially reduce the amount of disk space required and simplify the application of maintenance across z/OS images.

Table 6 explains the types of runtime environments that you can create during product configuration.

**Table 6. Types of runtime environments**

<table>
<thead>
<tr>
<th>Type of runtime environment</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full (self-contained) runtime</td>
<td>Runtime environment containing a full set of dedicated libraries, consisting of both LPAR-specific libraries and a copy of the SMP/E installation read-only base libraries eligible for sharing with other runtime environments. See “Example 1. Full (self-contained) runtime environment” on page 47.</td>
</tr>
<tr>
<td>Base runtime environment</td>
<td>Runtime environment containing exclusively read-only, shareable base libraries, which are a subset of the libraries required for running TMS:Engine-based products. Therefore, a base runtime environment must be shared by another runtime environment. See “Example 2. Base runtime environment” on page 48.</td>
</tr>
</tbody>
</table>
Table 6. Types of runtime environments (continued)

<table>
<thead>
<tr>
<th>Type of runtime environment</th>
<th>Description</th>
</tr>
</thead>
</table>

Tips:

See the following suggested strategies for deciding the type of runtime environment to configure based on your requirements:

- If you plan to install monitoring agents on many z/OS images, you can get good results with a sharing-with-base or sharing-with-SMP/E type of runtime environment. See the following examples for considerations when using any type of sharing environment.
- If you want to test quickly, use a sharing-with-SMP/E type of runtime environment.
- If you want to test your configuration on an isolated test system, use a full, self-contained type of runtime environment.

Naming convention for runtime environment data sets

Before you begin configuring runtime environments, you should understand the convention for naming the data sets in the runtime libraries.

Each data set in a runtime library has a name composed of the following parts:

Table 7. Runtime environment data set naming convention

<table>
<thead>
<tr>
<th>Part of the name</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>High-level qualifier, VSAM or non-VSAM</td>
<td>26 bytes</td>
</tr>
<tr>
<td>Mid-level qualifier, which identifies the runtime environment</td>
<td>8 bytes</td>
</tr>
<tr>
<td>Low-level qualifier, which identifies the data set and matches the DDNAME</td>
<td>8 bytes</td>
</tr>
</tbody>
</table>

You set the high-level qualifiers for VSAM and non-VSAM data sets and the mid-level qualifier when you create a runtime environment. The low-level qualifier for each data set is provided by the configuration software and has the format cKppffff

where

- $c$ is the data set class:
  - D for installation distribution libraries (DLIBs).
  - T for target libraries (TLIBs) that were installed by SMP/E.
  - R for runtime libraries.
- $pp$ is the 2-character code for the product or component.
- $ffff$ identifies the function of the data set; U as the final character identifies the data set as a user library. (Examples: CMD or CMDU for a commands data set, PAR or PARU for a parameters data set, and SAM or SAMU for a samples data set.)
For example, the data set IBM.RHILEV1.RTENAME1.RKANPARU has the high-level qualifier IBM.RHILEV1, the mid-level qualifier RTENAME1, and the low-level qualifier RKANPARU (which identifies the data set as a user library containing parameters for the component AN, the conglomerate component for the entire combination of products and components in the runtime environment).

**Naming convention for runtime environment jobs**

When you configure a runtime environment, you edit and submit several jobs. The PARMGEN and Configuration Tool methods use different naming conventions for these jobs.

For a job that is submitted using the PARMGEN method, the naming convention is

\[ KCJc\text{ssss} \]

where \( c = V \) if system variables are enabled and \( c = P \) if system variables are disabled and \( \text{sss} \) is the job name. For example, the KCJVALO job allocates the runtime libraries with system variables, and the KCJPALO job allocates the runtime libraries without system variables.

For job submitted by the Configuration Tool (ICAT) method, the member-naming convention for these jobs is

\[ pp\#t\text{ssss} \]

where

- \( pp \) is the 2-character code for the product or component.
- \( t \) indicates the job type. The following are examples of some of the more common types:
  1. Build the runtime environment.
  2. Load the runtime environment.
  3. Create the runtime members.
  4. Register the monitoring agent with the monitoring server.
  5. Allocate additional runtime libraries job.
  6. Create special HFS and UNIX System Services related members.
  7. Rename the VTAM major node.
  8. Create a single VTAM major node.
  D. Delete the runtime environment.
  L. Process the LU6.2 logmode.
  M. Migrate or upgrade the runtime environment, a product, or a component.
  P. Create runtime members for the persistent data store.
  Q. Allocate and initialize the persistent data store files.
  V. Create the system variable parameter member.
- \( \text{ssss} \) is the unique JCL suffix identifying the jobs submitted for the runtime environment.

**Possible configurations using different types of runtime environments**

The following five examples show different types of runtime environment configurations. The way you choose to set up your runtime environments depends on your site requirements and maintenance procedures.

**Tip:** The data set name (DSN) is composed of the high-level qualifier (rhilev), followed by the mid-level qualifier (&rte), followed by the low-level qualifier. The field settings and library names shown are for illustrative purposes only.
Example 1. Full (self-contained) runtime environment

The full runtime environment contains all libraries required by a particular IBM product and is the easiest runtime environment to create.

This type of runtime environment can be defined in any situation but is most suitable if at least one of the following statements is true:

• Your installation comprises only a single z/OS image.
• You want each z/OS image to be independent.
• You are creating a runtime environment for a specific combination of products that does not exist in any other runtime environment.

The following example represents a full runtime environment called RTE1 that is completely self-contained. All base libraries and LPAR-specific libraries are allocated in RTE1. The base libraries in a full runtime environment are a copy of the SMP/E installation libraries.

Name: RTE1
Type: Full
Hilev: PROD.CAN
Midlev: RTE1
Shares with: (none)

LPAR-specific library DD DSNAME resolution:

//RKANPAR DD DISP=SHR,
  // DSN=PROD.CAN.RTE1.RKANPARU
  // DSN=PROD.CAN.RTE1.RKANPAR

//RKANCMD DD DISP=SHR,
  // DSN=PROD.CAN.RTE1.RKANCMDU
  // DSN=PROD.CAN.RTE1.RKANCMD

//STEPLIB DD DISP=SHR,
  // DSN=PROD.CAN.RTE1.RKANMODU

Base library DD DSNAME resolution:

//RKANMOD DD DISP=SHR,
  // DSN=PROD.CAN.RTE1.RKANMOD
//RKANMODL DD DISP=SHR,
  // DSN=PROD.CAN.RTE1.RKANMODL

Figure 3 illustrates a full runtime environment.

Figure 3. Full runtime environment on a single system

Figure 4 on page 48 shows the way a full runtime environment can be expanded to more than one z/OS image. Each runtime environment is self-contained; the three runtime environments X, Y, and Z on
systems A, B, and C do not share any libraries.

Example 2. Base runtime environment
The base runtime environment allocates shareable base libraries only. A base runtime environment must be used in conjunction with a sharing-with-base runtime environment to provide the complete set of libraries required to run the installed products. The sharing-with-base runtime environment must contain the same set or a subset of the products and components in the base runtime environment.

A base runtime environment is typically used when storage devices are shared or when product maintenance synchronization across systems is desired. Sharing base libraries avoids unnecessary duplication, saves disk space, and simplifies the application of maintenance to a common point.

The following example represents a base runtime environment called RTE2.

Name: RTE2
Type: Base
Hilev: COMMON.BASE
Midlev: RTE2 (Optional for a base runtime environment)
Shares with: (none)

LPAR-specific library DD DSNAME resolution:
There are no LPAR-specific libraries in a base runtime environment.

Base library DD DSNAME resolution:
//RKNMOD DD DISP=SHR,
   // DSN=COMMON.BASE.RTE2.RKANMOD
//RKNMODL DD DISP=SHR,
   // DSN=COMMON.BASE.RTE2.RKANMODL
Figure 5 illustrates a base runtime environment.

```
<table>
<thead>
<tr>
<th>System A</th>
<th>System B</th>
<th>System C</th>
</tr>
</thead>
</table>
```

Note that while a mid-level qualifier is not required for a base runtime environment, it is considered a best practice to use one.

**Example 3. Sharing-with-base runtime environment**

The sharing-with-base configuration is a good choice for environments where storage devices are shared. Using the base runtime environment for common data sets, the sharing-with-base runtime environment contains only LPAR-specific libraries. The base runtime environment cannot contain the LPAR-specific libraries required to run the installed products. The sharing-with-base runtime environment must contain the same set or a subset of the products and components in the base runtime environment.

The configuration software resolves product configuration elements to point at the LPAR-specific libraries and the base runtime environment libraries as necessary.

The following example represents a sharing-with-base runtime environment called SHARRTE2, which obtains its base library information from the base runtime environment (RTE2).

- **Name:** SHARRTE2
- **Type:** Sharing
- **Hilev:** E.F.G
- **Midlev:** SHARRTE2
- **Shares with:** Base RTE2

**LPAR-specific library DD DSNAMES resolution:**

```
//RKANPAR DD DISP=SHR,
  // DSN=E.F.G.SHARRTE2.RKANPARU
  // DSN=E.F.G.SHARRTE2.RKANPAR
//RKANCMD DD DISP=SHR,
  // DSN=E.F.G.SHARRTE2.RKANCMDU
  // DSN=E.F.G.SHARRTE2.RKANCMD
//STEPLIB DD DISP=SHR,
  // DSN=E.F.G.SHARRTE2.RKANMODU
```

**Base library DD DSNAMES resolution:**

```
In typical product started tasks, the LPAR-specific libraries are concatenated ahead of the base libraries, as shown below:

```
//&PROCNAME PROC
//  SYS=SHARRTE2,
//  RHILEV='X.Y.Z',
//  BASEHLEV=COMMON.BASE.RTE2.R
.
//STEPLIB DD DISP=SHR,
//  DSN=&RHILEV..&SYS..RKANMODU
//  DD DISP=SHR,
//  DSN=&BASEHLEV.KANMODL
//  DD DISP=SHR,
//  DSN=&BASEHLEV.KANMOD
//RKANMODL DD DISP=SHR,
//  DSN=&RHILEV..&SYS..RKANMODU
//  DD DISP=SHR,
//  DSN=&BASEHLEV.KANMODL
//  DD DISP=SHR,
//  DSN=&BASEHLEV.KANMOD
```

Figure 6 illustrates a sharing-with-base runtime environment.

![Sharing-with-base runtime environment diagram](image-url)
**Example 4. Sharing-with-full runtime environment**  
The sharing-with-full runtime environment allocates LPAR-specific libraries only, and in this example, obtains its base library information from a full runtime environment. The sharing-with-base runtime environment must contain the same set or a subset of the products and components in the base runtime environment.

This configuration can also be used for environments where storage devices are shared, although the sharing-with-base configuration is the preferred approach.

The following example represents a sharing-with-full runtime environment called SHARRTE1, which obtains its base library information from the full runtime environment (RTE1).

Name: SHARRTE1  
Type: Sharing  
Hilev: E.F.G  
Midlev: SHARRTE1  
Shares with: Full RTE1

**LPAR-specific library DD DSNAME resolution:**

```
//RKANPAR DD DISP=SHR,
  // DSN=E.F.G.SHARRTE1.RKANPARU
  // DD DISP=SHR,
  // DSN=E.F.G.SHARRTE1.RKANPAR
//RKANCMD DD DISP=SHR,
  // DSN=E.F.G.SHARRTE1.RKANCMDU
  // DD DISP=SHR,
  // DSN=E.F.G.SHARRTE1.RKANCMD
//STEPLIB DD DISP=SHR,
  // DSN=E.F.G.SHARRTE1.RKANMODU
```

**Base library DD DSNAME resolution:**

```
//RKANMOD DD DISP=SHR,
  // DSN=PROD.CAN.RTE1.RKANMOD
//RKANMODL DD DISP=SHR,
  // DSN=PROD.CAN.RTE1.RKANMODL
```

*Figure 7 on page 52* illustrates a sharing-with-full runtime environment.
Example 5. Sharing-with-SMP/E runtime environment

The sharing-with-SMP/E runtime environment allocates LPAR-specific libraries only and obtains its base library information from target libraries managed by SMP/E.

Use the sharing-with-SMP/E configuration if at least one of the following statements is true:

- Space is limited on storage devices. This configuration method does not allocate base libraries in the runtime environment, thereby reducing storage requirements.
- You want to activate SMP/E-applied product maintenance immediately.

The following example represents a sharing-with-SMP/E runtime environment called SHARSMP, which obtains its base library information from SMP/E target libraries.

Name: SHARSMP
Type: Sharing
Hilev: E.F.G
Midlev: SHARSMP
Shares with: SMP/E target libraries using target Hilev (thilev)
           INSTALL.SMPE

LPAR-specific library DD DSNAMES resolution:

//Rkanpar DD DISP=SHR,
//    DSN=E.F.G.SHARSMP.RKANPARU
//    DD DISP=SHR,
//    DSN=E.F.G.SHARSMP.RKANPAR
//Rkancmd DD DISP=SHR,
//    DSN=E.F.G.SHARSMP.RKANCMDU
//    DD DISP=SHR,
//    DSN=E.F.G.SHARSMP.RKANCMD
//Steplib DD DISP=SHR,
//    DSN=E.F.G.SHARSMP.RKANMODU
//    DD DISP=SHR,
Figure 8 illustrates a sharing-with-SMP/E runtime environment.

```
//RKAINDD DD DISP=SHR,
  // DD DSN=INSTALL.SMPE.TKANINDD
//RKAINDDL DD DISP=SHR,
  // DD DSN=INSTALL.SMPE.TKANINDDL
```

**Decision 8: What historical data to collect and how to manage it**

Historical data collection is an optional feature that is enabled through the Tivoli Enterprise Portal. When you enable historical data collection, the monitoring agents are instructed to take data samples at a specified interval and store it. The collected data can be displayed in workspaces in the Tivoli Enterprise Portal, warehoused for in-depth analyses and long-term data reporting, and exported to third-party tools for reporting and analysis.

You configure and start historical data collection through the History Collection Configuration window of the Tivoli Enterprise Portal. In this window, you specify the attribute groups for which you want data to be collected, the interval for data collection, the location where you want the collected data to be stored (at the monitoring server or at the agent), and whether you want the data to be migrated to the Tivoli Data Warehouse.
On z/OS systems, short-term historical data is stored in data sets maintained by the persistent data store. When you configure monitoring servers and monitoring agents on a z/OS system, you allocate space for the persistent data store and specify the type of maintenance you want applied to the data sets.

Historical data collection consumes CPU and storage, so it is important to plan data collection carefully. Consider the following planning decisions before you begin to configure historical data collection and storage:

- "What data to collect and how often."
- "Where to collect data."
- "How much space to allocate" on page 55.
- "How to manage collected data" on page 56.

**What data to collect and how often**

The impact of historical data collection and warehousing depends on many factors, including collection interval, frequency of roll-off to the data warehouse, number and size of historical tables collected, amount of data, number of monitored resources, and system size.

The IBM Tivoli Monitoring: Administrator’s Guide discusses the impact of historical data collection and warehousing on Tivoli Management Services components. The documentation for many OMEGAMON XE monitoring agents provides information about the space requirements and size of individual attribute tables, to help you estimate the impact of data collection.

Give careful consideration to what data you actually require. Historical data collection can be specified for individual monitoring servers, products, and attribute tables. Depending on your requirements, you can configure historical data collection for only a subset of attribute tables. Such a configuration can reduce storage and CPU consumption, particularly if you choose not to perform historical data collection for high-volume attribute tables or for attribute tables with many bytes per row (many attributes). Collect only the data that you plan to use in historical reports. Collect that data only as frequently as your enterprise requires.

The collection interval set in the History Collection Configuration window can be as short as a minute or as long as a day. The shorter the interval, the faster and larger the history files grow at the collection location. Short collection intervals also increase CPU consumption and network traffic. Do not set a one-minute collection interval unless your work requires it. If you require frequent collection of historical data, be sure to allocate extra space for the persistent data store. Insufficient allocation of space results in inability to view short-term data and can result in loss of historical data. Allocate enough space for 24 hours of short-term historical data at the location of the persistent data store.

Decisions about what data to collect always involve trade-offs between the usefulness of the data collected and the cost of collecting and managing the data.

**Where to collect data**

Historical data can be stored as short-term data either at the monitoring agent or at the monitoring server to which the agent reports. Where you decide to collect the data determines what steps you take to configure the persistent data store.

**Best practices**

Two OMEGAMON XE monitoring agents (OMEGAMON XE on z/OS and OMEGAMON XE for Storage on z/OS) must be configured within the address space of a monitoring server. Therefore, any historical data collection for these agents must also be stored at the monitoring server. For all other agents, whenever possible configure historical data to be collected and stored at the monitoring agent rather than at the monitoring server.

If an agent is configured to run in the monitoring server address space, like OMEGAMON XE on z/OS and OMEGAMON XE for Storage on z/OS, the persistent data store is configured in two locations:
- During configuration of the monitoring server, the generic persistent data store (RPDSGRP) is configured and the generic data sets (RGENHIS*) are allocated.
- During configuration of the OMEGAMON XE monitoring agent, product-specific historical data sets are allocated.

If a monitoring agent runs in its own address space (stand-alone), and you intend to collect historical data only at the location of the monitoring agent, you can configure the persistent data store during configuration of the agent only. However, to ensure that you have flexibility to collect data at the monitoring server at some later date, configure the persistent data store at the location of the monitoring server as well. If you have no monitoring server in the runtime environment, configuring the persistent data store for the monitoring agent allocates the generic data sets as well as the private data sets.

Data can be uploaded to the Tivoli Data Warehouse for long-term storage and reporting. Data warehousing is configured in the Historical Data Collection window of the Tivoli Enterprise Portal. See the IBM Tivoli Monitoring: Installation and Setup Guide for instructions on setting up the Tivoli Data Warehouse and the Warehouse Proxy and Summarization and Pruning agents. See the IBM Tivoli Monitoring: Administrator’s Guide for instructions on configuring data warehousing.

How much space to allocate

The data written to the persistent data store is organized by tables (attribute groups), groups, and data sets. Each table is assigned to a group, which can have one or more tables and one or more data sets assigned to it. Normally, three data sets are assigned to each group. When you configure the persistent data store, the persistent data store processing computes how much space is required for the group data store files and how much additional required space is required for overhead information. (Overhead information includes the product dictionary, table records, index records, and spare room for buffers that must be reserved for when the data set is full.) However, you might have to adjust the space allocated.

Ideally, you want to allocate enough space to hold 24 hours of data. Eventually, you can determine the correct amount of space by observing how often the maintenance procedures are running and adjusting space according. To help you make more specific calculations, the product documentation for the monitoring agents provides estimated space requirements or information about attribute tables. You might want to make your own calculations, based on site-specific factors: what types of monitoring agents are running, what resources are being monitored, how many resources, and so on.

You can also take a trial-and-error approach. To compute space requirements by trial and error, start historical collection for all resources you want to store in short-term history, and collect at least 25 hours of data. If you cannot draw a report in the Tivoli Enterprise Portal containing the last 24 hours of data for any attribute table, then you must allocate additional storage or data sets or both. You can override the persistent data store defaults to specify data store file placement and space allocation, and to set up maintenance for the data store.

When historical data is collected in product-specific data sets at the agent, allocate enough space so that you can reliably retrieve 24 hours of data for short-term history queries. Of the allocated data sets, one is always in use (Active), one is held aside empty to ensure immediate switchover when the in-use data set fills up, and the remainder are either Empty, Partially Full, or Full. With a default group count of 3 data sets, you have to size the total space so that one-third of the space is sufficient to hold 24 hours of data.

Increasing the group count (number of data sets used to provide the total number of cylinders of persistent data store space) is a way to reduce the total space required, by reducing the percentage of the total space required for the in-use and empty data sets. Increasing the group count to 6 or 8 dramatically
reduces the total space required. (The maximum group count value is 36.) Another benefit of increasing
the group count is that roll-off to the warehouse as well as short-term history queries perform better
when the individual data sets are smaller. The overhead for more frequent data set switches caused by
the increased group count (and presumably by executions of the maintenance procedures) is more than
offset by the gains in efficiency.

When historical data is collected in generic history data sets at the monitoring server, the same
considerations apply, but the data sets must be sized to contain the data from all of the agents that write
to them.

How to manage collected data
If you intend to configure historical data collection, you must make provisions for handling the collected
data. Without such provision, on distributed computers the history data files grow unchecked, using up
valuable disk space. On z/OS systems, the data sets allocated to the persistent data store are emptied and
overwritten.

On distributed systems, if warehousing of the data is configured, the files are automatically pruned after
the data is inserted into the Tivoli Data Warehouse by the Warehouse Proxy agent. On z/OS systems, the
persistent data store data sets are not pruned after data is inserted into the Tivoli Data Warehouse by the
Warehouse Proxy agent. A record is written in the persistent data store to indicate where to begin
exporting data to the Warehouse Proxy agent the next time, and the exported data is kept in the
persistent data store until it is overwritten or deleted by other means.

Tivoli Management Services provides automatic maintenance for the data sets in the persistent data store.
When a data set becomes full, the persistent data store selects an empty data set to make active. When
the data set is active, the persistent data store checks to see whether any data sets are empty. If not, the
persistent data store begins maintenance on the oldest data set. However, before it begins processing, it
checks to see whether one or more of the following functions have been configured:

- BACKUP makes an exact copy of the data set being maintained.
- EXPORT writes the data to a flat file in an internal format that can be used by external programs to
  post-process the data. This function is also used for recovery when the persistent data store detects
  potential problems with the data.
- EXTRACT writes the data to a flat file in human-readable form suitable for loading into other database
  management systems (DBMS).

If no function has been specified, the data is deleted from the data set.

You specify which maintenance options are implemented when you configure the persistent data store.
Appendix E, “Maintaining the persistent data store,” on page 243 provides more information about how
the persistent data store operates and how maintenance is performed.

Decision 9: Which security options to enable
The security options you decide to employ can determine tasks you must complete in advance, such as
arrangements you must make with security administrators or accounts you must set up. Your security
decisions can also dictate certain choices during configuration, such as the selection of secure protocols
when configuring communication between components.

Tivoli Management Services and the OMEGAMON XE monitoring agents offer several security options.
The information in the following topics is intended to help you decide which security options you want
to enforce and what steps you need to take to enable that security.

- “OMEGAMON enhanced 3270 user interface security” on page 57
- “Secure communication between components” on page 57
- “Tivoli Enterprise Portal security” on page 58
Secure communication between components

User IDs and passwords sent between Tivoli Management Services components are encrypted by default. To secure other communications, use SPIPE as the protocol when you configure communication between the Tivoli Enterprise Portal Server and the hub Tivoli Enterprise Monitoring Server, between hub and remote monitoring servers, and between agents and monitoring servers.

Two additional protocols are used for securing communications:

- **HTTPS** to retrieve files and Interoperable Object Reference (IOR). The integrated browser in the client provides HTTPS support on the client side; for the server, consider using a web server that supports HTTPS, such as the IBM HTTP Server. See *IBM Tivoli Monitoring: Installation and Setup Guide* for more information on using web servers.

  For information about disabling an HTTPS server, see "Disabling the HTTPS or HTTP server" on page 43.

- **Internet Inter-ORB Protocol (IIOP)** to secure the communications between the portal server and client. The IIOP uses Secure Socket Layer (SSL). This secure communication uses public key cryptography. SSL on z/OS systems requires the z/OS Communications Server. See *IBM Tivoli Monitoring: Installation and Setup Guide* for more information on using SSL for the distributed components.

IBM Tivoli Monitoring includes the Global Security Toolkit (GSKit) for SSL processing, which is also used in SPIPE and HTTPS. GSKit is installed by default with all distributed components, and its utilities are used to create and manage the encryption of data between components through the use of digital certificates.

On z/OS systems, GSKit is known as the Integrated Cryptographic Service Facility, or ICSF. If ICSF is not installed on the z/OS system, the monitoring server uses an alternative, less secure encryption scheme. Because both components must be using the same scheme, if the hub system does not use ICSF, you must configure the Tivoli Enterprise Portal to use the less secure scheme (EGG1) as well. For more information, see *IBM Tivoli OMEGAMON XE and Tivoli Management Services on z/OS: Configuring the Tivoli Enterprise Monitoring Server on z/OS*.

A default certificate and key are provided with GSKit at installation. A stash file provides the database password for unattended operation. You can also use the key management facility (iKeyMan) in GSKit to generate your own certificates.

**OMEGAMON enhanced 3270 user interface security**

The OMEGAMON enhanced 3270 user interface uses the system authorization facility (SAF) interface to authorize and authenticate users. Planning for security includes deciding who requires access to the OMEGAMON enhanced 3270 user interface, what information they may view, and what Take Action commands they should have permission to invoke; choosing or creating an SAF class that will contain the SAF resources; then ensuring that the required IDs are given the appropriate authority to those resources.

The existence of the SAF user ID and its validity are always checked. Three types of SAF authorization checks are also performed:

- Does the user have the authority to log on to this instance of the OMEGAMON enhanced 3270 user interface?
- Does the user have the authority to view the data from queries for specific types of data (attribute groups) on a specific managed system?
- Does the user have the authority to transmit a Take Action request to a specific managed system?
By default, when the interface is first installed, all users are allowed to log on to the OMEGAMON enhanced 3270 user interface and to view data from any managed system, but all Take Actions are denied. After security is configured, user permissions and the amount of security imposed are assigned by site administrators. All authentication or authorization failures are logged. All Take Action requests are logged.

At a minimum, update the security settings to secure the Take Action function. Failure to correctly secure this powerful function of the OMEGAMON enhanced 3270 user interface might give all users full control to modify the managed system, including starting and stopping applications.

Before security is configured in the environment, a security administrator must complete the following setup tasks:

1. Define an SAF general resource class
2. Define logon profiles to control access to the OMEGAMON enhanced 3270 user interface
3. Define Take Action profiles to control access to OMEGAMON enhanced 3270 user interface data actions
4. Define Query profiles to control access to OMEGAMON enhanced 3270 user interface data sources
5. Permit access to the profiles by appropriate personnel
6. Define a PASSTKT profile to allow the OMEGAMON enhanced 3270 user interface to propagate, identify, and perform authentication checking for query and action requests sent to the Tivoli Enterprise Monitoring Server

**Tivoli Enterprise Portal security**

Access to the Tivoli Enterprise Portal (authorization) is controlled by user accounts (IDs) defined to the portal server. Authentication of users is controlled through either the hub Tivoli Enterprise Monitoring Server or the Tivoli Enterprise Portal Server.

In addition to defining the user IDs that are authorized to log on to the Tivoli Enterprise Portal, these accounts define the permissions that determine the Tivoli Enterprise Portal features a user is authorized to see and use, the monitored applications the user is authorized to see, and the Navigator views (and the highest level within a view) the user can access. An initial sysadmin user ID with full administrator authority is provided during installation so you can log in to the Tivoli Enterprise Portal and add more user accounts. No password is required to log on to the Tivoli Enterprise Portal, unless user authentication is enabled.

The hub Tivoli Enterprise Monitoring Server can be configured to authenticate, or validate, user IDs using either the local system registry or an external LDAP-enabled registry. The Tivoli Enterprise Portal Server can be configured to authenticate through an external LDAP registry.

Authentication by an external LDAP registry is not supported for a hub on z/OS systems. If the hub monitoring server is running on a z/OS system, you must configure RACF or your system authorization facility (SAF) product to authenticate your Tivoli Enterprise Portal users. Alternatively, you can configure authentication through the Tivoli Enterprise Portal Server. If authentication is not enabled through either the monitoring server or the portal server, no authentication is performed and no password is required to log on to the Tivoli Enterprise Portal.

User IDs that have to make SOAP Server requests (including user IDs that issue CLI commands that invoke SOAP server methods) can be authenticated only through the hub monitoring server. User IDs that require the ability to share credentials with other web-enabled Tivoli applications (single sign-on capability, or SSO) must be authenticated through the portal server and mapped to unique user identifiers in an LDAP registry shared by all SSO-eligible Tivoli applications.
Do not enable user authentication before completing and testing at least a basic installation of Tivoli Management Services components and monitoring agents. The first time you configure the hub monitoring server, do not enable security. Complete the following steps before you reconfigure the hub to enable security:

1. Configure all products and verify that they are operating correctly.
2. If you choose a third-party security package, verify that it is installed and configured correctly for your site.
3. Create user IDs in the Tivoli Enterprise Portal, and authorize the users to access resources.
4. Create the user IDs and passwords on the system hosting the hub monitoring server.

You do not have to define and authorize additional user IDs before you enable security, but you must define and authorize the **sysadmin** user ID.

As part of your preparation for deployment, determine which users require access to the Tivoli Enterprise Portal and which features, applications, and views the users must access.

Also determine which users you want to authorize to issue Take Action commands from the Tivoli Enterprise Portal. You can have the issuers of z/OS console commands authorized by Tivoli NetView for z/OS (see *IBM Tivoli Management Services on z/OS: Configuring the Tivoli Enterprise Monitoring Server on z/OS*). Monitoring agents can have product-specific requirements for authentication. Consult the documentation for each monitoring agent for more information.

For instructions on enabling authentication on a hub monitoring server on Windows, UNIX, and Linux operating systems, managing user accounts and permissions, or enabling the Tivoli Enterprise Portal Server for single sign-on, see the *IBM Tivoli Monitoring: Administrator’s Guide*. For instructions on enabling authentication on a hub monitoring server on a z/OS system, see *IBM Tivoli Management Services on z/OS: Configuring the Tivoli Enterprise Monitoring Server on z/OS*.

**OMEGAMON and OMEGAMON II interface security**

You can provide security for the OMEGAMON 3270 and OMEGAMON II interfaces by using a combination of security types and implementations. You must implement security at both the product level and the command level. Product-level security provides user ID and password validation to detect and prevent unauthorized access to the OMEGAMON II product, starting with the System Status panel. Command-level security prevents the unauthorized use of sensitive OMEGAMON for MVS commands from OMEGAMON II panels and by OMEGAMON users.

You can implement product-level and command-level security using either internal or external implementations, or a mixture. An internal implementation uses the security included with the product to control access. An external implementation uses another security package (RACF, CA-ACF2, or CA-TOP SECRET) to control access. To mix security choices at the product level and the command level, you can use, for example, RACF at the product level and internal security at the command level. If security is enabled on a z/OS hub monitoring server, you must use the same security implementation for the 3270 interfaces as is used for the hub.

See the configuration documentation for specific OMEGAMON XE monitoring agents for instructions on implementing security for 3270 interfaces.

**SOAP server security**

User IDs that require access to the SOAP Server, including user IDs that issue commands that invoke SOAP methods, must be authenticated through the hub monitoring server. If user authentication is not enabled on the hub monitoring server, anyone can make requests to the SOAP Server. If user authentication is enabled on the hub, the SOAP Server honors requests only from user IDs and passwords authenticated by the local or external registry. If type of access is specified for specific users, requests from only those users for which access is specified are honored.
You can control access to the SOAP server in two ways:

- You can control who is permitted to make requests by enabling user authentication on the hub monitoring server.

  If the Security: Validate User option is not enabled, the SOAP server honors all requests regardless of the sender. If the Security: Validate User option on the hub monitoring server is enabled, the SOAP server honors requests only from users defined to the operating system or security authorization facility of the host of the monitoring server.

- You can control what type of requests users are permitted to make by configuring the SOAP server.

  Important: If you specify a specific type of access for any users, the SOAP server honors requests only from those users, regardless of whether Security: Validate User is enabled.

For information on configuring the security on the SOAP server, see IBM Tivoli Management Services on z/OS: Configuring the Tivoli Enterprise Monitoring Server on z/OS.

IBM Tivoli Monitoring Service Console security

The Service Console performs user authentication using the native security facility of the host operating system. This means that if you use the Service Console on a z/OS system, your user ID and password are checked by the z/OS security facility (such as RACF/SAF). If you use the Service Console on Windows, the Windows workstation user ID and password are required for authentication.

The IBM Tivoli Monitoring Service Console enables you to read logs and turn on traces for remote product diagnostics and configuration. A password is always required to access the Service Console. Even if a user ID is allowed to log in to the operating system without a password, access to the Service Console is denied without a password. If necessary, create a password for the user ID that is being used to log in to the Service Console.

For more information about the Service Console, see IBM Tivoli Monitoring: Troubleshooting Guide.

What to do next

After you finish planning your deployment and installing components where desired, you can begin the configuration tasks.

Note: If you are planning new started tasks and system automation, see the recommended naming conventions and critical planning information in Appendix G, “Predefining and managing OMEGAMON started tasks,” on page 267.

- For the PARMGEN configuration method, follow the instructions in Chapter 4, “Configuring products using the PARMGEN method,” on page 63.
- For the Configuration Tool (ICAT) method, follow the instructions in Chapter 5, “Configuring products with the Configuration Tool,” on page 103.
Part 2. Configuring components on z/OS

The topics in this section describe the tasks that you perform in configuring your products and components on z/OS systems.

About this task

- For the PARMGEN configuration method, follow the instructions in Chapter 4, “Configuring products using the PARMGEN method,” on page 63.
- For the Configuration Tool method, follow the instructions in Chapter 5, “Configuring products with the Configuration Tool,” on page 103.

Tips:

- Both the PARMGEN Workflow tool and the Configuration Tool interactive and batch modes are currently supported. For a first-time configuration, use the PARMGEN Workflow user interface for configuration of your products and components.
- If you are upgrading your products from a previous version, follow the instructions in IBM Tivoli OMEGAMON XE and IBM Tivoli Management Services on z/OS: Upgrade Guide, and refer to this part of the Common Planning and Configuration Guide for supporting information.

The PARMGEN method can be used for upgrading existing runtime environments that were created from product versions for which the PARMGEN configuration method was enabled. To upgrade from a product version lower than the first PARMGEN-enabled version, you must use the Configuration Tool. To determine the lowest PARMGEN-enabled version of each of your products, see the PARMGEN Technote at http://www.ibm.com/support/docview.wss?uid=swg21417935
Chapter 4. Configuring products using the PARMGEN method

Using the Parameter Generator, or PARMGEN, method, you edit a comprehensive list of parameters for configuring all the products and components that will run in a runtime environment. You then submit a series of jobs to create a complete runtime environment with the parameter values you specified. The PARMGEN Workflow user interface provides a convenient way for you to navigate through the steps for creating a new PARMGEN runtime environment, cloning an existing runtime environment, and maintaining or upgrading a runtime environment.

About this task

The PARMGEN method can be used for creating new runtime environments. It can also be used to upgrade existing environments that were created from product versions for which the PARMGEN configuration method was enabled. To upgrade from a product version lower than the first PARMGEN-enabled version, you must use the Configuration Tool. To determine the lowest PARMGEN-enabled version of each of your products, see the PARMGEN Technote at http://www.ibm.com/support/docview.wss?uid=swg21417935.

A conversion tool is provided for using the parameter values in an existing runtime environment configured by the Configuration Tool method to set up initial values for new runtime environments to be configured by the PARMGEN method. After you use the PARMGEN method to configure new runtime environments, you cannot use the Configuration Tool to edit or maintain them.

The topics in this section gives basic instructions for using the PARMGEN method of configuration. A sample implementation, a runtime environment named DEMO, is presented in which the RTE_HILEV, RTE_VSAM_HILEV, and RTE_PLIB_HILEV parameters are set to TSTEST.&userid. Upon completion of the topics in this section, you should be able to create a new runtime environment containing a monitoring server, monitoring agents, and other components such as the Tivoli OMEGAMON enhanced 3270 user interface. For detailed instructions for implementing specific scenarios, see the IBM Tivoli OMEGAMON XE and Tivoli Management Services on z/OS: PARMGEN Reference.

Review the information in these topics in conjunction with the product-specific planning and configuration guides to ensure that you understand and modify all required parameters for the environment that you want to create. The IBM Tivoli OMEGAMON XE and IBM Tivoli Management Services on z/OS: PARMGEN Reference provides a set of typical deployment scenarios to help you create and maintain runtime environments.

PARMGEN Workflow configuration tool

You can manually create and submit the jobs necessary to set up and configure runtime environments. However, you can also use the Parameter Generator (PARMGEN) Workflow tool. The PARMGEN Workflow interface guides you through the steps required to edit configuration profiles, and to create, edit, and submit the jobs that set up and tailor the runtime environment. It also provides access to PARMGEN libraries and utilities.

The PARMGEN Workflow user interface puts a wrapper around the entire PARMGEN configuration process. From the Parameter Generator (PARMGEN) Workflow main menu you can execute the steps required to configure, and maintain your runtime environments and the products in them.

On the Workflow main panel, you specify the parameters to identify the environment you want to create or modify. Then you select from a list of options to create, clone, maintain, and upgrade runtime environments.
The last two of the options on the menu are utilities.

- **Option U** displays the PARMGEN work libraries (WK*), the GBL_USER_JCL library, and the RTE_X_SECURITY_EXIT_LIB common security exit library, in ISPF EDIT mode.
- **Option R** clears the parameter values identifying the previous runtime environment, and the status and dates for the various steps invoked.

### Configuration profiles

The parameter values for each runtime environment are stored in *configuration profiles*. An LPAR-specific profile contains the configuration values for each runtime environment and for the components and products configured in it. A global profile contains values for global system libraries. A third profile contains user system variable definitions.

**%RTE_NAME%**

LPAR-specific configuration file. This file contains parameters that define the configuration for the runtime environment itself and for all the shared components of the infrastructure and for each product configured in the runtime environment. This profile is named for the runtime environment and is stored in the WCONFIG data set.

**$GBLSUSR**

Configuration file for global system libraries. This profile is stored in the WCONFIG data set. This file can be copied to other runtime environments if these global system libraries are typically the same across LPARs. Customization of this profile is required if this is not a Configuration Tool-to-PARMGEN conversion, or if this is a product upgrade.

Review and possibly customization of this profile is required if the runtime environment is being upgraded and there may be new GBL_* parameters introduced in the new version of the product that were not in the original Configuration Tool batch member supplied. The batch member may still be at the previous version of that product that is now being converted into PARMGEN, and
upgraded at the same time. Review is also required to customize product-specific GBL_* common or global parameters that may apply to more than one product or apply to several components within a product family (for example, GBL_DB2_KD2_CLASSIC_STC parameter applies to all OMEGAMON XE for DB2 PE/PM components).

%RTE_NAME%

System variables configuration file. This file is stored in the data set specified by the GBL_USER_JCL parameter.

Customization of the system variables configuration file is required if the runtime environment is enabled for system variables (RTE_SYSV_SYSVAR_FLAG=Y) and user-defined symbols are being used. If variable overrides are enabled in the runtime environment configuration profile (RTE_X_SYSV_OVERRIDE_SYMBOLS=Y), the values for all types of symbolics must be defined in this member, regardless if they are system, KCIIPARSE-extracted, or a user-defined.

PARMGEN maintains default LPAR ($CFG$IBM) and global profiles ($GBL$IBM), which are used as the templates for user-customized profiles. These default profiles are refreshed with any new or changed configuration parameters when the KCIJPUP1 job is run during upgrades or maintenance.

Default values are provided for all required parameters and for some optional ones. If you do not want to customize these parameters, and you do not want to enable optional features, you can complete the configuration by accepting these defaults. Otherwise, you can specify custom values for these parameters. You can also specify custom values for optional parameters that have no defaults.

Tip: The PARMGEN configuration profiles can have a large number of parameters to review. It may be useful, therefore, to EXCLUDE certain PARMGEN parameters while you focus on a particular configuration task. For example, EXCLUDE all PARMGEN parameters that are for started tasks. Then FIND all occurrences of these parameters so you can quickly do a CHANGE ALL command. PARMGEN provides an EXCLUDE-FIND (XF) macro for this task (see "XF edit macro" on page 243).

For help with the parameter values, consult the following resources:

- Comments in the profiles
- Online help for parameters (see "KCIRPLBS online help macro" on page 243 for more information)
- [PARMGEN technote](http://www.ibm.com/support/docview.wss?uid=swg21417935)
- IBM Tivoli OMEGAMON XE and Tivoli Management Services on z/OS: Common Parameter Reference
- Parameter Reference for each product

**Imbed members**

PARMGEN configuration creates a set of files that get imbedded in a number of the most-commonly updated runtime members in the user libraries--such as KANCMDU(KppAGST), KANPARU(KppENV), KANPARU(KppSYSIN), or KANSAMU(&stc). These imbed files can be used to add new user-defined parameters that might otherwise be overwritten when maintenance or upgrades are performed, or to override existing values.

The imbed files include the following:

- Kpp$C*
- Kpp$P* and Kpp@P*
- Kpp$S*
- KCI$XW2R

where * represents either ENV, SYSIN, or &stc_name. The override members are copied from the IK* interim staging libraries to the corresponding WCONFIG PARMLIB control library by the KCIJPUP1 job. They are then imbedded into the appropriate members (such as started tasks and profiles) by the $PARSE
or $PARSESV job. These files are not overlaid during maintenance or upgrades unless they are renamed. Edit these files before you run the PARSE/PARSESV job.

You can edit these files from the PARMGEN Workflow user interface. Select option U from the PARMGEN Workflow main to view and edit the files. After you have customized the files, you can copy the tailored versions from one runtime environment to another. The KCIJPCCF job can be used to copy them.

Individual products may include their own imbed members. See the product documentation for more information.

**Before you begin**

To ensure a successful configuration, ensure that your system is properly prepared before you begin the configuration of runtime environments using the PARMGEN method.

- Ensure that you have completed SMP/E installation of the required levels of IBM Tivoli Management Services on z/OS components and OMEGAMON XE products for your monitoring environment.
  Check that all components that you want to configure are of the minimum supported level to support the PARMGEN method. For detailed information about supported product versions, see the PARMGEN Technote at [http://www.ibm.com/support/docview.wss?uid=swg21417935](http://www.ibm.com/support/docview.wss?uid=swg21417935).
- Back up the gbl_target_hilev.TKAN* target libraries that were installed by SMP/E. As a minimum, back up the gbl_target_hilev.TKANMOD and gbl_target_hilev.TKANCUS libraries.
- Apply the latest HKCI310 PTF to ensure that you have the latest version of PARMGEN support. For the number of the latest PTF, see the Enablement Support section of the PARMGEN Technote at [http://www.ibm.com/support/docview.wss?uid=swg21417935](http://www.ibm.com/support/docview.wss?uid=swg21417935).
  The HKCI310 PTF adds PARMGEN elements to the following target libraries:
  - gbl_target_hilev.TKANCM
  - gbl_target_hilev.TKANCD
  - gbl_target_hilev.TKANMOD
  - gbl_target_hilev.TKANPAR
  - gbl_target_hilev.TKANSAM

  where gbl_target_hilev is the high-level qualifier of the target libraries that were installed by SMP/E.
- Ensure that the following requirements for using the PARMGEN Workflow user interface are met:
  - A minimum region size of 6000K for the TSO user session
  - The PARMGEN Workflow user interface uses several standard TSO/E commands, and REXX and ISPF services. If you have modified or restricted access to any of these commands or services, the PARMGEN tool may not function correctly.
- If you are planning to configure a monitoring server on a z/OS system and intend to enable the self-describing agent feature, ensure that you have created the appropriate folder structure on z/OS UNIX System Services (USS) under the Hierarchical File System (HFS) or zSeries File System (zFS). This step is required to store and process the provided self-describing files. The monitoring server also requires a Java runtime environment running under the IBM 31-bit or 64-bit JAVA SDK Version 5 (or higher), which is to be installed within the USS file system. A sample job to create a zFS file system is provided in the gbl_target_hilev.TKANSAM(KCIUSSJJB) member.
- Read through the checklists available through the Welcome panel of the z/OS Installation and Configuration Tools:
Start the Workflow user interface

The Parameter Generator (PARMGEN) Workflow user interface steps you through the process of configuring and maintaining runtime environments.

About this task

Use the following procedure to start the Workflow user interface:

Procedure

1. Launch the interface using the following command:

   ```
   EX 'gbl_target_hilev.TKANCUS'
   ```

   The Welcome screen for the z/OS Installation and Configuration Tools is displayed:
2. Select 5 (Configure z/OS products with Parameter Generator Workflow). The main Parameter Generator (PARMGEN) Workflow (KCIP@PG0) panel is displayed:

Welcome to the z/OS Installation and Configuration Tools for IBM Tivoli Management Services (TMS) dependent products

Option ===>

Read/Print Checklists prior to installation and configuration:

1. Checklist: Planning and Acquisition
2. Checklist: Prepare the system
3. Checklist: Review the product components & installation steps
4. SMP/E-install z/OS products with Install Job Generator (JOBGEN)
5. Configure z/OS products with Parameter Generator Workflow (PARMGEN)
6. Checklist: Complete the post-configuration steps
7. About joining the community: Service Management Connect (SMC)

P. Print checklist(s) selectively

F1=HELP F2=SPLIT F3=END F4=RETURN F5=RFIND F6=RCHANGE
F7=UP F8=DOWN F9=SWAP F10=LEFT F11=RIGHT F12=RETRIEVE

2. Select 5 (Configure z/OS products with Parameter Generator Workflow). The main Parameter Generator (PARMGEN) Workflow (KCIP@PG0) panel is displayed:
In the top section of the panel, you specify the parameters specific to the configuration and runtime environment that you are creating or modifying. The bottom part of the panel provides a menu of options for performing the tasks that must be completed to create, update, and maintain a functioning runtime environment. The Workflow interface informs you about the status of each task and when the status was last changed.

What to do next

Identify the runtime environment you want to work with, then proceed with the appropriate steps for the configuration or maintenance scenario you are performing.

Step 1. Set up the PARMGEN work environment for a runtime environment

The first step in the PARMGEN configuration process is to create the PARMGEN work environment. This step defines the libraries used throughout the runtime environment creation process, including the output libraries, and other parameters used in the generation of jobs that contribute to the creation of a runtime environment.

Procedure

1. In the top section of the main Workflow panel, specify the parameters that identify the runtime environment that you are creating or modifying. If this configuration is new or you are returning to an existing configuration, specify all of the following parameters.

   **Tip:** Select option R to clear all the library, status, and date information before you enter new values.

   **GBL_USER_JCL**

   Specifies the data set name of the PARMGEN global user JCL library. If the runtime
environment is enabled for system variables, this data set is concatenated in the CONFIG DDNAME of product started tasks. If the data set does not exist, you are prompted to correct the name or allocate the data set.

The PARMGEN global user JCL library is also where the PARMGEN process creates the system variables configuration profile. Customization of this member is required if a runtime environment is enabled for system variables and you are using user-defined symbols or overriding the default values for system symbols.

**RTE_PLIB_HILEV**

Specifies the non-VSAM high-level qualifier that you want to use for the following PARMGEN work libraries:

- WCONFIG control library where configuration profiles and other members are stored by the KCIJPUP1 job.
- Interim staging (IK*) libraries where PARMGEN product runtime members (template versions) are stored by the KCIJPUP1 job.
- Work output (WK*) libraries where PARMGEN file-tailored runtime members and jobs are created by the $PARSE or $PARSER job.

**RTE_NAME**

Specifies a unique name for the runtime environment that you are configuring. The name that you specify is appended to the RTE_PLIB_HILEV values to make each set of PARMGEN and runtime libraries unique: `rte_plib_hilev.rte_name.WCONFIG`.  

---

2. From the Workflow main panel (KCIP@PG0), select option 1 and press Enter.

**Note:** The values specified (or imported) in this series of panels are stored as GBL_* or RTE_* parameter values. If you modify these values, you will need to rebuild the runtime environment by going through the configuration process again, as the runtime members, jobs, and runtime data sets may need to be modified or re-allocated, depending upon what you are changing. The configuration profiles must also be refreshed. For more information, see the runtime environment maintenance scenarios in the *IBM Tivoli OMEGAMON XE and Tivoli Management Services: PARMGEN Reference*. The first Set Up PARMGEN Work Environment for an RTE panel (KCIP@PG1) is displayed:
3. Specify values for the required parameters.

**RTE profile library and RTE member name**

Specify the runtime environment profile library and runtime environment member name, if applicable. The following options are available:

- If creating a brand new RTE, leave this field blank; or
- If creating another new RTE and you want to clone a PARMGEN-created RTE's configured product set, specify the WCONFIG profile library and RTE member name to clone (for example: `&hlq.&rte.WCONFIG(&clone_from)`); or
- If reconfiguring or upgrading this existing DEMO RTE, specify its values (TDITN.IDST.RTE.WCONFIG(DEMO)); or
- If converting an ICAT-created RTE to PARMGEN mode, specify the ICAT RTE Batch member location and RTE member (for example: `&hlq.INSTJOBS(DEMO)`).

Specify the Install Job Generator (JOBGEN) output library if you want PARMGEN to reuse CALLLIBS parameters from the JOBGEN repository:

```
**==
(Type ? for last referenced JOBGEN library discovered, if any.)
```

Enter Jobcard data:
```
==> //&USER.A JOB (ACCT),'&SYSMEMBER' - NAME',CLASS=A, ______________________
==> /* MSGCLASS=X,MSGLEVEL=(1,1),NOTIFY=&SYSUID,,REGION=OM ______________________
==> //** RTE_NAME=&RTE_NAME% ________________________________________________
==> //** SYSJOBNAME=&SYSJOBNAME% SYSMEMBER=&SYSMEMBER% _________________
```

**Figure 11. Set up the PARMGEN work environment for a runtime environment (1 of 3)**

**Install Job Generator (JOBGEN) output library**

Specify the Install Job Generator (JOBGEN) output library if you want PARMGEN to reuse parameter values from the JOBGEN output library repository. These are values such as the jobcard, CALLLIBS override data, and other CSI values such as the high-level qualifier of the SMP/E target (TK*) datasets. If the specified dataset does not exist, you are prompted to correct the name.

**Note:** For more information about the JOBGEN output library, refer to the Installation Job Generator Utility section in the product's Program Directory.

**Jobcard data**

If a customized jobcard is already available, it is retrieved from the ISPF user profile pool. If it is not available and if you did not specify a JOBGEN output library, the jobcard is harvested from `gbl_target_hilev.TKANSAM` SMP/E target library where the initial PARMGEN
sample jobcard default is supplied. If you specified a JOBGEN output library, the jobcard information is harvested from that location. You can modify the retrieved data as needed. The customized jobcard is saved in the ISPF user profile pool and persists across ISPF sessions. On this panel, specify the parameter values appropriate for your environment. In most cases, the PARMGEN configuration parameters on this panel are required.

4. **After all required parameters are specified, press Enter to move to the second setup panel** ([Figure 12](#)).

---

**Figure 12. Set up the PARMGEN work environment for a runtime environment (2 of 3)**

---

<table>
<thead>
<tr>
<th>Command</th>
<th>SET UP PARMGEN WORK ENVIRONMENT FOR AN RTE (2 OF 3) -----------</th>
</tr>
</thead>
<tbody>
<tr>
<td>GBL_USER_JCL:</td>
<td>TDITN.IDTST.PARMGEN.JCL</td>
</tr>
<tr>
<td>RTE_PLIB_HILEV:</td>
<td>TDITN.IDTST</td>
</tr>
<tr>
<td>RTE_NAME:</td>
<td>DEMO</td>
</tr>
</tbody>
</table>

Enter parameter values appropriate for your environment:

| GBL_INST_HILEV: | __________ |
| HLQ of ICAT INSTLIB/INSTJOBS installation datasets |

| GBL_TARGET_HILEV: | __________ |
| HLQ of SMP/E target (TK*) datasets |

| GBL_SYSDA_UNIT: | __________ |
| Non-VSAM disk UNIT (global work datasets) |

Note: Type BACK to go back one panel.

---

**GBL_INST_HILEV**

If you have already created runtime environments with the Configuration Tool and you plan to convert those runtime environments into PARMGEN mode, specify the high-level qualifier of the INSTJOBS ICAT installation jobs library. The INSTJOBS library is where the Configuration Tool batch runtime environment member is stored. This member is used for the Configuration Tool-to-PARMGEN conversion (using the KCIJPCNV conversion job).

If you have not created runtime environments with the Configuration Tool, you are not required to provide a value for this parameter.

By default, this field is prepopulated with the runtime environment profile library field value on the previous panel, if you customized that field.

**GBL_TARGET_HILEV**

Specify the SMP/E high-level qualifier of the SMP/E target (TK*) libraries.

**GBL_SYSDA_UNIT**

Specify the non-VSAM disk UNIT for global work data sets.

---
6. After all required parameters have been specified, press Enter to move to the third setup panel
(Figure 13).

![KCIP0PG3 - SET UP PARMGEN WORK ENVIRONMENT FOR AN RTE (3 OF 3) ---------]

Command ===> Scroll ===> PAGE

Enter parameter values appropriate for your RTE=DEMO.

Note: See F1=Help for SMS-related and RTE HLQ-related considerations
when VOLUME, UNIT, STORCLAS, and MGMGTCLAS parameters are required for
the global RTE_* parameters and the Kpp_* product-specific parameters.

More: +

RTE_SMSPOSE_FLAG: Y (POSE flag (Y, N))
RTE_SMSUNIT: ________ (Non-VSAM disk UNIT type)
RTE_SMSVOLUME: ________ (Non-VSAM disk VOLSER)
RTE_SMSSTORCLAS: ________ (Non-VSAM disk STORCLAS)
RTE_SMSVSMAVOLUME: ______ (VSAM disk VOLSER)
RTE_SMSVSMAMGMTCLAS: ________ (VSAM disk MGMGTCLAS)
RTE_SMSVSMASTORCLAS: ________ (VSAM disk STORCLAS)
RTEHILEV: TTEST.&userid__________ (Non-VSAM prod. RK* HLQ)
RTEVSMHILEV: TTEST.&userid__________ (VSAM production RK* HLQ)
RTEVOLUME: __________ (VSAM disk VOLSER)
RTEVMAMGMTCLAS: ________ (VSAM disk MGMGTCLAS)
RTEVMASTORCLAS: ________ (VSAM disk STORCLAS)

Note: Type BACK to go back one panel.

Figure 13. Set up the PARMGEN work environment for a runtime environment (3 of 3)

The values specified on this panel are inherited by the corresponding parameters in the LPAR
configuration profile.

Notes:

- If you intend to copy the customized WCONFIG members (Kpp$C*, Kpp$P*/Kpp$P*, Kpp$S*),
$JOBCARD, and $GBLSUSR) from another runtime environment, do not need to make any
changes to these parameters, as the values will be overwritten. Change the corresponding
parameters in the LPAR configuration profile after you have cloned the members.

- If you are using a SMS-managed high-level qualifier index, VOLUME, UNIT, STORCLAS, and
MGMGTCLAS parameters for the global RTE_* parameters and the Kpp_* product-specific
parameters are not required. If you are using non-SMS-managed RTE_HILEV and
RTE_VSAM_HILEV high-level qualifiers, the RTE_SMS_VOLUME, RTE_SMS_VSAM_VOLUME
and RTE_SMS_UNIT values are required, as well as the product-specific Kpp_* VOLUME, UNIT,
STORCLAS, and MGMGTCLAS equivalent parameters. (Certain products allow for
product-specific Kpp_* VOLUME, UNIT, STORCLAS, and MGMGTCLAS parameters for their
specific dataset needs.) Review all *_VOLUME and other SMS and non-SMS parameters in the
WCONFIG(rte) LPAR configuration profile.

7. Specify the following parameters as appropriate. Depending upon your screen resolution, you may
have to scroll down (PF8) to see all the parameters. (Note the More: + indicating that there is
additional content on the panel.)
• If the new runtime environment will share libraries with other runtime environments, set the RTE_TYPE parameter to SHARING.
  – To create a sharing-with-base environment, provide a name for the shared runtime environment using RTE_SHARE and the high-level qualifier for the shared environment using RTE_X_HILEV_SHARING. The configuration software will create the base runtime environment.
  – To share with the SMP/E target libraries, specify SMP for RTE_SHARE.
  – To create a sharing-with-full environment, provide the name and high-level qualifier.
• If the runtime environment you specify has not yet been created, the configuration software will create the libraries for the full environment you have yet to create. If the runtime environment will have system variables enabled, you can use symbolics, so that you will not have to replace started tasks across runtime environments.
• If you are going to use variables, set any occurrence of SYSG to &SYSNAME. (For example, set RTE_TEMS_NAME_NODEID to PLB3&SYSNAME.:CMS.)
• If required by your site conventions, use RTE_STC_PREFIX to specify a prefix of up to 4 characters to replace the default CANS prefix for the names of started tasks created by the configuration software. If you choose to use symbolics, you can use up to 18 characters that resolve to 4 characters. For example, &SYSCLONE.
• If required by your site conventions, use RTE_VTAM_APPLID_PREFIX to specify a prefix of up to 4 characters (or a symbolic of up to 18 characters that resolve to 4) for all VTAM major nodes and applids. For example, TS&SYSCLONE.
• Set the RTE_SYSV_SYSVAR_FLAG parameter to Y to enable system variables.
• Set the following parameters as applicable:
  RTE_SMS_PDSE_FLAG
  Specify whether or not (Y or N) partitioned data sets are to be allocated as PDSE. The default is Y.
  RTE_SMS_UNIT
  Specify the Non-VSAM disk UNIT type to be used for data set allocation.
  RTE_SMS_VOLUME
  Specify the Non-VSAM disk VOLSER to be used for data set allocation.
  RTE_SMS_MGMTCLAS
  Specify the Non-VSAM disk MGMTCLAS to be used for data set allocation.
  RTE_SMS_STORCLAS
  Specify the Non-VSAM disk STORCLAS to be used for data set allocation.
  RTE_SMS_VSAM_VOLUME
  Specify the VSAM disk VOLSER to be used for data set allocation.
  RTE_SMS_VSAM_MGMTCLAS
  Specify the VSAM disk VOLSER to be used for data set allocation. Specify the VSAM disk MGMTCLAS to be used for data set allocation.
  RTE_SMS_STORCLAS
  Specify the VSAM disk STORCLAS to be used for data set allocation.
  RTE_HILEV
  Specify the non-VSAM high-level qualifier that you want to use for the allocation of the non-VSAM production runtime (RK*) libraries. By default, the initial value for a brand new runtime environment to be created by using the PARMGEN process is the same value as the RTE_PLIB_HILEV value that you specified on the PARAMETER GENERATOR (PARMGEN) WORKFLOW - WELCOME panel.
  RTE_VSAM_HILEV
  Specify the VSAM high-level qualifier that you want to use for the allocation of the VSAM
production runtime (RK*) libraries. By default, the initial value for a brand new runtime environment to be created by using the PARMGEN process is the same value as the RTE_PLIB_HILEV value you specified on the Parameter Generator (PARMGEN) Workflow - Welcome panel.

**RTE_TYPE**
Specify the type of runtime environment being created. The following types are valid:

- **Full**
  Allocates both private and base libraries. Use this if you have only one runtime environment or if you have a runtime environment for a unique set of products.

- **Sharing**
  Allocates private libraries only. This type can either share base libraries with a base or full runtime environment, or use target libraries that were installed by SMP/E for its base libraries. Define one sharing runtime environment for each z/OS image.

**Note:** In PARMGEN, base libraries are created as part of creating the first runtime environment that shares the base. Subsequently created runtime environments can point to an existing base using the RTE_SHARE parameter.

**RTE_SHARE**
Specifies the name of the Base or Full runtime environment from which the sharing runtime environment obtains its base library information. If target libraries that were installed by SMP/E are to be shared, provide the value SMP. Required for a sharing runtime environment.

**RTE_X_HILEV_SHARING**
Specifies the value of the non-VSAM high level qualifier of the runtime environment being shared to.

**RTE_TEMS_CONFIGURED_FLAG**
Specifies whether or not a Tivoli Enterprise Monitoring Server is to be configured in this runtime environment. Required for a full or sharing runtime environment that contains a monitoring server. If you are creating a runtime environment for stand-alone monitoring agents without a monitoring server, specify N. Otherwise, specify Y (the default) to allocate libraries for the monitoring server.

**RTE_TEMS_NAME_NODEID**
Specifies the unique name that identifies the monitoring server for internal processing. Required for a full or sharing runtime environment that contains a monitoring server.

**RTE_STC_PREFIX**
For full and sharing runtime environments, specify a prefix to be used when generating started task procedures for products configured in the runtime environment. The default value of CANS is provided, but specifying your own prefix prevents confusion with jobs generated by other runtime environments on the same system.

**RTE_VTAM_APPLID_PREFIX**
Specifies the a VTAM applid prefix that contains system variables, if system variables are enabled and you intend to use SNA communications. The resolved prefix can have a maximum of 4 characters. Be sure to place a period (.) after the last symbolic in the specification. If none of the products or components you intend to configure require SNA communications, delete the default value and leave this parameter value blank.

**RTE_SYSV_SYSVAR_FLAG**
Determines whether or not to enable z/OS system variables, which are elements that allow systems to share PARMGEN definitions while retaining unique values in those definitions. If you use system variables, the components inherit the system values for the system on which they are started (the host z/OS system). These system-specific values are then automatically loaded into dynamic in-memory parameter members that exist only
while the component runs. The result is that the software runs correctly by using the system-specific parameter values for the host z/OS system.

**RTE_SECURITY_USER_LOGON**

Security system to be used for the runtime environment. If you specify a security system, verify that it is installed and configured correctly for your site. Specify one of the following values: RACF, ACF2, TSS, NAM, SAF, or None. If you specify ACF2, you must also provide the name of the ACF2 macro library as the value of the `GBL_DSN_ACF2_MACLIB` parameter. The System Authorization Facility (SAF) provides a generic API to interface to z/OS security software.

Specifying a security system here indicates which system will be used for security validation of users signing on to the Tivoli Enterprise Portal, but it does not enable validation. Security validation of users is enabled by the `KDS_TEM_Security_KDS_VALIDATE` parameter.

**RTE_SECURITY_FOLD_PASSWORD_FLAG**

Determines whether mixed case passwords are supported and being used. Required for full and sharing runtime environments.

By default, logon passwords are folded to uppercase. However, IBM RACF V1.7 and later supports mixed-case passwords. If you want to implement mixed-case passwords and if all your monitoring agents support them, set this field to N. If any of your monitoring agents do not support mixed-case passwords, do not activate the SETROPTS PASSWORD(MIXEDCASE) option in RACF and do not enable mixed-case passwords in your runtime environments. Accept the default value of Y for this parameter.

**RTE_SECURITY_CLASS**

Specifies a common System Authorization Facility (SAF) security class name for OMEGAMON enhanced 3270 user interface security controls. The individual products have additional SAF security settings that are specific to them (for example, how to secure product-specific Take Action requests). See the product-specific documentation for information.

**RTE_X_SECURITY_EXIT_LIB**

Specifies the name of the global runtime environment library that houses all the OMEGAMON and Tivoli Management Services related product security exits (such as KOBSSUPDT OMEGAMON KppSUPDI exits, TMS:Engine security exits, external security exits). This value overrides the SYSIN DD statement where the user exits may have been customized (if other than the default RKANSAM location). The KppJPS3 input members to the composite KCIIJPS3 security job point the SYSIN DD statement to RKANSAM by default. If you need to make further changes to the sample exit, copy the user exit to the xKANSAMU library, and make your changes accordingly. Then modify the `RTE_X_SECURITY_EXIT_LIB` parameter and change the value to RKANSAMU instead.

If you are converting a runtime environment built by the Configuration Tool, the security exit source library varies for each product. Refer to the application-specific security jobs for more information.

**GBL_DSN_ACF2_MACLIB**

Specifies the name of an ACF2 macro library. This parameter is required if the value specified for the `RTE_SECURITY_USER_LOGON` environment variable is ACF2. To include more than one ACF2 macro library, define more than one `GBL_DSN_ACF2_MACLIBn` parameter, with a unique number substituted for n at the end of each parameter name (example: `GBL_DSN_ACF2_MACLIB1`, `GBL_DSN_ACF2_MACLIB2`, and so on).

**KDS_TEM_Security_TYPE**

Specifies whether to configure a hub monitoring server or a remote monitoring server.
KDS_TEMS_HA_TYPE

Specify whether to configure the Tivoli Enterprise Monitoring Server as a high-availability hub. This configuration requires a sysplex environment with dynamic virtual IP addressing (DVIPA) and shared DASD. A high-availability hub is configured in its own runtime environment, without monitoring agents, and can be configured on the same LPAR with a remote monitoring server.

8. After you have specified all required parameters, press Enter. The Display PARMGEN Environment Analysis panel (KCIP@PG4) is displayed.

```
KCIP@PG4------------- DISPLAY PARMGEN ENVIRONMENT ANALYSIS --- Row 1 to 6 of 6
Command ===> Scroll ===> PAGE

Review message traffic before proceeding. These Kpp components are installed into the TDITNT.DEV.1TM62351.TKANCUS SMP/E target library and are available for (re)configuration and/or upgrade into the RTE=MYRTE. Refer to the next panel ("EXCLUDE PRODUCTS FROM PARMGEN CUSTOMIZATION") for a legend of the Kpp components' product names.

34 Components installed in TDITNT.DEV.1TM62351.TKANCUS:
KAH KC2 KC5 KDF KD0 KDS KD2 KD4 KDS KGW
KHL KI2 KI5 KMC KMQ KM2 KM5 KNA KN3
KOB KON KQ1 KRG KRJ KRK KRN KRV KRW
KS3 KT1 KWO KYN
End of data

```

Figure 14. Display PARMGEN environment analysis

This panel displays information about the SMP/E environment and the products available for configuration and upgrade. If you are cloning an existing PARMGEN environment, or converting a Configuration Tool environment, this panel will also display the products configured in that environment, and the products that are installed but not configured in the new runtime environment.

9. Review the SMP/E message traffic before proceeding, then press Enter to proceed to the next panel. The Exclude Products from PARMGEN Customization panel (KCIP@PG5) is shown.
This panel lists the products that are installed in the CSI target zone and therefore available for configuration. The products are listed by product code (Kpp) and product name and version. If the list exceeds screen size, you can scroll down.

10. To exclude any of the products from the runtime environment that you are configuring, place an X before the product code.

If you specified a Configuration Tool batch parameter member to convert from or an existing PARMGEN WCONFIG profile to clone on the Set Up PARMGEN Work Environment for an RTE (1 OF 3) panel, any products already configured within that member are flagged with an asterisk following the product code. If an existing runtime environment configuration profile has been specified, the first line in the list of products specifies ALL as the product code and an asterisk marks all products already configured in the referenced environment. Place an X next to ALL to exclude all products not already configured in the environment.

11. Press Enter to generate the gbl_user_jcl(KCIJPCFG) job tailored with the values you supplied.

12. Review this job and submit it. After you return to the main Workflow panel, you will note a > before option 1 to indicate that this is the last step you performed. The status and date fields for that option are updated.

**What to do next**

Before you complete the remaining steps in the configuration process, you might want to review the index of PARMGEN jobs that must or can be submitted to carry out the configuration.
Step 2. (Optional) Review the PARMGEN job index

The $JOBINDX is an index of jobs used by PARMGEN to create the runtime environment. The names and descriptions of the jobs are presented in the order in which they should be submitted. You can review this file at any time for planning purposes.

About this task

Procedure

1. On the Workflow main panel, select option 2 (Review PARMGEN job index).

The job index is displayed:

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1&gt;</td>
<td>KCIJPFCFG Set up PARMGEN work environment for an RTE.</td>
</tr>
<tr>
<td>2.</td>
<td>$JOBINDX Review PARMGEN job index.</td>
</tr>
<tr>
<td>3.</td>
<td>KCIJPGCF Clone customized WCONFIG members.</td>
</tr>
<tr>
<td>4.</td>
<td>KCIJPUP1 Update interim libraries and create profiles.</td>
</tr>
<tr>
<td>5.</td>
<td>KCIJPMC1 Merge profile from backup RTEDEMO.</td>
</tr>
<tr>
<td>6.</td>
<td>KCIJPMC2 Merge profile from model RTE.</td>
</tr>
<tr>
<td>7.</td>
<td>KCIJPCNV Convert an ICAT RTE Batch member.</td>
</tr>
<tr>
<td>8.</td>
<td>Customize PARMGEN configuration profiles.</td>
</tr>
<tr>
<td>9.</td>
<td>KCIJPVAL Validate PARMGEN profile parameter values.</td>
</tr>
<tr>
<td>10.</td>
<td>Create the RTE members and jobs.</td>
</tr>
<tr>
<td>11.</td>
<td>SUBMIT: Submit batch jobs to complete PARMGEN setup.</td>
</tr>
<tr>
<td>U</td>
<td>Utility: Access PARMGEN utilities.</td>
</tr>
<tr>
<td>R</td>
<td>New RTE: Reset RTE, Status, and Date fields.</td>
</tr>
</tbody>
</table>

Note: Enter n (1-11) to perform tasks. Enter ns (1s-11s) for detailed job/task status.

Status Date

RC= 00000 2012/10/11

The job index is displayed:
2. Review the index, then press F3 to return to the main panel.

Step 3. (Conditional) Clone customized WCONFIG members

If you are using the same global system procedure libraries, VTAM lists, and the like for your runtime environments, or if you have made customizations to WCONFIG override imbeds, to the $JOBCARD member, or the global PARMGEN configuration profile members such as $GBL$IBM or $GBL$USR for a previous runtime environment that you want to include in a new runtime environment, you can clone those members for a new runtime environment. Cloning existing WCONFIG members means that you do not have to configure those files again in the new environment. You might also clone the WCONFIG members as part of cloning an entire runtime environment, including the LPAR configuration profile.

Before you begin

The name of the WCONFIG profile library and RTE member to clone should have been specified on the Set up PARMGEN Work Environment for an RTE (1 of 3) panel (KCIP@PG1).
**Procedure**

1. From the Workflow main menu, select option 3.

```plaintext

KCIP@PG0 ------ PARAMETER GENERATOR (PARMGEN) WORKFLOW
Option ===> 3 Scroll ===> PAGE
Enter PARMGEN parameter values appropriate for your environment:

- GBL_USER_JCL: TDITN.IDTST.PARMGEN.JCL
  (PARMGEN common/global library for RTEs (CONFIG DD lib. in STCs)
- RTE_PLIB_HILEV: TSTEST.CCAPI
  High-Level Qualifier (HLQ) of work libraries (IK*,WCONFIG,WK*)
- RTE_NAME: DEMO (Type ? for a list of configured RTEs)
  Runtime environment (RTE) name for this LPAR

Note: Enter n (1-11) to perform tasks. Status Date
Enter ns (1s-11s) for detailed job/task status. --------- ----------
1. KCIJPFCFG Set up PARMGEN work environment for an RTE. RC= 00000 2012/12/14
2. $JOBINDEX Review PARMGEN job index. Viewed 2012/12/14
3. KCIJPCCF Clone customized WCONFIG members. (COND)
4. KCIJPUP1 Update interim libraries and create profiles.
5. KCIJPMC1 Merge profile from backup PLB1SP22 (COND)
6. KCIJPMC2 Merge profile from model RTE. (COND)
7. KCIJPCNV Convert an ICAT RTE Batch member. (COND)
8> PLB1SP22 Customize PARMGEN configuration profiles.
9. KCIJPVAL Validate PARMGEN profile parameter values.
10. $PARSE Create the RTE members and jobs.
11. SUBMIT Submit batch jobs to complete PARMGEN setup.
U Utility Access PARMGEN utilities. (Optional)
R New RTE Reset RTE, Status and Date fields. (Optional)
```

The KCIJPCCF job is presented.

2. If necessary, change the library concatenated in the //OLDWCNFG DDNAME. By default, this DDNAME is already set to the WCONFIG library you specified in the KCIP@PG1 panel. You can specify additional members to clone by using the customizing the * SELECT MEMBER=(&MBR1,&MBR2*) statement.

3. Submit the job and return to the main Workflow panel.

**Results**

The files are copied to the WCONFIG library of the new runtime environment.

**What to do next**

Update the interim libraries and create the configuration profiles. If you are cloning a model runtime environment, you can then merge the values from the LPAR configuration profile from the model environment into the LPAR profile for the new environment using the KCIJPMC2 merge job.

If you are cloning a runtime environment that has OMEGAMON XE on z/OS configured and the monitoring server is the sysplex proxy (KM5_SYSPLEX_PROXY_POSITION=PRIMARY), ensure that the KM5_SYSPLEX_PROXY_POSITION for the monitoring server in any cloned runtime environments is set to BACKUP.

**Step 4. Update the interim libraries and create profiles**

This step generates and submits the KCIJPUP1 job. The KCIJPUP1 job has two functions. First, it populates (or updates, if this is an existing runtime environment) the IK* staging libraries with the product-specific PARMGEN elements from the SMP/E target libraries. Second, it prepares the various PARMGEN elements based on the customizations made in Step 1 (the KCIJPFCFG job).
About this task

For more information about the job and the members and jobs, see online help for this option (4. Update interim libraries and create profiles).

Procedure

1. From the Workflow main menu, select option 4.

```
Option ===> Scroll ===> PAGE

Enter PARMGEN parameter values appropriate for your environment:

GBL_USER_JCL: TDIN.IDST.PARMGEN.JCL
PB Parmgen parameter generator for RTEs (CONFIG DD lib. in STCs)
RTE_LIB_HILEV: TSTEST.&userid
High-Level Qualifier (HLQ) of work libraries (IK*,WCONFIG,WK*)
RTE_NAME: DEMO_ (Type ? for a list of configured RTEs)
Runtime environment (RTE) name for this LPAR

Note: Enter n (1-11) to perform tasks. Status Date
Enter ns (1s-11s) for detailed job/task status. --------- ----------
1. KCIJPCFG Set up PARMGEN work environment for an RTE. RC= 00000 2012/10/11
2. $JOBINDX Review PARMGEN job index. Viewed 2012/10/11
3> KCIJPCCF Clone customized WCONFIG members. (COND) RC= 00000 2012/10/11
4. KCIJPUP1 Update interim libraries and create profiles.
5. KCIJPMCl Merge profile from backup RTEDEMO. (COND)
6. KCIJPMC2 Merge profile from model RTE. (COND)
7. KCIJPCNV Convert an ICAT RTE Batch member. (COND)
8. Customize PARMGEN configuration profiles.
9. KCIJPVAL Validate PARMGEN profile parameter values.
10. Create the RTE members and jobs.
11. SUBMIT Submit batch jobs to complete PARMGEN setup.
U Utility Access PARMGEN utilities. (Optional)
R New RTE Reset RTE, Status and Date fields. (Optional)
```

The KCIJPUP1 job is presented.

2. Review the notes, then submit the job. On completion, you should receive a good return code. Press F3 to return to the main panel.

Steps 5 and 6. (Conditional) Merge environment configuration parameters from backed up or model profiles

The KCIJPMCl and KCIJMC2 jobs merge already customized values from an existing runtime environment into a newly created profile. You can merge values from both LPAR and global profiles.

About this task

There are several scenarios in which you use the merge jobs:
1. Converting an existing runtime environment created using the Configuration Tool to PARMGEN.
   You merge the values from the converted profile to the one created for the new environment. Any new parameters that were not present in the existing runtime environment are preserved, and the values from the existing environment are added.

2. Upgrading an environment.
   In this scenario, you first rename the LPAR profile to preserve its customized parameter values, create a new profile that incorporates any new parameters, then merge the values from the backed-up profile into the new one.

3. Applying maintenance.
In this scenario, the steps are the same as in the upgrade scenario. You first rename the LPAR profile to preserve its customized parameter values, create a new profile that incorporates any new parameters, then merge the values from the backed up profile into the new one.


In this scenario, you use the values from a model runtime environment on another LPAR to create a new runtime environment that uses the same product set and parameter values, with LPAR-specific customization.

The KCIPJPMC1 job is used for the first three scenarios. The KCIPJPMC2 job is used for the fourth scenario.

In scenarios 2 and 3, you rerun the KCIPJCFG job before you run the KCIPJPMC1 job. The KCIPJCFG job prompts you to name and back up the LPAR profile, and automatically runs the KCIPJUP1 and KCIPJPMC1 jobs.

For scenarios using the KCIPJPMC1 and KCIPJPMC2 jobs, see the IBM Tivoli OMEGAMON XE and Tivoli Management Services on z/OS: PARMGEN Reference.

**Procedure**

From the Workflow main menu:
- Select option 5 if you want to merge parameters from a converted Configuration Tool batch parameter member, a backed up version of the PARMGEN LPAR configuration profile for the same runtime environment, or the $GBL$USR profile.
- Select option 6 if you want to merge parameters from a model runtime environment on another LPAR.

**What to do next**

Make any necessary changes to the configuration profiles, such as specifying LPAR-specific values.

---

**Step 7. (Conditional) Convert a Configuration Tool runtime environment batch member**

You can use the properties and parameters from an existing runtime environment created using the Configuration Tool to create a new PARMGEN runtime environment. PARMGEN includes a job that converts Configuration Tool batch parameter members so that the settings are applied and serve as input to either the $PARSE (if the runtime environment is not enabled for system variables) or the $PARSESv (if the runtime environment is enabled for system variables) file-tailoring job in the new PARMGEN-configured runtime environment.

**About this task**

This step is applicable only if you are converting an existing runtime environment created using the Configuration Tool. After you have converted the runtime environment to PARMGEN, you cannot use the Configuration Tool to manage the runtime environment any more unless you change the high-level qualifiers for the new runtime environment.

For complete, end-to-end instructions for converting a Configuration Tool environment, see Scenario PGN05 in the IBM Tivoli OMEGAMON XE and Tivoli Management Services on z/OS: PARMGEN Reference.

**Procedure**

1. If one does not already exist, create a batch mode parameter member with the existing Configuration Tool-configured runtime environment. See Chapter 8, “Using the Configuration Tool batch mode to replicate a configured environment,” on page 179 for details on creating a batch mode parameter member. If you want the new PARMGEN
runtime environment to use different high-level qualifiers than the Configuration Tool environment, change them in the batch parameter deck before you run the conversion program. Changing the high-level qualifiers leaves the Configuration Tool environment available for staged upgrades.

To change the qualifiers, edit the batch parameter member and do a change all to change the RTE high-level qualifiers to the new qualifiers. For example, RTE_HILEV, RTE_VSAM_HILEV and variants.

**Important:** If you are converting a sharing type environment, you must also change the high-level qualifiers for the shared environment (base or full).

2. From the Workflow main panel, select option 7. The JCL for the KCIJPCNV job is displayed.

3. Review the instructions and notes in the job. Pay particular attention to the sections on specifying additional runtime environments to convert in the PLBIN DDNAME and customizing parameters in the SYSTSIN DDNAME.

4. Submit the job. On completion, you should receive a good return code.

**Results**

The properties and parameters of the converted runtime environments are ready for use as input to the $PARSE or $PARSESV file tailoring jobs for the new PARMGEN runtime environment.

**What to do next**

Now you need to customize the configuration profiles. If you have another runtime environment whose customizations you would like to use, you can merge the configuration profile from that environment into the new configuration profile.

**Step 8. Customize PARMGEN configuration profiles**

In this step, you customize the parameters and settings for each of the components that you are deploying in this runtime environment, which may include a Tivoli Enterprise Monitoring Server, OMEGAMON enhanced 3270 user interface, and any OMEGAMON monitoring agents.

**About this task**

PARMGEN configuration uses three profiles:

%RTE_NAME%

LPAR-specific configuration file. This file contains parameters that define the configuration for all the shared components of the infrastructure and for each product configured in this runtime environment. This profile is named for the runtime environment and is stored in the WCONFIG data set.

$GBL$USR

Configuration file for global system libraries. Review and possibly customization of this profile is required. This profile is stored in the WCONFIG data set. This file can be copied to other runtime environments if these global system libraries are typically the same across LPARs.

Review of the $GBL$USR profiles is required under either of the following conditions:

- if the runtime environment is being upgraded and there may be new GBL_* parameters introduced in the new version of the product. In the case of a conversion, these parameters may not have been in the original Configuration Tool batch member supplied, as the batch member may still be at the previous version of that product that is now being converted into PARMGEN, and upgraded at the same time.
- to customize product-specific GBL_* common or global parameters that may apply to more than one product or apply to several components within a product family (for example, GBL_DB2_KD2_CLASSIC_STC parameter applies to all OMEGAMON XE for DB2 PE/PM components if configuring this product family).
System variables configuration file. This file is stored in the data set specified by the
GBL_USER_JCL parameter. Customization of the system variables configuration file is required if
the runtime environment is enabled for system variables (RTE_SYSV_SYSVAR_FLAG=Y) and
user-defined symbols are being used. In addition, if variable overrides are enabled in the runtime
environment configuration profile (RTE_X_SYSV_OVERRIDE_SYMBOLS=Y), you are required to
define all types of symbolics in this member and what the symbolics resolve to, regardless if this
is a system, a KCIPARSE-extracted, or a user-defined symbol.

Default values are provided for all required parameters and some optional ones. If you do not want to
customize these parameters, and you do not want to enable optional features, you can complete the
configuration by accepting these defaults. Alternatively, you can specify custom values. You can also
specify custom values for optional parameters that have no defaults.

Complete this step in conjunction with the product-specific planning and configuration guides and the
PARMGEN Reference guide to configure your components correctly.

**Tips:** While editing the PARMGEN configuration profiles, you may have a large number of parameters to
review. It may be useful, therefore, to EXCLUDE certain PARMGEN parameters while you focus on a
particular configuration task. For example, EXCLUDE all PARMGEN parameters that are for started tasks.
Then FIND all occurrences of these parameters so you can quickly do a CHANGE ALL command. PARMGEN
provides a macro for this task; see “XF edit macro” on page 243

For help with the parameter values, consult the following resources:

- Comments in the profiles
- Online help for parameters (see “KCIRPLBS online help macro” on page 243 for more information)
- [PARMGEN technote](http://www.ibm.com/support/docview.wss?uid=swg21417935)
- Scenarios in *IBM Tivoli OMEGAMON XE and Tivoli Management Services on z/OS: Common Parameter
  Reference*
- Parameter Reference for each product

**Procedure**

1. From the Workflow main panel, select option 8.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>KC1JPCFG Set up PARMGEN work environment for an RTE. RC= 00000 2012/10/11</td>
</tr>
<tr>
<td>2.</td>
<td>$JOBINDX Review PARMGEN job index. Viewed 2012/10/11</td>
</tr>
<tr>
<td>3.</td>
<td>KC1JPCCF Clone customized WCONFIG members. (COND)</td>
</tr>
<tr>
<td>4&gt;</td>
<td>KC1JPUPU Update interim libraries and create profiles. RC= 00000 2012/10/11</td>
</tr>
<tr>
<td>5.</td>
<td>KC1JPMCI Merge profile from backup RTEDEMOR (COND)</td>
</tr>
<tr>
<td>6.</td>
<td>KC1JPMC2 Merge profile from model RTE. (COND)</td>
</tr>
<tr>
<td>7.</td>
<td>KC1JPCNV Convert an ICAT RTE Batch member. (COND)</td>
</tr>
<tr>
<td><strong>8. DEMO</strong></td>
<td>Customize PARMGEN configuration profiles.</td>
</tr>
<tr>
<td>9.</td>
<td>KC1JPVAL Validate PARMGEN profile parameter values.</td>
</tr>
<tr>
<td>10.</td>
<td>Create the RTE members and jobs.</td>
</tr>
<tr>
<td>11.</td>
<td>SUBMIT Submit batch jobs to complete PARMGEN setup.</td>
</tr>
<tr>
<td>U</td>
<td>Utility Access PARMGEN utilities. (Optional)</td>
</tr>
<tr>
<td>R</td>
<td>New RTE Reset RTE, Status and Date fields. (Optional)</td>
</tr>
</tbody>
</table>
The Customize PARMGEN Configuration Profile Members (KCIP@PG6) panel is displayed.

---

```
KCIP@PG6 ------ CUSTOMIZE PARMGEN CONFIGURATION PROFILE MEMBERS ---------------
OPTION ===>
(Required)* Select option 1 to customize the DEMO RTE LPAR profile
1. DEMO RTE LPAR CONFIG profile in WCONFIG

(Conditional)* Select option 2 and/or 3 if applicable to this RTE:
2. $GBLSUSR Global parameters CONFIG profile in WCONFIG
   (Required if this is not an ICAT-to-PARMGEN conversion)
3. DEMO System Variables CONFIG profile in GBL_USER_JCL
   (TSTEST.&userid.PARMGEN,JCL)
   (Required if using user-defined symbols or overriding system symbols' resolved values - see F1=Help)

*Note: The PARMGEN configuration profiles above are preserved
   (initially created by KCIJUP1 job).

(Reference) IBM-supplied default profiles (refreshed by KCIJUP1 job):
4. $CFG$IBM IBM default RTE LPAR CONFIG profile in WCONFIG
5. $GBLSIBM IBM default Global parameters CONFIG profile in WCONFIG
6. $SYSIN SPARSE/SPARSESV SYSIN controls for processing which:
   - CONFIG profiles (CONFIG MEMBER=&config_profile)
   - runtime members (SELECT MEMBER=(*,&mbr1,&mbr2??))
   to (re)create from PARMGEN IX*-to-WK* output libraries.

(Optional) Select option 7 for member list of the WCONFIG library:
7. WCONFIG TDITN.IDTST.DEMO.WCONFIG

Enter=Next F1=Help F3=End/Cancel
```

---

Figure 16. Customize PARMGEN configuration profile members

2. Configure the profile members as required.

   The configuration involves several substeps. The substeps that you must take are dependent on the runtime environment that you are creating and the amount of customization you want to make to default parameters. The first substep is required for all configurations. The remaining topics in this section describes a basic set of customizations users typically want to make to their runtime environment.

**Customize the LPAR configuration profile**

The LPAR configuration profiles contains the parameter values for a runtime environment and the products and components configured in it. You customize this member based on site-specific values.

**About this task**

The LPAR configuration profile contains two types of parameters:

- RTES/RTE_* parameters, which specify global runtime environment values
- Kpp$/ or Kpp_* parameters, which specify product-specific runtime environment values.

The profile also contains new Kpp_X_* parameters, where Kpp is the three-letter product code for a component or product. In the Configuration Tool, these were hardcoded settings that were not externalized.

**Procedure**

To edit the profile, select option 1 from the Customize the PARMGEN Configuration Profile Members (KCIP@PG06) panel.
The LPAR profile is displayed.

What to do next

Complete the following tasks as appropriate.

Update the USER PROLOG
The configuration profile contains a prolog section in which you can log changes you make in this member for future reference.

About this task

Procedure

Perform a FIND on USER PROLOG to locate this section and update it accordingly. Figure 17 on page 88 shows an example of this section in the LPAR profile for the DEMO runtime environment.
Verify the list of products to be configured within this runtime environment

The configuration profile contains a list of products and components to be configured as part of this runtime environment. This list should match with the list of products that you defined when you set up the work libraries for the runtime environment. If any products are added or removed from this runtime environment after initial configuration, update this section accordingly.

Procedure

Review the list of products in the CONFIGURE PRODUCTS section and set the flags as required. Figure 18 shows an example of this section.

Figure 17. Update the user prolog

Verify the list of products to be configured within this runtime environment

The configuration profile contains a list of products and components to be configured as part of this runtime environment. This list should match with the list of products that you defined when you set up the work libraries for the runtime environment. If any products are added or removed from this runtime environment after initial configuration, update this section accordingly.

Procedure

Review the list of products in the CONFIGURE PRODUCTS section and set the flags as required. Figure 18 shows an example of this section.
Update the TCP/IP port values used across the runtime environment

The ports for all monitoring agents and any monitoring servers must be the same for communication to be successful. If you are configuring the runtime environment with the default protocols of IP and IP.PIPE and plan to use a different port number from the default value (1918), update the values for all ports specified within the profile.

Procedure

1. Perform an EXCLUDE FIND on _PORT to list all the TCP/IP port values to be used by the products within this runtime environment to communicate with each other.
2. Update all port numbers to the value that you want to use. Figure 19 shows an example of this step, where nnnn is the port number that you choose to use.

```
000486 RTE_TCP_PORT_NUM 1918
000654 KDS_TEMS_TCP_PIPE_PORT_NUM 1918 * IP.PIPE
000655 KDS_TEMS_TCP_UDP_PORT_NUM 1918 * IP.UDP
000656 KDS_TEMS_TCP_PIPE6_PORT_NUM ** * IP.PIPE for IPV6
000657 KDS_TEMS_TCP_UDP6_PORT_NUM ** * IP.UDP for IPV6
000658 KDS_TEMS_TCP_PIPE6S_PORT_NUM ** * Secure IP.PIPE
000659 KDS_TEMS_TCP_PIPE6S_PORT_NUM ** * Secure IP.PIPE for IPV6
000758 KDS_PH01_TEMS_TCP_PORT_NUM 1918
001094 KC5_TEMS_TCP_PIPE_PORT_NUM 1918 * IP.PIPE
001095 KC5_TEMS_TCP_UDP_PORT_NUM 1918 * IP.UDP
001096 KC5_TEMS_TCP_PIPE6_PORT_NUM ** * IP.PIPE for IPV6
001097 KC5_TEMS_TCP_UDP6_PORT_NUM ** * IP.UDP for IPV6
001098 KC5_TEMS_TCP_PIPES_PORT_NUM ** * Secure IP.PIPE
001099 KC5_TEMS_TCP_PIPE6S_PORT_NUM ** * Secure IP.PIPE for IPV6
```

Figure 19. Update TCP/IP port values

Specify the TCPIP.DATA data set

Every agent and monitoring server started task generated in a runtime environment contains an SYSTCPD DD statement. This statement specifies the data set to be used to obtain the parameters defined by TCPIP.DATA if no GLOBALTCPIDATA statement is configured. The DD statement is tailored using parameters specified in the configuration profile for each runtime environment.

Procedure

To enable the SYSTCPD DDNAME automatically for all started tasks in a runtime environment, customize the following two parameters in the LPAR profile:

\[ \text{KDS\_X\_STC\_SYSTCPD\_INCLUDE\_FLAG} \]
\[ \text{KAG\_X\_STC\_SYSTCPD\_INCLUDE\_FLAG} \]

For example:

```
EDIT IBM.ITM62351.DEMO.WCONFIG(DEMO)
Command ===> C ' 1918 ' ' nnnn ' ALL Scro 85 csr
000541 KDS_X_STC_SYSTCPD_INCLUDE_FLAG Y <-- default is N
000542 KAG_X_STC_SYSTCPD_INCLUDE_FLAG Y <-- default in N
```

Note: You will also need to customize the GBL\_DSN\_TCP\_SYSTCPD\_TCPDATA parameter in the $GBL$USR global profile.
Results

When PARMGEN creates the monitoring server and agent started tasks, the DDNAME is automatically generated and file-tailored to point to the TCPDATA library.

**Update the global VTAM major node**

By default, the name of the VTAM major node is KCANDLE1. You might want to change the name in accordance with the naming conventions for your environment.

**Procedure**

1. Perform a FIND on RTE_VTAM_GBL_MAJOR_NODE to find the parameter for the global VTAM major node:

```
000568 ** VTAM SNA values:
000569 000570 RTE_VTAM_APPLID_PREFIX CTD
000571 RTE_VTAM_NETID USCACD01
000572 RTE_VTAM_LU62_DLOGMOD CANCTDCS
000573 RTE_VTAM_LU62_MODETAB KDSMTAB1
000574 RTE_VTAM_GBL_MAJOR_NODE KCANDLE1
000575 RTE_VTAM_APPLID_MODEL Y
```

2. Replace KCANDLE1 with the appropriate value.

**Enable APF-authorization statements**

PARMGEN provides the ability to generate the required APF-authorization commands for all libraries concatenated within the STEPLIB and RKANMODL DD names for all generated started tasks (STCs). If you do not enable this parameter, you must manually APF-authorize all libraries.

(For more information on APF authorizing libraries, see Chapter 6, “Completing the configuration,” on page 129.)

**Procedure**

1. In the LPAR profile, perform a FIND on RTE_X_STC_INAPF_INCLUDE_FLAG.
2. Change the value of the parameter to Y. Figure 20 shows an example of the modified parameter.

```
000537 RTE_X_STC_INAPF_INCLUDE_FLAG Y
```

**Update the UNIX System Services directory**

If any of the products that you are configuring require any UNIX System Services directories running on either the Hierarchical File System (HFS) or on the zSeries File System (zFS), or if you are planning to enable the self-describing agent feature, you must specify the directory to be used.
Procedure

1. Perform a FIND on the RTE_USS_RTEDIR parameter.
2. Specify the name of the USS directory.
   
   RTE_USS_RTEDIR    "rtehome"

Enable the self-describing agent feature at the Tivoli Enterprise Monitoring Server

By default, the self-describing agent feature is enabled within a remote monitoring server and any
monitoring agent that provides self-describing support but disabled at the hub monitoring server. If you
want to use the self-describing agent feature, you must enable it on the hub.

Procedure

1. If you are configuring a hub monitoring server as part of this runtime environment, perform a FIND
   on KDS_KMS_SDA in the LPAR configuration profile.
2. Change the value from N to Y. Figure 21 shows an example of this change.

   Ensure that the related parameters are set appropriately. The related GBL_ parameters are typically
   found in the WCONFIG($GBL$USR) profile for further customization.

Configure a Tivoli Enterprise Monitoring Server

If you are configuring a Tivoli Enterprise Monitoring Server as part of this runtime environment, either as
a hub or remote, you might need to modify the default parameters. The parameters that affect this
component use the prefix KDS within the configuration profile.

About this task

For more information about parameters to update, see the following sources:

- IBM Tivoli OMEGAMON XE and IBM Tivoli Management Services on z/OS: Configuring the Tivoli
  Enterprise Monitoring Server on z/OS
- Related scenarios in IBM Tivoli OMEGAMON XE and IBM Tivoli Management Services on z/OS:
  PARMGEN Reference
- IBM Tivoli OMEGAMON XE and IBM Tivoli Management Services on z/OS: Common Parameter Reference

Configure the Tivoli OMEGAMON enhanced 3270 user interface

If you want to use the OMEGAMON enhanced 3270 user interface, you must configure at least one
instance in each sysplex. The parameters that affect this component use the prefix KOB in the
configuration profile. Default values are supplied for all KOB parameters, so the interface is configured in
every runtime environment unless you explicitly exclude it from configuration. However, you can
customize the parameters to meet site requirements.
Before you begin

The enhanced 3270 user interface uses BPXmmm services for TCP/IP communications. BPXmmm services are a part of UNIX System Services, so a user ID for the enhanced 3270 user interface started task must be defined with an OMVS segment (see "Create a UNIX ID for the address space" on page 145).

About this task

You can set site-specific values for the started task name for the interface address space, VTAM applid, and VTAM node.

You can also configure security for enhanced 3270 interface. Authentication and authorization for users of the enhanced 3270 user interface is provided using the interface. Security for the enhanced 3270 user interface is configured by specifying the name of a system authorization facility (SAF) general resource class for the RTE_SECURITY_CLASS parameter. However, you should not enable security until you have completed and verified configuration of the runtime environments that the interface will serve. For more information, see "Enable security" on page 140.

Complete the following steps to configure the OMEGAMON enhanced 3270 user interface as part of this runtime environment.

Procedure

- Find the KOB$ section and update the parameters as needed. Figure 22 shows an example of this section.

```
EDIT   TDITN.&userid.DEMO.WCONF(Demo) - 01.02   Columns 00001 00072
Command ==> Scroll ==> CSR
001153 * PARAMETER NAME USAGE FOR COMPONENTS:
001154 * 1. KOB$ KOB* for OMEGAMON e3270UI
001155 * *****************************************************
001156 KOB$ BEGIN --------- OMEGAMON ENHANCED 3270 USER INTERFACE ---------
001157 *****************************************************
001158 KOB$ PARMGEN CONFIG Parameter         PARMGEN CONFIG Value
001159 *****************************************************
001160 KOB$ Tivoli OMEGAMON Enhanced 3270 UI started task options:
001161 ** Note: You only need to configure one OMEGAMON Enhanced 3270 User
001162 ** Interface address space in a Sysplex. Post-configuration, a
001163 ** CUASITE member in both the RTE's UXDBATF and RKANPARU
001164 ** Libraries may need to be created. Please refer to the
001165 ** 3270 Interface profile and threshold discussions in the Enhanced
001166 ** Interface Guide publication for more information.
001167 ** Related PARMGEN CONFIG profile parameters:
001168 ** - KDS_TEMS_DRA_FLAG
001170 ** - RTE_SECURITY_CLASS
001171 KOB$ KOB_TOM_STC CANSTOM
001172 KOB$ KOB_TOM_VTAM_NODE CTDOBN
001173 KOB$ KOB_TOM_VTAM_APPL_LOGON CTDOBAP
001174 KOB$ KOB$ END --------- OMEGAMON ENHANCED 3270 USER INTERFACE ---------
001176
```

where:

**KOB_TOM_STC**

Is the name of the started task for the enhanced 3270 user interface (by default, CANSTOM).

**KOB_TOM_VTAM_NODE**

Is the name used to build the VTAM node entry for the OMEGAMON enhanced 3270 user interface (by default, CTDOBN).
KOB_TOM_VTAM_APPL_LOGON

Is the name used to build the VTAM logon APPLID for the enhanced 3270 user interface (by default, CTDOBN).

See the IBM Tivoli OMEGAMON XE and Tivoli Management Services on z/OS: Common Parameter Reference for more information on these parameters or use the online help (see "KCIRPLBS online help macro" on page 243).

• To enable security, specify a value for RTE_SECURITY_CLASS. This parameter specifies the SAF security class to be used for log-on, query, and Take Action security for the interface. However, you might not want to enable security until you have verified that all components have been successfully configured and started. If you want to enable security after you have completed configuration of the runtime environment, edit the LPAR configuration file to specify the RTE_SECURITY_CLASS. Then follow the instructions in the RTE03 scenario in the IBM Tivoli OMEGAMON XE and Tivoli Management Services on z/OS: PARMGEN Reference to refresh the environment. If more granular security is required, you can override the RTE_SECURITY_CLASS parameter for logon, queries, and Take Action commands. For more information, see "Enable security" on page 140.

What to do next

A security administrator must define the system authorization facility (SAF) general resource class if it does not already exist and define profiles to control access to the interface, to control access to the data queries issued by the interface, and to control actions performed by the interface. Users or user groups must be given access to the profiles. "Enable security" on page 140 for more information.

A site-specific logon profile must be created (see "Create a CUASITE profile" on page 139).

Configure OMEGAMON monitoring agents and other components

Before completing all the updates to the PARMGEN profile, review and update, if needed, all parameters specific to the various OMEGAMON monitoring agents and any other components that are configured within this runtime environment. Component-specific parameters are prefixed with Kpp (where pp identifies the individual product).

About this task

See Appendix F, “Product codes,” on page 265 for a full list of product codes. Review the product-specific Planning and Configuration Guide and Parameter Reference for information on requirements or values required for a specific product.

Customize the global configuration profile

After all the settings in the WCONFIG(rte) member have been updated, the next step is to review and update the global parameters that will be used by this runtime environment and any other runtime environment that you may configure. This step is required if you are not performing a Configuration Tool-to-PARMGEN conversion or have not cloned the $GBL$USR profile from another runtime environment.

About this task

The WCONFIG($GBL$USR) member contains the parameters that define the global system libraries to be used by all components. All global parameters are prefixed GBL_.

Procedure

1. To edit the $GBL$USR member, select option 2 from the Customize PARMGEN Configuration Profile Members panel.
The WCONFIG($GBL$USR) member is displayed.

2. Review and update the parameter values as appropriate.

The key parameters to check are the common system library values:

- **GBL_DSN_SYS1_SBPXEXEC** = "SYS1.SBPXEXEC"
- **GBL_DSN_SYS1_VTAMLST** = "SYS1.VTAMLST"
- **GBL_DSN_SYS1_PARMLIB** = "SYS1.PARMLIB"
- **GBL_DSN_SYS1_PROCLIB** = "SYS1.PROCLIB"
- **GBL_DSN_SYS1_SAXREXEC** = "SYS1.SAXREXEC"
- **GBL_DSN_SYS1_VTAMLIB** = "SYS1.VTAMLIB"
- **GBL_DSN_SYS1_VTAMLST** = "SYS1.VTAMLST"
- **GBL_DSN_HZSPROC_LOADLIB** = "USER.LOADLIB"
- **GBL_DSN_CSFSFMOO** = "CSF.SCSFMOD0"

Health check configuration value: this library is used to set up the health check elements. By default, two health checks are provided (for STC and APF authorization):

- **GBL_DSN_HZSPROC_LOADLIB** = "USER.LOADLIB"

If you are enabling the ITM Password Encryption (KAES256) across the enterprise, you must set the ICSF system library:

- **GBL_DSN_CSFSFMOO** = "CSF.SCSFMOD0"

If any of the components that you are configuring within this runtime environment require Java support or if you are planning to enable the self-describing agent feature, a valid path to a Java installation on HFS or zFS must be specified. Note that /bin is automatically appended:

- **GBL_HFS_JAVA_DIR1** = "/usr/lpp/java/IBM/J6.0"

Also, if you are enabling the self-describing agent feature or any other function that uses z/OS UNIX System Services, check the value of the CLIST EXEC library name. This EXEC library is used during the creation of USS directories running on either the Hierarchical File System (HFS) or on the zSeries File System (zFS):

- **GBL_DSN_SYS1_SBPXEXEC** = "SYS1.SBPXEXEC"

If a monitoring server is using any of the IPUDP-related or IP.PIPE-related communication protocols for connection, but the IP domain name resolution is not fully configured on the z/OS system, the SYSTCPD statement must be supported by the monitoring server and all monitoring agents that
report to it. SYSTCPD explicitly identifies which data set to use to obtain the parameters defined by TCPIP.DATA when no GLOBALTCPIPDATA statement is configured. To support SYSTCPD, uncomment and set the following parameter:

`GBL_DSN_TCP_SYSTCPD_TCPDATA "TCPIP.SEZAINST"`

The name of the SYSTCPD data set is installation-specific. Get the correct specification from your network administrator. Also ensure the following flags are enabled within the LPAR config profile:

- `KAG_X_STC_SYSTCPD_INCLUDE_FLAG Y` (for monitoring agent support)
- `KDS_X_STC_SYSTCPD_INCLUDE_FLAG Y` (for monitoring server support)

3. After all required changes have been made, save the changes and return to the Customize PARMGEN Configuration Profile Members panel.

**What to do next**

If necessary, customize the `gbl_user_jcl` system variables files. Then return to the main Workflow menu and validate the profile parameter values.

**Customize the system variables profile**

If support for system variables is enabled, and you are using user-defined variables, or if you have enabled system variable overrides, you must customize the system variables profile.

**Procedure**

1. To edit the profile, from the Customize PARMGEN Configuration Profile Members, select option 3.

   ![Customize PARMGEN Configuration Profile Members](image)

   The system variables profile is displayed.

   2. Provide values for any user-defined values and values for any variables whose values you want to override in this runtime environment.

   3. Press F3 to save the changes, then press F3 to return to the main Workflow panel.
What to do next

The next step is to validate the parameter values you specified.

Step 9. Validate the PARMGEN profile parameter values

In this step, you generate and submit a job to validate all the parameters generated and edited in the previous step.

Procedure

1. From the Workflow main panel, select option 9.

2. Review the notes and submit the job. The parameter validation report is written to 
   `hilev.rte.WCONFIG($VALRPT)`

3. If the job gets a condition code greater than 4 (COND CODE 0004), enter 9s to review the $VALRPT. 
   Return to the previous step and correct the parameter values, then resubmit the validation job.

What to do next

Submit $PARSE or $PARSESV (if system variables are enabled in this runtime environment) to create the 
runtime members and the jobs that will be submitted to complete the set up of the runtime environment.

Step 10. Create the runtime environment members and jobs

This step generates jobs that parse the profile parameters detailed in previous steps. The jobs then 
generate the runtime environment members and jobs required to complete the PARMGEN setup.
Procedure

1. From the main Workflow panel (KCIP@PG0), select option 10.

<table>
<thead>
<tr>
<th>Procedure</th>
<th>1. From the main Workflow panel (KCIP@PG0), select option 10.</th>
</tr>
</thead>
<tbody>
<tr>
<td>KCIP@PG0</td>
<td>------ PARAMETER GENERATOR (PARMGEN) WORKFLOW - WELCOME -----------</td>
</tr>
<tr>
<td>Option</td>
<td>Scroll ==&gt; PAGE</td>
</tr>
<tr>
<td>Enter PARMGEN parameter values appropriate for your environment:</td>
<td></td>
</tr>
<tr>
<td>GBL_USER_JCL: TDITN.IDST.PARMGEN.JCL_______________________</td>
<td></td>
</tr>
<tr>
<td>RTE PLIB_HILEV: TSTEST.&amp;userid_____________</td>
<td></td>
</tr>
<tr>
<td>RTE_NAME: DEMO_ (Type ? for a list of configured RTEs)</td>
<td></td>
</tr>
<tr>
<td>Note: Enter n (1-11) to perform tasks. Status Date</td>
<td></td>
</tr>
<tr>
<td>Enter ns (1s-11s) for detailed job/task status. --------- ----------</td>
<td></td>
</tr>
</tbody>
</table>

1. KCIJPFCFG Set up PARMGEN work environment for an RTE. RC= 00000 2012/10/11  
2. $JOBINDX Review PARMGEN job index. Viewed 2012/10/11  
3. KCIJPCCF Clone customized WCONFIG members. (COND)  
4. KC1JUPUP1 Update interim libraries and create profiles. RC= 00000 2012/10/11  
5. KCIJPMC1 Merge profile from backup RTEDEMO (COND)  
6. KCIJPMC2 Merge profile from model RTE. (COND)  
7. KCIJPCNV Convert an ICAT RTE Batch member. (COND)  
8. DEMO Customize PARMGEN configuration profiles. Edited 2012/10/11  
9.+ KCIJPVAL Validate PARMGEN profile parameter values. RC= 00000 2012/10/11  
10. SPARSE Create the RTE members and jobs.  
11. SUBMIT Submit batch jobs to complete PARMGEN setup.  
U Utility Access PARMGEN utilities. (Optional)  
R New RTE Reset RTE, Status and Date fields. (Optional)  

The Submit $PARSE Batch Jobs To Complete PARMGEN Setup (KCIP@PRS) panel is displayed.

<table>
<thead>
<tr>
<th>Procedure</th>
<th>The Submit $PARSE Batch Jobs To Complete PARMGEN Setup (KCIP@PRS) panel is displayed.</th>
</tr>
</thead>
<tbody>
<tr>
<td>KCIP@PRS</td>
<td>---- $PARSE/$PARSESV: CREATE THE RTE MEMBERS AND JOBS -----------------------------------</td>
</tr>
<tr>
<td>Option</td>
<td>1</td>
</tr>
</tbody>
</table>
| Select option 1 to SUBMIT the full $PARSE job in WCONFIG for RTE=PLB1SP22.  
Alternatively, select other options to SUBMIT $PARSE jobs individually.  
Press Fl=Help for additional considerations when selecting options 2-5.  |
| Note: Enter ns (1s-5s) for detailed task status. Status Date  |

1. $PARSE Composite $PARSE job  
2. $PARSECM IKANMODU/WKANMODU $PARSE job (Optional)  
3. $PARSEPR IKANPARU/WKANPARU $PARSE job (Optional)  
4. $PARSESMS IKANSAMU/WKANSAMU $PARSE job (Optional)  
5. $PARSEDV Generate listing of symbolics (Optional)  

Note: If this is a reconfiguration of an existing PARMGEN-maintained RTE, then after rerunning the $PARSESV, $PARSEPR or $PARSESMS job, you must rerun the WKANSAMU(KCIJVUPV) System Variable IEBUPOTE job next, to refresh the variable-named members.  
Please see the help panel for more information.  

Figure 23. Create the runtime environment members and jobs

2. Choose one of the following approaches.  
   • To submit the full set of parsing actions within a single job, choose option 1.  
   • To submit each action individually, choose options 2 through 5 in turn.  

You may want to follow the second approach if you want to create (or recreate) only certain runtime members.  

Tip: Note that the library-specific $PARSE%% jobs regenerate the runtime members in the PARMGEN work libraries (such as WKANPARU and WKANCMDU, and so on) because PARMGEN supports a staged deployment and does not update the running production user libraries and execution libraries.
(such as RKANPARU and RKANCMDO). You must then run the WKANSAMU(KCIJPW2R) job to deploy runtime members from the WK* libraries to the RK* production user libraries. However, if you would like the $PARSE* jobs to update the production user libraries directly, without having to run KCIJPW2R, update the SYSUT2 DD to point to RK* instead of the WK* PARMGEN work libraries.

**Results**

The runtime members and jobs are created.

**What to do next**

Submit the jobs to complete the set up of the runtime environment.

---

**Step 11. Submit batch jobs to complete the PARMGEN setup**

The final step in the PARMGEN Workflow user interface provides a series of batch jobs that you submit to create the runtime environment that you have defined in the previous steps.

**Procedure**

1. From the Workflow main menu, select option 11.
   The Submit Batch Jobs To Complete PARMGEN Setup (KCIP@SUB) panel is displayed.

   ![Submit batch jobs to complete PARMGEN setup](image)

   **Figure 24. Submit batch jobs to complete PARMGEN setup**

   - **GBL_USER_JCL**: TDITN.IDTST.PARMGEN.JCL
   - **RTE_PLIB_HILEV**: TSTEST.&userid
   - **RTE_NAME**: DEMO

   Note: Enter n (1-11) to perform tasks. Status Date
   Enter ns (1s-11s) for detailed job/task status. --------- ----------
   1. KCJPCFG Set up PARMGEN work environment for an RTE. RC= 00000 2012/10/11
   2. $JOBINDX Review PARMGEN job index. Viewed 2012/10/11
   3. KCJPCCF Clone customized WCONFIG members. (COND)
   4. KCJPPUI Update interim libraries and create profiles. RC= 00000 2012/10/11
   5. KCJPPMC1 Merge profile from backup RTEDEMOR (COND)
   6. KCJPPMC2 Merge profile from model RTE. (COND)
   7. KCJPCNV Convert an ICAT RTE Batch member. (COND)
   8. DEMO Customize PARMGEN configuration profiles. Edited 2012/10/11
   9. KCJPPVAL Validate PARMGEN profile parameter values. RC= 00000 2012/10/11
   10. $PARSE Create the RTE members and jobs. RC= 00000 2012/10/11
   11. SUBMIT Submit batch jobs to complete PARMGEN setup.

---

98  IBM Tivoli OMEGAMON XE and Tivoli Management Services on z/OS: Common Planning and Configuration Guide
Select option 1 to SUBMIT the composite jobs in WKANSAMU for RTE=DEMO.
Alternatively, select other options to SUBMIT each job individually.
Press F1=Help for additional considerations when selecting options 4-14.

Note: Enter ns (1s-14s) for detailed job/task status.

---

1. KCIJPSUB Composite SUBMIT job (See JCL comments)
** or **
2. KCIJPALO Allocate runtime libraries
3. KCIJPLOD Load TK-->RK* runtime libraries
4. KCIJPSEC Product security (Conditional)
5. N/A System Variables IEBUPDTE (Conditional)
6. KCIJPUSP USS preparation (Conditional)
7. KCIJPUSS USS system set-up (Authorization required)
8. KCIJPSYS System set-up (Authorization required)
9. KCIJPLNK ASM/Link RKANMODU modules (Conditional)
10. KCIJPIVP Configuration verification
11. KCIJPCPW Backup WK* work libraries (Conditional)
12. KCIJPCPR Backup RK* production user libs (Conditional)
13. KCIJPW2R WK*->RK* deployment (Conditional)
14. KCIJPWNT Composite maintenance job (Conditional)
---

2. Choose one of the following approaches.
   - To submit the full set of actions within a single job, choose option 1
   - To submit each action individually in turn, choose options 2 through 12.

Note that some options, such as UNIX System Services system setup, may require specific user
privileges to make changes to the file system. Because the user who has been working through
PARMGEN may not have the correct authorization in place, specific jobs can be excluded and
submitted by a user who does have the correct authorization.

If you decide to submit all the jobs automatically, submit the composite KCIJVSUB job (if system
variables are enabled) or KCIJPSUB job (if system variables are disabled) in the WKANSAMU library.
Otherwise, submit the jobs individually.

In the following list of batch job names, c = V if system variables are enabled and c = P if system
variables are disabled. For example, the KCIJVALO job allocates the runtime libraries with system
variables, and the KCIJPALO job allocates the runtime libraries without system variables.

KCIJcALO
This required job allocates the RK* runtime libraries for all the products and components in
the runtime environment.

KCIJcLOD
This required job copies members of the target libraries that were installed by SMP/E to the
read-only RK* libraries. It also deletes any runtime members from the different libraries based
on the product versions' FMID requirements. For example, if the latest version of the product
removes or renames any SMP/E elements.

Specifically, the BUILDEX/DELRUN step bypasses delete requests for elements that may
already have been deleted from a previous run of the KCIJcLOD RTE load job. Additionally, if
any of the libraries listed in the DELRUN DDNAME is part of the system link libraries, or the
library may be in use, the KCIJcLOD RTE load job may not be able to delete these elements if
the libraries are enqueued. If either of these situations are true, run the BUILDEX/DELRUN
step when those libraries are available.

KCIJcSEC
This job is required if the product-specific IBM-supplied security exit or input must be
customized. The job creates security-related members (load modules, encryption key, and
other elements) based on the product security requirements.
Review the KppSUPDI "Modify Classic command table" security steps. The KppJPSCo input members to the composite KCIIjPSEC security job point SYSIN DD to WKANSAMU(KppSUPDI) by default. If you must make further changes to the sample exit, modify the SYSIN DD accordingly. Also see the RTE_X_SECURITY_EXIT_LIB parameter in the PGN3MVSE CONFIG LPAR profile.

KCIIc-USP
This job is required if you are configuring at least one product that requires UNIX System Services. The job creates the USS-related members in the RKANDATV runtime library for use in the composite KCIIcUSS job.

Tip: This job is required if you enabled the self-describing agent feature.

KCIIc-USS
This job is required if you are configuring at least one product that requires UNIX System Services. The job creates the Hierarchical File System (HFS) or the zSeries File System (zFS) directories and subdirectories, and copies files to HFS or zFS.

Tip: This job is required if you enabled the self-describing agent feature.

KCIIc-SYS
This job copies the product started tasks, VTAM major node members, and health check elements for the products and components into system libraries, and assembles and links product modules into system libraries. The job requires write access to system libraries.

KCIIc-LNK
The job assembles and links elements into the SYSLMOD RKANMOD* load library.

KCIIc-UPV
If system variables are enabled (RTE_SYSV_SYSVAR_FLAG = Y), this job must be submitted in the target LPAR where the symbolics are resolved. This job populates variable-named members contained in the application-specific KppJPUPB composite IEBUPDTE members in the WK* work output libraries. See the KCII@SUB help panels for more information.

KCIIc-CPY
This job backs up the RK* runtime libraries and the WK* work output libraries. If you run this job individually (rather than running the composite job to submit all the batch jobs), run the job three times:
  a. Clone the PARMLIB interim staging libraries (IK*).
  b. Clone the PARMLIB work output libraries (WK*).
  c. Clone the existing production runtime user libraries (RK*).

KCIIc-W2R
This job copies the WK* work output libraries to the respective production RK* runtime libraries. If you do not run this job, you must copy the work output libraries to the runtime libraries by some other method, following your normal change control process.

KCIIc-IVP
This required job verifies that all the required runtime data sets, members, and configuration jobs for the runtime environment were created, and that the jobs were executed successfully.

3. On completion of these jobs, review the output of the KCIIcIVP configuration verification job:
   a. Review the $IVPRPT report, which is stored in the WCONFIG library. This report shows the results of each KCIIjP* job that was run to configure the runtime environment. In the following example, the report shows that the KCIIJPOD job had an ABEND E37 out-of-space condition in one or more of the libraries loaded by the job:

<table>
<thead>
<tr>
<th>JOB</th>
<th>STATUS</th>
<th>JOBNAME</th>
<th>JOB#</th>
<th>DATE</th>
<th>TIME</th>
<th>HI-CC</th>
</tr>
</thead>
<tbody>
<tr>
<td>KCIIJPCFG</td>
<td>OK</td>
<td>CCAPI$JP</td>
<td>J29242</td>
<td>10.056</td>
<td>22:46:42</td>
<td>00000</td>
</tr>
</tbody>
</table>
b. Review the WSUPERC SYSTSPRT report, which is stored in the WSUPERC sequential library. This report compares the PARMGEN WK* work output libraries with the production RK* runtime libraries. The PARMLIB WK* work output libraries are the data sets created and populated by the $PARSE job. In the following example, two files in the new WKANCMDU data set do not have equivalents in the existing RKANCMDU data set:

```
48 TOTAL MEMBER(S) PROCESSED AS A PDS
1 TOTAL MEMBER(S) PROCESSED HAD CHANGES
47 TOTAL MEMBER(S) PROCESSED HAD NO CHANGES
2 TOTAL NEW FILE MEMBER(S) NOT PAIRED
0 TOTAL OLD FILE MEMBER(S) NOT PAIRED
1 ISRSUPC - MVS/PDF FILE/LINE/WORD/BYTE/SFOR COMPARE UTILITY- ISPF

NEW: &rhilev.&rte..WKANCMDU.
OLD: &rhilev.&rte..RKANCMDU
```

MEMBER SUMMARY LISTING (LINE COMPARE)

<table>
<thead>
<tr>
<th>NON-PAIRED NEW FILE MEMBERS</th>
<th>NON-PAIRED OLD FILE MEMBERS</th>
</tr>
</thead>
<tbody>
<tr>
<td>KC2OPS01</td>
<td></td>
</tr>
<tr>
<td>KC2STA01</td>
<td></td>
</tr>
</tbody>
</table>

c. Correct any errors and rerun any jobs that did not complete successfully. Resubmit the KCIJcIVP job whenever the work output libraries change (for example, after applying maintenance and rerunning the $PARSE or $PARSEV job).

Results

The runtime environment DEMO is created. Complete any configuration steps required outside the configuration software (see Chapter 6, “Completing the configuration,” on page 129).

What to do next

After you successfully complete all the steps listed within the PARMGEN Workflow User Interface, a new runtime environment is created. Several steps must be completed outside of the PARMGEN Workflow User Interface before the runtime environment can be started. For more information, see Chapter 6, “Completing the configuration,” on page 129.

You can also use the PARMGEN method to replicate environments. For instructions on using PARMGEN to replicate (“clone”) runtime environments, see the Chapter 7, “Using the PARMGEN method to replicate a configured runtime environment,” on page 149 and IBM Tivoli OMEGAMON XE and Tivoli Management Services on z/OS: PARMGEN Reference.
Chapter 5. Configuring products with the Configuration Tool

Using the Configuration Tool (formerly known as ICAT) method, you supply configuration parameters in a series of ISPF panels and then submit the jobs that create a runtime environment and configure products in the environment using those parameters.

Before you begin

Before you can begin the procedures in this section, you must complete the SMP/E installation of any monitoring agents that you want to configure and the following Tivoli Management Services components:

- Configuration Tool
- Tivoli Enterprise Monitoring Server
- TMS:Engine

To install the components, follow the instructions in the IBM Tivoli Management Services on z/OS: Program Directory for IBM Tivoli Management Services on z/OS. To install monitoring agents, consult the Program Directory for each product.

About this task

This section contains the procedures for using the Configuration Tool for configuring the Tivoli Enterprise Monitoring Server on z/OS and z/OS-based monitoring agents:

If you are using the PARMGEN configuration method, skip this section and follow the instructions in Chapter 4, “Configuring products using the PARMGEN method,” on page 63.

The Configuration Tool provides default values wherever possible. These default values are generally sufficient to complete the installation, but can be changed to meet the requirements of your enterprise. For information about the features of the Configuration Tool, see the Common Planning and Configuration Guide. For information about the parameters being configured, see the Common Parameter Reference and the Parameter Reference for each product you are configuring.

Step 1. Create or update the Configuration Tool work library

Setting up the Configuration Tool involves copying the contents from a target library into the Configuration Tool work library. You can set up the Configuration Tool in an existing consolidated software inventory (CSI), a cloned copy of an existing CSI, or a new CSI.

About this task

- If you are using an existing CSI that contains the Configuration Tool version 310.12 or higher, you can skip this step and go to “Step 2. Start the Configuration Tool” on page 104. The Configuration Tool libraries are updated automatically.
- If you are using a cloned copy of an existing CSI that contains the Configuration Tool V310.12 or higher, go to “If you use a cloned copy of a CSI containing the Configuration Tool.”
- If you are using a new CSI, go to “If you use a new CSI” on page 104.

If you use a cloned copy of a CSI containing the Configuration Tool

About this task

If you use a cloned copy of an existing CSI that contains the Configuration Tool, follow these steps to update the work library:
Procedure

1. Use the following JCL to create a job to copy the contents of the *thilev*.TKCIINST library to the *shilev*.INSTLIB library:

```
//COPY EXEC PGM=IEBCOPY
//SYSPRINT DD SYSOUT=*  
//IN DD DSN=thilev.TKCIINST,DISP=SHR  
//OUT DD DSN=shilev.INSTLIB,DISP=SHR  
//SYSIN DD *  
COPY O=OUT,I=((IN,R))
```

where *thilev* is the SMP/E target high-level qualifier and *shilev* is the installation high-level qualifier.

 Tip: To receive notification of the results of a job, add this option to your job card:

```
NOTIFY=userid
```

2. Submit the job.

If you use a new CSI

About this task

If you are using a new CSI, follow these steps to create the work library:

Procedure

1. Customize and submit the KCIJALOC job, which you can find in the *thilev*.TKCIINST library (where *thilev* is the high-level qualifier of the target libraries that were installed by SMP/E). Follow the instructions in the JCL. The KCIJALOC job allocates the following installation libraries, which are required by the Configuration Tool:

   - *shilev*.INSTDATA
   - *shilev*.INSTDATW
   - *shilev*.INSTJOBS
   - *shilev*.INSTLIB
   - *shilev*.INSTLIBW
   - *shilev*.INSTLOG
   - *shilev*.INSTQLCK
   - *shilev*.INSTSTAT

   (where *shilev* is the high-level qualifier of the installation libraries). The job also copies elements from the *thilev*.TKCIINST library to the *shilev*.INSTLIB library.

2. Recycle the ISPF session.

Step 2. Start the Configuration Tool

About this task

Follow these steps to start the Configuration Tool:

Procedure

1. Log on to a TSO session on the target system.

   The target system is the logical partition (LPAR) where you are going to create and configure a runtime environment for the Tivoli Enterprise Monitoring Server and any monitoring agents you intend to install and configure on the same LPAR.

2. Enter ISPF

   The Primary Option Menu is displayed.

3. From the Primary Option Menu, enter option 6 (Command).
4. Turn off the predefined function (PF) keys so that the Configuration Tool function keys are not truncated. To do this, enter PFSHOW on the command-line and repeat this command until the ISPF default function keys are no longer displayed.

5. Enter the following command:
   EX 'shilev.INSTLIB'

   where shilev is the high-level qualifier that you specified when you created the INSTLIB data set (see "Step 1. Create or update the Configuration Tool work library" on page 103)

   **Tip:** You do not have to specify a member name in this command.

   A panel similar to Figure 25 is displayed, or a job is displayed that you can use to update the Configuration Tool to the most recent version.

   ![Configuration Tool Main Menu](image)
   
   **Figure 25. Configuration Tool Main Menu**

   **Configuration Tool usage notes:**
   - Do not use the ISPF feature for edit recovery. If the ISPF RECOVERY ON command is entered, edits produce a recovery error message.
   - Avoid using a split screen, because the last line of information at the bottom of the screen might not be displayed, and you cannot scroll down and might miss important messages or steps.
   - Enter panelid in the OPTION ===> line to see the panel ID in the upper left corner of each Configuration Tool panel.
   - The online help for the Configuration Tool contains detailed information about using the Configuration Tool panels. To display help information, press the F1 key (Help) or enter HELP on the command-line.

   README files provided by the Configuration Tool give additional usage information. If you enter R on the Runtime Environments panel, the Configuration Tool displays a list of available README files for selection. You can also enter README ccc (where ccc is the 3-character code for the type of README file) at the command-line of any Configuration Tool panel. For example, README RTE is the command for information about runtime environments, and README BAT is the command for information about batch mode. To obtain information about new features that affect the configuration of all components and products installed in the CSI, enter README AAA.

   - The content of some files displayed by the Configuration Tool is generated dynamically and contains information specific to your environment. Examples include some of the online help, the README information, and the list of tasks shown in Complete the Configuration.
Step 3. Set up the Configuration Tool work environment

In this step, if you have not already done so, you set up the Configuration Tool work environment by specifying the location of the work libraries, customizing the JCL for batch jobs, setting initial default values for the configuration environment, and allocating the work libraries.

About this task

To set up the work environment, follow these steps:

Procedure

1. Specify the JCL options.
   a. From the Main Menu panel shown in Figure 25 on page 105, enter 1 (Set up work environment). The Set Up Work Environment panel is displayed (Figure 26).

   KCIPIN1 ------------------- SET UP WORK ENVIRONMENT ---------------------------
   Enter the number to select an option: Date Time
   1 Specify JCL options
   2 Set up configuration environment
   3 Create batch mode job
   4 Allocate work libraries

   Note: Once you create and submit the "Allocate work libraries" job, you must exit the installer and allow the job to run before restarting the installer.

   F1=Help F3=Back

   Figure 26. Set up Work Environment panel

   b. Enter 1 (Specify JCL options) to specify values for generating batch configuration jobs and for creating the work data sets required by the Configuration Tool. The Specify JCL Options panel is displayed (Figure 27).

   KCIPIN2 ---------------------- SPECIFY JCL OPTIONS -------------------------------
   COMMAND ===>
   Specify allocation and processing options:
   JCL REGION value ==> 0M (Specify K/M suffix)
   Unit/ Storclas/ VolSer Mgmtclas PDSE
   Installation work data sets ............... unit N
   volser

   Specify the job statement for generated JCL:
   ==> //useridA JOB (ACCT), 'NAME',CLASS=A,MSGCLASS=A,NOTIFY=SYSUID
   ==> // DEFAULT JCL
   ==> //
   ==> //

   Note: Press F1=Help for Batch Mode considerations when modifying jobcard values.

   Enter=Next F1=Help F3=Back

   Figure 27. Specify JCL Options panel
c. Modify the values on the Specify JCL Options panel as required by your enterprise.

**JCL REGION value**
Specify the required storage value, in either KB or MB, for the REGION parameter on the EXEC statement of batch configuration jobs. The syntax is *numberK* or *numberM*. The default value is 0M (zero MB).

**Installation work data sets**
Specify the attributes for allocating the work data sets required by the Configuration Tool.

**SMS considerations:**
If you plan to allocate SMS-managed data sets, ensure that the following conditions are all true:

- SMS is active on the z/OS image.
- The high-level qualifier of the data sets is eligible for SMS-managed volumes.
- You specify a combination of VOLSER, UNIT, STORCLAS, and MGMTCLAS parameters that is valid at your site. Because SMS can be implemented in several different ways, the Configuration Tool does not attempt to validate these parameters. The data set allocation jobs use the values you enter.

**Unit**
Specify the unit name to be used for allocating the non-VSAM work data sets. This value is required if the data sets are not managed by SMS.

**Volser**
Specify the volume serial numbers to be used for allocating the work data sets. This value is required if the data sets are not managed by SMS.

**Storclas**
If the data sets are to be managed by SMS, specify the SMS storage class to be used for the allocation. If your site does not require the SMS STORCLAS parameter, you can leave this field blank.

**Mgmtclas**
If the data sets are to be managed by SMS, specify the SMS management class to be used for the allocation. If your site does not require the SMS MGMTCLAS parameter, you can leave this field blank.

**PDSE**
If the non-VSAM data sets are to be managed by SMS, you can specify Y to allocate PDSE data sets instead of PDS data sets. PDSE data sets do not require compression and are not limited by a predefined number of directory entries.

The default is N. Even if you specify Y, most load module libraries (RKANMOD, RKANMODL, RKANMODR, and RKANMODU) are not allocated as PDSE data sets. The main exception is RKANMODP, a load module library used for the SMP/E CALLLIBS facility.

**Job statement**
Supply the standard jobcard to be used for batch jobs generated by the Configuration Tool. Control variables can be used but require an additional ampersand (&) for correct resolution of the variable. For example, if you use the SYSUID control variable with the NOTIFY parameter in your job statements, specify NOTIFY=SYSUID on this panel, as shown. This value resolves to SYSUID in the batch job generated by the Configuration Tool.

If you use batch mode processing (see Part 3, “Replicating configured runtime environments,” on page 147), this standard jobcard is also used in batch mode configuration jobs. When you create the CICATB batch job member in the shileo.INSTJOBS data set, the Configuration Tool also creates the batch jobcard ISPF table KCITPIG1 in the shileo.INSTDATA data set. The contents of the KCITPIG1 table are used as the standard
jobcard in Configuration Tool-generated batch configuration jobs. If you modify the jobcard on this panel, you must recreate the CICATB batch job to refresh the KCITPIG1 jobcard table before running CICATB.

d. Press Enter to save the values and return to the Set Up Work Environment panel.

2. Set up the configuration environment.
   a. On the Set Up Work Environment panel (shown in Figure 26 on page 106), enter 2 (Set up configuration environment).
      The Set Up Configuration Environment panel is displayed, as shown in Figure 28. The values specified on this panel become the default values for the first runtime environment you configure.

      | Command | Set Up Configuration Environment |
      |---------|----------------------------------|
      | RTE allocation routine | IKJEFT01 (IKJEFT01/IEFBR14) |
      | VSAM data sets High-Level Qualifier Unit/ Storclas/ data sets High-Level Qualifier Unit/ Storclas/ |
      | RVHILEV unit | VSAM RVHILEV unit |
      | N Volser Mgmtclas PDSE |
      | N Volser N |
      | Work data sets High-Level Qualifier Target data sets High-Level Qualifier |
      | Unit |
      | Unit |
      | Unit |
      | Unit |

      Figure 28. Set up Configuration Environment panel

   b. Specify the high-level qualifiers for the VSAM, non-VSAM, and SMP/E runtime libraries.
      The length of the high-level qualifier for the runtime libraries must be no longer than 26 characters.
      If high-level qualifiers for the runtime libraries have already been specified, unlock them by selecting Unlock runtime high-level qualifiers from the Configuration Services and Utilities panel (Figure 49 on page 231). See Appendix C, “Configuration Tool reference,” on page 231 for more information.
      The high-level qualifiers specified here are used as defaults for the first runtime environment you configure. After that, the default values are those of the runtime environment most recently worked on in the Configuration Tool.

   c. Review the storage specifications and make any changes required for your enterprise.

   d. Press Enter to save the values and return to the Set Up Work Environment panel.

3. Allocate the Configuration Tool work libraries.
   a. On the Set Up Work Environment panel (shown in Figure 26 on page 106), enter 4 (Allocate work libraries).
      The JCL to allocate the work libraries is displayed.

   b. Review the JCL and edit if necessary.
      The job allocates the following work libraries with the shilev high-level qualifier:

      **INSTDATA**
      Contains all installation-specific information.

      **INSTDATW**
      Contains installation-specific information that was loaded from the distribution media.
INSTJOBS
Contains all generated jobs.

INSTLIBW
Contains the latest release of the Configuration Tool found on the distribution media. The contents of the INSTLIBW data set are moved to shilev.INSTLIB unless the latest version of the Configuration Tool is already in the INSTLIB data set.

INSTLOG
Contains the cumulative configuration batch job log. For each job run, the log includes the data set and member name of the job; and for each job step, the log lists the date and time of job completion, the JES job name and number, the step name, the procedure step name, the completion code, and the name of the program executed.

INSTQLCK
Used by internal installer processes as a locking data set. Used to prevent concurrent updates to the same data set.

INSTSTAT
Contains status-specific installation information.

The default space allocations for the work libraries are adequate for most environments. However, the INSTDATA and INSTJOBS libraries expand when runtime environments are added and products are configured in those runtime environments. Review the space allocations specified in the JCL and increase them if required.

c. Submit the job to allocate the work libraries.
If you do not submit the job while you are running the Configuration Tool, the job is stored in the CISUPG2 member in the INSTJOBS data set. You can submit the job later when you exit the Configuration Tool. Depending on the environment, some of the data set members might not contain data.
d. Press F3 repeatedly until you exit the Configuration Tool. (The Allocate Work Libraries job cannot run while you are in the Configuration Tool.)
e. Verify that you have a return code of 0 for the Allocate Work Libraries job.

Step 4. Set up and build a runtime environment
Before you configure monitoring agents or monitoring servers on a z/OS system for the first time, you must set up the runtime environments and build in which you will configure them.

Before you begin
Review "Decision 7: What types of runtime environments to set up" on page 44 before you begin setting up your runtime environments.

About this task
Tips:
• If you intend to create runtime environments that share a base runtime environment, create the base runtime environment first.
• Before you begin using product-specific configuration instructions, read "Step 5. Configure your products and components in the runtime environment" on page 121, which suggests best practices for product configuration.
• See Table 14 on page 240 for more information on when each runtime environment configuration option is required.

Follow this procedure to set up and build a new runtime environment.
**Procedure**

1. Access the Runtime Environments (RTE) panel.

   The Runtime Environment panel is one of the primary recurring panels in all configuration operations. This panel lists the actions you can perform to create and manage runtime environments.

   To access the Runtime Environments (RTE) panel, complete the following steps:

   a. If the Configuration Tool is not already running, start it:

      ```
      shilev.INSTLIB
      ```

   b. From the Configuration Tool Main Menu (shown in Figure 25 on page 105), enter 3 (Configure products). The Configure Products panel is displayed:

      ![Configure Products panel](image)

      **Figure 29. Configure Products panel**

   c. On the Configure Products panel (Figure 29), enter 1 (Select product to configure).

      The Product Selection Menu (Figure 30) lists the products available for configuration.

      ![Product Selection Menu](image)

      **Figure 30. Product Selection Menu**

   d. Enter S (Select) to the left of the name of one of the monitoring agents or Tivoli Management Services.

      **Important:** If you want to configure both a monitoring server and monitoring agents in the runtime environment, select one of the monitoring agents, not Tivoli Management Services on z/OS. If you select a monitoring agent, you can configure a monitoring server and one or more monitoring agents in the same runtime environment. If you select Tivoli Management Services on z/OS, you can only configure a stand-alone monitoring server in its own runtime environment, without any monitoring agents. For instructions on configuring a stand-alone monitoring server, see IBM Tivoli Management Services on z/OS: Configuring the Tivoli Enterprise Monitoring Server on z/OS.

      The Runtime Environments (RTEs) panel (Figure 31 on page 111) is displayed.
2. Add a runtime environment.
   a. On the Runtime Environments (RTEs) panel, type A (Add RTE) in the Action field beside the first (empty) row and type a name for your new runtime environment in the Name field.

   The runtime environment name is a unique identifier of up to 8 characters. It is automatically used as the mid-level qualifier for full and sharing runtime environments. You can optionally specify a mid-level qualifier for base runtime environments.

   **Tips:**
   - Use a variation of the runtime environment name for the JCL suffix (used as the suffix of the name of the member containing the JCL in the INSTJOBS data set). If you specify a runtime environment name no more than 4 characters long, you can specify the same name for the JCL suffix. This setup makes it easy to identify the runtime environment associated with each job generated by the Configuration Tool and stored in the INSTJOBS data set.
   - If you plan to use system variables, the runtime environment name must correspond in whole or in part to the value of the symbolic specified in a later panel. When you use system variables, components inherit the system values of the host z/OS system. These system-specific values are loaded into temporary data sets that exist only for the duration of the component execution. See Appendix A, “Enabling system variable support,” on page 199.
   - When you enter a C (Configure), B (Build), or L (Load) next to the name of a runtime environment that has a previous version of the Tivoli Enterprise Monitoring Server installed, the Configuration Tool prompts you to confirm that you want to upgrade to the newer version. A batch job completes the upgrade and retains all previously configured values. See IBM Tivoli OMEGAMON XE and IBM Tivoli Management Services on z/OS: Upgrade Guide.

   b. Specify the type of runtime environment being created. If you intend to create any sharing environments, start by creating the base or full runtime environment that they will share, so that you can then add the sharing runtime environment later. The following types are valid:

      **Full**  Allocates both private and base libraries. Use this type if you require only one runtime environment or if you require a runtime environment for a unique set of products.

      **Base**  Allocates read-only base libraries and does not execute alone. Use this type only in conjunction with sharing runtime environments. Create the base runtime environment first.

      **Tip:** You cannot configure products in base runtime environments because they contain shared, read-only libraries exclusively.

      **Sharing**  Allocates private libraries only. This type can either share base libraries with a base or full
runtime environment, or use target libraries that were installed by SMP/E for its base libraries. Define one sharing runtime environment for each z/OS image.

For more detailed information about the different types of runtime environments, see [Decision 7: What types of runtime environments to set up] on page 44.

c. (For sharing runtime environments only) Type the name of the base or full runtime environment from which the sharing runtime environment obtains its base library information. If target libraries that were installed by SMP/E are to be shared, type SMP.

d. Type a description for this runtime environment. The description can be any information that is useful for you and others at your site.

e. When you have specified all required values on the Runtime Environments (RTEs) panel, press Enter.
   - If you are adding a base runtime environment, the panel shown in Figure 32 is displayed.
   - If you are adding a full or sharing runtime environment, the panel shown in Figure 33 on page 113 is displayed.

![Figure 32. Add Runtime Environment panel for a base runtime environment](image-url)
The values on the Add Runtime Environment panels are used to allocate runtime libraries and provide configuration defaults.

f. Use the following information to complete the first Add Runtime Environment panel.

**High-level qualifier**
Specify the high-level qualifiers to be used for allocating the runtime data sets. These values are required. The names of the runtime data sets are generated by appending a suffix to the high-level qualifier.

**SMS considerations:**
If you plan to allocate SMS-managed data sets for the runtime environment, ensure that the following conditions are all true:

- SMS is active on the z/OS image.
- The high-level qualifier that you specify is eligible for SMS-managed volumes.
- You specify a combination of VOLSER, UNIT, STORCLAS, and MGMTCLAS parameters that is valid at your site. Because SMS can be implemented in several different ways, the Configuration Tool does not attempt to validate these parameters. The data set allocation jobs use the values you enter.

**Volser**
Specify the volume serial numbers to be used for allocating the runtime data sets. This value is required if the runtime data sets are not managed by SMS.

**Unit**
Specify the unit name to be used for allocating the non-VSAM runtime data sets. This value is required if the runtime data sets are not managed by SMS.

**Storclas**
If the runtime data sets are to be managed by SMS, specify the SMS storage class to be used for the allocation. If your site does not require the SMS STORCLAS parameter, you can leave this field blank.

**Mgmtclas**
If the runtime data sets are to be managed by SMS, specify the SMS management class to be used for the allocation. If your site does not require the SMS MGMTCLAS parameter, you can leave this field blank.

**PDSE**
If the non-VSAM data sets are to be managed by SMS, you can specify Y to allocate PDSE.
data sets instead of PDS data sets. PDSE data sets do not require compression and are not limited by a predefined number of directory entries.

The default is N. Even if you specify Y, most load module libraries (RKANMOD, RKANMODL, RKANMODR, and RKANMODU) are not allocated as PDSE data sets. The main exception is RKANMODP, a load module library used for the SMP/E CALLLIBS facility.

**Mid-level qualifier**
(Optional, for base runtime environments only) Specify a mid-level qualifier for base runtime library allocations. This field is modifiable for base runtime environments only. For full and sharing runtime environments, the mid-level qualifier is set to the runtime environment name and cannot be changed.

**JCL suffix**
Specify a unique JCL suffix, no more than 4 characters long, to identify the batch job members created in the shilev.INSTJOBS data set by the Configuration Tool for this runtime environment. Check the INSTJOBS data set to ensure that the suffix is not already in use.

If possible, specify the runtime environment name (or an abbreviated version of the runtime environment name) as the JCL suffix. This setup makes it easy to identify the runtime environment associated with each job generated by the Configuration Tool and stored in the INSTJOBS data set.

**STC prefix**
For full and sharing runtime environments, specify a prefix to be used when generating started task procedures for products configured in the runtime environment. The default value of CANS is provided, but specifying your own prefix prevents confusion with jobs generated by other runtime environments on the same system.

**SYSOUT class**
Specify the value of the SYSOUT class for non-diagnostic output DDNAMEs, such as RKPDLOG, in generated JCL. This value replaces the previously hardcoded SYSOUT class value.

**Load optimization**
Specify whether you want to optimize the loading of this runtime environment, when the runtime environment load action (L) is selected after maintenance is applied or products are reconfigured.

If you specify Y, the load job has the following characteristics:
- Copies only modified modules from target to runtime libraries.
- Requires access to IBM’s SuperC (ISRSUPC) utility.
- Uses less DASD space.
- Performs additional analysis, which uses more CPU processing and file I/O.

If you specify N (the default), the load job has the following characteristics:
- Copies all members from target to runtime libraries, whether or not they were modified.
- Requires more DASD space.
- Uses less CPU time.

**Tip:** Even if you enable load optimization, the first load operation copies all members to the runtime data sets. Load optimization takes effect on the second and subsequent load operations.

**Remote RTE for transport**
Specify Y if you want to create this runtime environment with the intention of transporting it to other systems, rather than of running it on the local system. When you finish adding the runtime environment, you skip building the runtime libraries (action B on Figure 31 on page 111). You configure the runtime environment (action C), but you do not run the generated configuration batch jobs or load the libraries (action L). Subsequently, you can
transport the runtime environment to other systems, where you build the runtime libraries, run the configuration batch jobs, and load the runtime libraries. Because there are no runtime libraries on the system where the runtime environment was originally created, you cannot analyze the libraries for user-modified elements on that system. But you can analyze the libraries on the first system to which you transport the runtime environment and on which you build the libraries by using the Analyze user-modified elements option.

The default is N.


Runtime members analysis
The Configuration Tool generates configuration batch jobs for the runtime environment and can identify user-modified data set members that each batch job will affect. If runtime members analysis is enabled, a report of user-modified members is displayed. You can also generate these reports from the RTE Utility Menu (action Z).

The default is Y.

Diagnostic SYSOUT class
Specify the value of the SYSOUT class for diagnostic output DDNAMEs, such as SYSUDUMP and SYSABEND, in generated JCL. This value replaces the previously hardcoded SYSOUT class value. The default is X.

Will this RTE have a Tivoli Enterprise Monitoring Server?
This question is asked for full and sharing runtime environments only. If you are creating a runtime environment for stand-alone monitoring agents without a monitoring server, specify N. Otherwise, specify Y (the default) to allocate libraries for the monitoring server.

Be sure to make a note of the TEMS name value displayed here. You must specify it when you configure components and products in the runtime environment, and when you set up communications with the distributed components of Tivoli Management Services (IBM Tivoli Monitoring). The TEMS name parameter is case sensitive on all platforms.

Copy configuration values from RTE
For full and sharing runtime environments, you can specify the name of an existing runtime environment from which you want configuration values to be copied for this runtime environment. This procedure makes an exact copy of the existing runtime environment. If the two runtime environments will not contain the same products, do not use this procedure.

g. After you have specified all required values on the first Add Runtime Environment panel, press Enter.

• If you are adding a base runtime environment, the Runtime Environments (RTEs) panel (Figure 31 on page 111) is displayed. This step completes the addition of the runtime environment. You must build the runtime libraries before adding a sharing runtime environment for the product-specific libraries. Go on to 3 on page 120.

• If you are adding a full or sharing runtime environment, the second Add Runtime Environment panel (Figure 34 on page 116) is displayed.
h. Use the following information to complete the second Add Runtime Environment panel.

**Use z/OS system variables?**
Specify Y if you want to use z/OS system variables. The default is N.

When you use system variables, components inherit the system values of the host z/OS system. These system-specific values are loaded into temporary data sets that exist only for the duration of the component execution. See [Appendix A, “Enabling system variable support,” on page 199](#).

**Tips:**
- If system variable support is enabled, you can enter README SYS at the command-line for more information on how the Configuration Tool uses z/OS system symbols to process VTAM applids.
- For complete information about z/OS system symbols, see the [z/OS MVS Initialization and Tuning Reference](#).

If you specify Y for system variables, supply values for the remaining fields in this section:

**RTE name specification**
Specify the system variable for the name of the runtime environment. The default is SYSNAME.

The resolved name must be a single valid JCL symbol, as defined in the [z/OS MVS JCL Reference](#). This field becomes the value of the SYS= parameter (for example, 
SYS='SYSNAME') in all started task members. This field can contain both literals and symbolics. For example, if you specify a value of CANSYSNAME, the value resolves to a runtime environment name of CANSYSA if SYSNAME=SYSA.

**RTE base alias specification**
If the runtime environment is sharing with a base runtime environment, specify a system variable for the base runtime environment. This value is then inserted into the base runtime environment data set references in all started tasks. The resolved name must be a valid data set name qualifier.

If the runtime environment you are defining is not sharing with a base runtime environment, the label n/a is displayed beside this field.

**Applid prefix specification**
If you are using system variables, enter a VTAM applid prefix that contains system
variables. The resolved prefix can be a maximum of 4 characters. Be sure to place a period (dot) after the last symbolic in the specification. SYSCLONE is the default.

For information on system variable considerations for the VTAM Major Node name, enter README SYS at the command-line and read the sample usage scenario.

**Use VTAM model applids?**

Specify Y if you want this runtime environment to use VTAM model applids; that is, applids with wildcard suffixes such as ? or *. These model applids allow the use of any applids that match the pattern in the VTAM node.

The default is N.

**Security system**

Specify which, if any, security system is to be used for this runtime environment. The default is NONE. If you specify a security system, verify that it is installed and configured correctly for your site. If you specify ACF2, you must also provide the name of the ACF2 macro library.

**Tip:** Specifying a security system here indicates which system will be used for security validation of users signing on to the Tivoli Enterprise Portal, but it does not enable the validation. Security validation of users is enabled in a Tivoli Enterprise Monitoring Server configuration panel. For more information about configuring security, see IBM Tivoli Management Services on z/OS: Configuring the Tivoli Enterprise Monitoring Server on z/OS.

**Fold password to upper case**

By default, TMS:Engine converts logon passwords to uppercase. However, RACF V1.7 supports mixed-case passwords. If you want to implement mixed-case passwords and if all your monitoring agents support them (this requires application of maintenance for each monitoring agent), set this field to N.

If any of your monitoring agents do not support mixed-case passwords, do not activate the SETROPTS PASSWORD(MIXEDCASE) option in RACF and do not enable mixed-case passwords in your runtime environments. Leave this field with the default value Y.

For more information about the MIXEDCASE option in RACF, see the z/OS V1R8.0 Security Server RACF Security Administrator’s Guide. For more information about the way TMS:Engine handles security, enter README SEC at the command-line of any Configuration Tool panel.

**Global SAF class name**

Specify a common System Authorization Facility (SAF) security class name for OMEGAMON enhanced 3270 user interface security controls. The value is used in the RTE_SECURITY_CLASS parameter in the KppENV member of the RKANPARU library. If this field is not specified, the RTE_SECURITY_CLASS parameter is generated as a placeholder comment in the RANAPARU(KppENV) of components exploiting the parameter. Specify a valid SAF class name if you want to specify a value. If you are using ACF2 as your external security resource manager, specify a maximum of three characters.

**VTAM communication values**

VTAM communication values are required for some monitoring agents and for the monitoring servers with which those monitoring agents communicate directly. Examples of such monitoring agents include OMEGAMON XE on z/OS (for the EPILOG facility of the OMEGAMON II component) and OMEGAMON XE for Messaging on z/OS (for the 3270 interface component). See the product-specific configuration guides for further information about SNA requirements.

The values you specify here become the default values for the products and components configured in the runtime environment. If none of the products or components are to use the SNA communication protocol (for example, if you are defining a runtime environment for a high-availability hub), clear these fields.
Applid prefix

Type a global VTAM applid prefix (from 1 to 4 characters) to be used when building the VTAM applids for products in this runtime environment. The default is CTD. If none of the products or components you intend to configure in this runtime environment require SNA communications, clear this field.

Tips:
- If you have enabled system variables, the applid prefix value specified in the system variables section of this panel is used instead of the value specified in the VTAM communication values section.
- Enter README APP on the command-line to review information about default applids and understand how the Configuration Tool processes VTAM applids.

Network ID

Identify your VTAM network, as defined in the NETID parameter of the VTAMLST startup member ATCSTRnn. This parameter is required for SNA communications. If none of the products or components you intend to configure in this runtime environment require SNA communications, clear this field.

Logmode table

Type the logmode table name for LU6.2 logmode entries. The default is KDSMTAB1. If none of the products or components you intend to configure in this runtime environment require SNA communications, clear this field.

This table is assembled into the system library (usually SYS1.VTAMLIB) that contains VTAM logmode tables.

LU6.2 logmode

Type the LU6.2 logmode for this runtime environment. The default is CANCTDCS. If none of the products or components you intend to configure in this runtime environment require SNA communications, clear this field.

TCP/IP communication values

The values you specify here become the default TCP/IP values for the products and components configured in the runtime environment.

Tip: In the TCP/IP communication values section of the panel, the fields Hostname, Address, and Started task support system variables defined in the system variable parameter member, but the fields require literal values for input on this panel. Specify the literal values here, and specify the symbolic variables in the system variable parameter member. For more information, issue the README SYS command from any Configuration Tool panel.

Hostname

The TCP/IP host name or dotted-decimal IP address of the z/OS system where the runtime environment is being defined. To obtain the host name and IP address, enter TSO HOMETEST at the command-line.

If the z/OS domain name resolver configuration specifies a search path that includes the target domain suffix, specify only the first qualifier of the host name. (Example: sys is the first qualifier of the fully qualified host name sys.ibm.com.) Otherwise, specify the fully qualified host name.

Started task

Identifies the TCP/IP stack to be used. If the LPAR contains a single TCP/IP stack, accept the default value, which is the wildcard asterisk character (*), to select the first TCP/IP stack that was started.

If the LPAR contains more than one TCP/IP stack, you can specify the started task name of the TCP/IP stack you want to use; or you can specify the number sign
(#), which is translated to a blank and allows the TCP/IP environment to choose the
stack to use, either through TCP/IP definitions or through the use of the
SYSTCPD DD statement.

**Tip:** Whichever method is used to select a TCP/IP stack in a multi-stack
environment, the Tivoli Management Services components continue to use that
stack, even if a different stack becomes the primary stack. Therefore, in a
multi-stack environment, it is best to specify the started task name of the TCP/IP
stack to be used, rather than specifying a wildcard or a blank.

If IP domain name resolution is not fully configured on the z/OS system, the
SYSTCPD DD statement is required (see “Add support for the SYSTCPD
DDNAME in the started tasks” on page 129).

**Port number**
The number of the well-known port to be used for IP communications. The default
is 1918. (See “Port number assignments” on page 40.)

i. After you have finished defining the runtime environment, press Enter. A third Add Runtime
Environment panel is displayed (Figure 35).

```
COMMAND ==>
If you require TN3270E Telnet session link support override, complete these
values:
    Hostname ==> 
    Port number ==> 
    LUGROUP ==> 

If any products in this RTE require Unix System Services (USS) directories
be created, specify the RTE HFS/zFS home directory:
    RTE home directory ==> /rtehome________________________ (rtehome)
    (Press Fl=Help for rtehome RTE path name considerations.)
```

Enter=Next  F1=Help  F3=Back

Figure 35. Add Runtime Environment (3 of 3) panel for a full or sharing runtime environment

j. Complete this panel if at least one of the products to be configured in this runtime environment
has at least one of the following conditions:

- The product supports the Dynamic XE to 3270 (Classic) linking feature, and you must override
  the default values.
  The default values or the override values you specify on this panel are displayed during TN3270
  logon and can be modified then for an individual TN3270 session.
- The product requires UNIX System Services (USS) directories to be created (for example, if the
  agent stores files in the USS file system).

Use the following information to complete this panel:

**Hostname**
If the LPAR in which you are defining a runtime environment does not have an active
Telnet listener, specify the network address of an LPAR that does have an active Telnet
listener. A network address can be specified as one of the following values:
• Fully qualified hostname (for example, sys.ibm.com)
• First qualifier of the fully qualified hostname (for example, sys for sys.ibm.com)
• 32-bit IPv4 address in dotted decimal notation (for example, 9.67.1.100)

To get this value, issue the TSO HOMETEST command on the LPAR of the Telnet listener.

Port number
The default port number of the Telnet listener is 23. To override this value, specify the port number of the Telnet listener.

LUGROUP
The Dynamic XE to 3270 (Classic) linking feature requires the VTAM Unformatted System Services screen to accept a LOGON APPLID() DATA() command. If the default Telnet USS screen does not accept this command, supply the name of a Logical Unit (LU) group that does accept it. The TN3270 session will be joined to that LU group.

RTE home directory
If a product to be configured in the runtime environment requires USS directories to be created, specify the main Hierarchical File System/zSeries File System (HFS/zFS) directory. The directory name must begin with a slash (/) and represents the #rtedir value (the name of the file system directory where the runtime environment is to be created). Examples:

/omonrte/rte1/kan/bin
/omonrte/etc/rte1/kan/bin
/omonrte/rte1/kds/support/TEMS

in the following format:
#rtedir/#rtename/#ppp/#filetype
#rtedir/#rtename/#ppp/#workarea (/#filetype)

where:
#rtedir = /omonrte or /omonrte/etc
#rtename = rte1
#ppp = kan or kds
#filetype = bin
#workarea = /support/TEMS

Security tip:
Some products create their own USS directories with a default group and user USS directory permissions of read/write/execute (777). Other products require a higher level of permissions for the USS directories and subdirectories they use. To implement a more secure access scheme for the USS directories of the runtime environment, use group-based security. For detailed information, enter README USS on the command-line.

k. Press Enter to return to the Runtime Environments (RTEs) panel (Figure 31 on page 111).

Tip:
Enter V (View values) to verify the runtime environment information and U (Update) to make any necessary changes.

3. Build the runtime libraries.
This step generates JCL to allocate the required runtime libraries for the selected product. You must perform this step for all runtime environments you create, including base runtime environments. You must also perform this step when you are installing a new product into an existing runtime environment. Note that if you are setting up a base runtime environment, only shared libraries are allocated, not product-specific libraries.
The libraries built by this step depend on the product selection made on the Product Selection Menu. If you want to add other products to the runtime environment, you must select each from the Product Selection Menu and perform the Build option for each runtime environment in which you want to configure it.

a. On the Runtime Environments (RTEs) panel, type B to the left of the name of the runtime environment for which you want to build the libraries, and press Enter.
   The JCL to allocate the runtime libraries is displayed.

b. Review the JCL, edit if necessary, and submit the job.
   You might want to change the jobname to match the member name so that you can identify the jobs later on.

c. Verify that the job completes successfully.
   Expect to receive a return code of zero. If you do not, check the log to diagnose errors, and then rerun the job.

d. Press F3 (Back) to return to the Runtime Environments (RTEs) panel.
   If you have built the runtime libraries for a base runtime environment, you must add a sharing runtime environment for product-specific libraries. Go back to Step 2 and complete the steps for defining a sharing runtime environment and building its libraries.
   Note that placing B beside a sharing-with-base or a sharing-with-full runtime environment builds the libraries for both the shared runtime environment (base or full) and the sharing runtime environment.

What to do next

After you have set up a runtime environment, you can proceed to configure products in it by following the product-specific instructions in the configuration guide for each product.

Step 5. Configure your products and components in the runtime environment

This step contains best practices that apply to the configuration of all products. Read these topics before you start configuring your products according to the instructions in the individual product configuration guides.

Configure a monitoring server

The product configuration menu for every OMEGAMON XE monitoring agent and for Tivoli Management Services on z/OS begins with an option to configure a monitoring server. IBM Tivoli OMEGAMON XE on z/OS and IBM Tivoli OMEGAMON XE for Storage on z/OS require that the agents run in the same address space as a monitoring server, so you must configure a monitoring server first in every runtime environment in which you configure an OMEGAMON XE on z/OS or OMEGAMON XE for Storage monitoring agent. Monitoring agents that can be configured to run in their own address space (stand-alone) do not require that a monitoring server be configured in the same runtime environment.

About this task

If you are configuring a monitoring server in the runtime environment, configure it before you configure any monitoring agents. Do not configure more than one monitoring server in an LPAR, unless you are configuring a high-availability hub in one runtime environment and a remote monitoring server and agents in another runtime environment on the same LPAR. For instructions on configuring a high-availability hub, see IBM Tivoli Management Services on z/OS: Configuring the Tivoli Enterprise Monitoring Server on z/OS.
Set appropriate parameter values

In most cases, you set parameter values for a runtime environment and for the products and components in the runtime environment by accepting defaults or supplying values in the Configuration Tool interactive panels, and then you use batch mode and the batch parameter member to propagate the parameter values to other LPARs. However, in special cases, under the guidance of IBM Software Support, you might have to add parameters or parameter options that are not presented in the Configuration Tool interactive panels.

About this task

The topics in this section explain how to use the Configuration Tool to obtain parameter reports, modify parameter values in already-configured runtime environments, and add parameters to members of your runtime libraries:

Obtaining parameter reports

Parameter reports, also called parameter maps, list all parameters associated with the components and products that were installed into the target libraries by SMP/E, and gives a short description and other basic information about each parameter. You generate parameter reports, from RTE Utility Menu panel.

About this task

Complete the following steps to generate a parameter report.

Procedure

1. Select option 11 (Verify Configuration and Generate Parameter Map) on the RTE Utility Menu panel

   Figure 36. Verify Configuration and Generate Parameter Map panel

   KCIPPV - Verify configuration and generate parameter map / RTE: rte -------

   1 Verify configuration (batch)

      This process can only be run in batch. In a complex environment, it might be a long running job.

   2 Generate parameter map (foreground)

      This process can only be run in foreground. In a complex environment, it might run for several minutes.

      See F1=Help for pre/post configuration considerations.

   Enter=Next  F1=Help  F3=Back

2. The type of parameter map you can generate depends on whether you have already configured components and products in the selected runtime environment. Choose one of the following options.
   • To generate a parameter map after creating a runtime environment but before configuring any components or products in it, select option 2 (Generate parameter map).
   • To generate a parameter map after configuring components and products in the runtime environment, follow these steps:
     a. Select option 1 (Verify configuration) to run a batch job that produces a configuration verification report for the runtime environment.
        The Configuration Tool displays the JCL for the shilev.INSTJOBS(IVP@ssss) batch job, where ssss is the JCL suffix for the runtime environment.
     b. Submit the configuration verification batch job.
The configuration verification job generates a report that lists the required data sets and members; indicates the batch job that creates each one; identifies which data sets and members are missing, and which batch jobs are missing or did not complete successfully; and specifies which of the missing or unsuccessful batch jobs and which of the missing data sets and members are required and which are optional. If a large number of components and products are configured in the runtime environment, the configuration verification batch job might take a long time to run.

c. Check to make sure that the configuration verification batch job has run successfully and that the configuration verification report is stored in the `shlev.INSTJOBS.IVP$ssss(IVP1ssss)` member.

d. Select option 2 (Generate parameter map) on the Verify Configuration and Generate Parameter Map panel.

The Configuration Tool generates a parameter map listing all parameters for the configured products and components in the runtime environment, along with the value for each configured parameter and other post-configuration parameter information such as the name of the configuration job that stored each parameter and data about completion of each job and each step. The job and step completion codes reported in the parameter map are retrieved from the configuration batch job log stored in the `shlev.INSTLOG` data set.

The parameter map is generated in the foreground and is displayed as soon as it is complete. If a large number of components and products are configured in the runtime environment, generating the parameter map might take several minutes.

The parameter map for a runtime environment is stored in the `shlev.INSTJOBS.IVP$ssss(IVP1ssss)` member.

The top of each parameter map provides the following information to help you locate key data sets and files for a configured runtime environment:

- **RTE Name:** `rte_name`
- **JCL Suffix:** `ssss`
- **INSTJOBS:** `shlev.INSTJOBS`
- **INSTLOG:** `shlev.INSTLOG`
- **IVP Detail:** `shlev.INSTJOBS.IVPrte_name(IVPDssss)`
- **INSTDATA:** `shlev.INSTDATA`
- **TKANCUS:** `shlev.TKANCUS`

See the following example of the kinds of information the parameter map provides for each configured parameter:

<table>
<thead>
<tr>
<th>NAME</th>
<th>KDS_CMS_STC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>'Server started task name'</td>
</tr>
<tr>
<td>V ARNAME</td>
<td>DSSPPMEM</td>
</tr>
<tr>
<td>VALUE</td>
<td>CIDSST</td>
</tr>
<tr>
<td>PANELS</td>
<td>1 PANEL DESCRIPTION</td>
</tr>
<tr>
<td>SKELS</td>
<td>1 KDS621SC</td>
</tr>
<tr>
<td>MEMBERS</td>
<td>1 MEMBER JOB STEP LIB JOBNAME JOB# HI-CC</td>
</tr>
</tbody>
</table>

| CIDSST | D5#3ssss STEP1 | RKANSAMU userid$ST J19881 00000 |
| CIDSST | D5#4ssss STEP1 | RKANSAMU userid$ST J19916 00000 |
| CIDSST | MQ#4ssss STEP1 | RKANSAMU userid$ST J19943 00000 |

**Name**  Batch parameter name

**Description**  Brief text describing the parameter

**V ARNAME**  For IBM internal use only

**Panel**  Panel ID of a Configuration Tool interactive panel on which the parameter value is set

**Description**  Panel name of a Configuration Tool interactive panel on which the parameter value is set
SKELS
For IBM internal use only

Member
Name of a runtime member where the parameter value is stored

Job
Name of a batch job that generates the parameter value from Configuration Tool input

Step
Name of a job step that generates the parameter value from Configuration Tool input

Lib
Runtime library where the parameter value is stored

Jobname
ID of the job card submitter

Job#
Number of the submitted job

Hi-CC
Highest condition code resulting from submission of the job

Tips:
• A parameter map is specific to one runtime environment. Generate a separate parameter map for each runtime environment.
• Only one parameter map can be stored for each runtime environment. If you want to keep a previously generated parameter map (for example, if you want to keep the pre-configuration map when you generate a post-configuration map), back it up before you generate a new parameter map for the runtime environment.
• If you subsequently reconfigure components or products, or configure additional ones, in the runtime environment, regenerate and rerun the configuration verification job and then regenerate the parameter map.
• For detailed information about the configuration verification report and parameter map, enter README IVP at the command prompt from any Configuration Tool panel.

Modifying configured parameter values
After you finish configuring your products, you might have to change the value of one or more parameters. For example, it is a best practice to disable security validation on the hub monitoring server until after you finish configuring all components and verifying that you can start them and that they can communicate with each other. At that point, you probably want to enable security validation on the hub. You do this by changing the value of a parameter.

About this task
It is possible to change a parameter value by editing the runtime member where the value is stored. However, if you edit a runtime member directly, your changes are overwritten when you reinstall or reconfigure the product. Therefore, it is best to use the interactive panels of the Configuration Tool to change parameter values that were originally set in the Configuration Tool. If you want to make the same change on every LPAR, you can then use batch mode to propagate the change to your other runtime environments.

Adding, changing, or deleting nonstandard parameters in a runtime environment
In special cases, under the guidance of IBM Software Support, you might have to add, change, or delete parameters or parameter options that are not presented in the Configuration Tool interactive panels.

For example, "Decision 6: How to set up communications between components" on page 36 explains changes to the KDE_TRANSPORT environment variable that might be required in certain environments. Some KDE_TRANSPORT options, such as the selection of communication protocols to be used by monitoring servers and monitoring agents, can be set in the fields of the Specify Communication Protocols panel (for a monitoring server) or the Specify Agent Address Space Parameters panel (for a stand-alone monitoring agent). However, some KDE_TRANSPORT options do not have fields in those panels.
About this task

Parameters or parameter options that do not correspond to fields in the Configuration Tool interactive panels are known as nonstandard parameters. To add, modify, or delete a nonstandard parameter, use the Specify Nonstandard Parameters panel.

Important: Use extreme caution in specifying nonstandard parameters. No error checking is provided and syntax is not validated. The presence of the data set and member specified is not validated. If faulty syntax or other errors cause your edits to fail, no warning or error message is issued. Use this facility only under the guidance of IBM Software Support.

Procedure

1. To display the Specify Nonstandard Parameters panel (shown in Figure 37), press F5=Advanced from the Specify Advanced Configuration Values panel for a monitoring server or the Specify Advanced Agent Configuration Values panel for a monitoring agent.

```
KDSPNST1 ------ SPECIFY NONSTANDARD PARAMETERS / RTE: rte ---------------
OPTION ===>  
This panel is used to add nonstandard parameters to runtime members.
Warning: This should be used under the guidance of IBM Software Support.
Parameter: _______________________________________
New Value: __________________________________________
Old Value (if replacing): _____________________________
Low-level data set qualifier: _______ Member: ________
Parameter: _______________________________________
New Value: __________________________________________
Old Value (if replacing): _____________________________
Low-level data set qualifier: _______ Member: ________
Parameter: _______________________________________
New Value: __________________________________________
Old Value (if replacing): _____________________________
Low-level data set qualifier: _______ Member: ________
Enter=Next  F1=Help  F3=Back
```

Figure 37. Specify Nonstandard Parameters panel

2. Use the following information to complete the Specify Nonstandard Parameters panel:

   **Parameter**
   The name of the configuration parameter to be added, replaced, or deleted.

   **New Value**
   The value you want the parameter to have. Include format characters. (For example, if the format in the runtime member is `parameter=value`, precede the value with an equal sign; or if the format is `parameter(value)`, surround the value with parentheses.) If you want to delete an existing parameter, leave the **New Value** field blank.

   **Old Value (if replacing)**
   The existing parameter value that you want to replace or delete. Include format characters. The character string in the **Old Value** field must match exactly the existing value for the parameter in the runtime member, or the value is not replaced.

   **Low-level data set qualifier**
   The low-level qualifier of the data set containing the member with the parameter to be added, replaced, or deleted.

   **Member**
   The member containing the parameter to be added, replaced, or deleted.
For each runtime environment, you can specify a maximum of 18 parameters for each monitoring
server address space, a maximum of 6 parameters for each stand-alone OMEGAMON XE monitoring
agent address space, and a maximum of 12 parameters for each OMEGAMON (3270) or
OMEGAMON II components.

Examples:

Adding a parameter:

```
Parameter: PARM1
New Value: =YES
Old Value (if replacing): ____________________________
Low-level data set qualifier: RKANPARU Member: KDSENV
```

Changing the value of an existing parameter:

```
Parameter: PARM1
New Value: =NO
Old Value (if replacing): =YES
Low-level data set qualifier: RKANPARU Member: KDSENV
```

Deleting a parameter:

```
Parameter: PARM1
New Value: ____________________________
Old Value (if replacing): =YES
Low-level data set qualifier: RKANPARU Member: KDSENV
```

For examples of commonly specified nonstandard parameters, see the "Configuration, and
Deployment: Highlights of Common z/OS Configuration Tool Enhancements" Technote at

What to do next

After you have added nonstandard parameters, you must regenerate the Configuration Tool pp#3xxx job
created in the “Create runtime members” step for the monitoring server or monitoring agent, as
applicable.

**Verify the configuration before loading the runtime libraries**

You can use the Verify configuration option of the RTE Utility Menu to verify the creation and successful
execution of all configuration batch jobs and the creation of all required runtime data sets and members.
You can run the configuration verification job at any time; it is a good idea to run it as soon as you finish
configuring a runtime environment and before you load the runtime libraries. Additionally, the
configuration verification job is run automatically when you load the runtime libraries. This job can be
run in batch mode only; it cannot be run in foreground. If a large number of components and products
are configured in the runtime environment, the configuration verification batch job might take a long
time to run.

What to do next

After you run the configuration verification job, you can also generate a detailed parameter map for your
newly configured runtime environment. You can find more information about parameter maps in
Figure 50 on page 233 and "Obtaining parameter reports” on page 122.

Enter README IVP on the command-line for more information about the configuration verification
report. You can also find more information in “Runtime environment utilities” on page 232.
If there are errors, go back and correct them, then run the verification job again. If there are no errors, load the runtime libraries.

**Load runtime libraries**

The last step in configuring each product using the Configuration Tool is loading its runtime libraries. Each time you run the Load function, the Configuration Tool loads the runtime libraries for the entire runtime environment. If you are configuring multiple products in the runtime environment at one time, you can wait until you have configured all the products and then perform the Load step. However, if different people are configuring different products, or if you are configuring products in the runtime environment at different times, load the runtime libraries after you configure each product.

**About this task**

For more information on the Load function, see "The Runtime Environments (RTE) panel" on page 239.

**Eliminate duplicate Complete the Configuration steps**

At the end of each product configuration menu is a step to display a list of Complete the Configuration tasks. You do not have to perform these tasks immediately after you finish configuring each product. If you are configuring multiple products within a runtime environment, you can postpone these additional tasks and perform them after you finish configuring all the products that you want to configure at this time. You can use the Configuration Tool to generate a concatenated list of those tasks for all the products in a runtime environment.

For information about generating a consolidated list of configuration completion tasks, see "Runtime environment utilities" on page 232.

**About this task**

A number of steps are listed in the Complete the Configuration list for every product. The following steps, which appear in every list, can be performed once for all the products in a runtime environment:

- "Add support for the SYSTCPD DDNAME in the started tasks" on page 129.
- "Copy started task procedures to your procedure library" on page 130.
- "Copy the VTAM definitions to your system VTAMLST" on page 131.
- "Vary the VTAM major node active" on page 131.
- "APF-authorize the runtime load libraries" on page 132.
- "Complete the configuration for the OMEGAMON subsystem" on page 132.
- "Complete the configuration for OMEGAMON monitoring agents" on page 134.
- "Manually install application support" on page 134.
- "Verify the configuration" on page 134.
- "Enable security" on page 137.
- "(Optional) Enable maintenance of the historical data store" on page 145.
- "(Optional) Enable historical data collection" on page 146.
- "Run the ITMSUPER Tools (optional)" on page 146.

If you are configuring several products in a runtime environment at one time, or if you are using batch processing to replicate existing runtime environments, you can use the following procedure.

If different people are configuring different products, or if you are configuring products at different times, perform the Load and Complete the Configuration steps for each product as you configure it to avoid missing required steps. Chapter 6, "Completing the configuration," on page 129 describes the actions...
required to complete each of the required steps and the recommended order in which to complete them.

**Procedure**

1. Configure your products in the runtime environment, or configure the runtime environment by using the CICATB batch job. Until you have finished configuring all products in the Configuration Tool, postpone loading the runtime libraries and performing any Complete the Configuration steps outside the Configuration Tool for any products.

2. Run the runtime environment Load job (Load All Product Libraries After SMP/E). If you use the CICATB batch job, this step is completed for you.

3. Generate the concatenated list of Complete the Configuration steps for the runtime environment. For more information, see “Runtime environment utilities” on page 232.

4. Review and perform the Complete the Configuration steps.

5. Review and perform the product-specific Complete the Configuration steps for each product configured in the runtime environment. Skip the common steps that you have already performed.
Chapter 6. Completing the configuration

After you finish the configuration steps described in either Chapter 4, “Configuring products using the PARMGEN method,” on page 63 or Chapter 5, “Configuring products with the Configuration Tool,” on page 103, you must take additional steps to complete the configuration of your runtime environment. The steps that you are required to complete depend on the steps that you have already taken, the configuration options that you have chosen, and the components or products that you are configuring. Some of the steps are required and others are optional.

About this task

Many of the steps required are the same for multiple products and can be completed all at one time for all the products. To help you configure your components correctly, work through the topics in this section in conjunction with the corresponding section within the product-specific planning and configuration guides.

In the Configuration Tool, the Complete the configuration option displays a list for the steps required to complete the configuration outside the tool. In the PARMGEN method, the KDSDMG* jobs available in the WKANSAMU library provide the same information.

In the Configuration Tool, the Complete the configuration option displays a list for the steps required to complete the configuration outside the tool. In the PARMGEN method, the KDSDMG* jobs available in the WKANSAMU library provide the same information.

The topics in this section give basic instructions for completing the configuration of common or shared components within your runtime environment. On completion, be able to verify that all components have started correctly and be able to complete the configuration of optional advanced features such as collecting historical data and enabling security.

Add support for the SYSTCPD DDNAME in the started tasks

SYSTCPD explicitly identifies which data set to use to obtain the parameters defined by TCPIP.DATA when no GLOBALTCPIPDATA statement is configured. If a monitoring server is using any of the IP.UDP-related or IP.PIPE-related communication protocols for connection, but the IP domain name resolution is not fully configured on the z/OS system, the SYSTCPD statement must be supported by the monitoring server and the monitoring agents that report to it. This step is applicable only if you configured your runtime environment by using the Configuration Tool.

Note: This step is applicable only if you configured your runtime environment by using the Configuration Tool (ICAT).

About this task

If you are certain that SYSTCPD is not required in your environment, you can skip this step. However, you might gain a small performance benefit by avoiding multiple dynamic data set allocations if you supply a SYSTCPD DD statement.

Procedure

To support SYSTCPD, uncomment the following statement in the started task members in RKANSAMU and provide the name of the SYSTCPD data set:

```bash
//*SYSTCPD DD DISP=SHR,
//* DSN=TCPIP.SEZAINST(TCPDATA)
```

The name of the SYSTCPD data set is installation-specific. Get the correct specification from your network administrator.
What to do next

The next step is to copy the started tasks from RKANSAMU to PROCLIB.

If you reconfigure products, you must uncomment the statement again and refresh the monitoring server and monitoring agent started tasks in PROCLIB.

Copy started task procedures to your procedure library

During configuration, the configuration software creates a number of started task procedures in the rhilev.rte.RKANSAMU data set (and in the rhilev.rte. RKD2SAM data set if the IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS or IBM Tivoli OMEGAMON XE for DB2 Performance Monitor on z/OS product or both are being configured). If you configured your runtime environment by using the Configuration Tool, these procedures must be copied to your procedure library.

Note: This step is applicable only if you configured your runtime environment by using the Configuration Tool (ICAT).

About this task

The following procedures must be copied to your procedure library under a user ID with Write authority to the PROCLIB data set:

- the monitoring server started task (default: CANSDSST)
- the maintenance procedure for the persistent data store (KPDPROC1). The KPDPROC1 procedure is present only if you configured a persistent data store in the runtime environment.
- started task procedures for any monitoring agents configured in their own address spaces in the runtime environment.

You can copy the procedures manually or use the runtime environment sample system procedure copy JCL within the Configuration Tool. To copy the procedures by using the utility, perform the following procedure.

Be careful not to overwrite any members in the data set that have already been modified. If necessary, copy the started task procedures manually.

Procedure

1. On the Runtime Environments (RTE) panel, type Z (Utilities) next to the runtime environment whose started tasks you want to copy and press Enter.
2. From the RTE Utility Menu, enter 6 (Generate sample system procedure copy JCL).
3. On the next panel, type the name of your procedure library, for example, SYS1.PROCLIB, and press Enter.
   The JCL (CB#PIDD) that creates the copy job is displayed.
4. Edit the JCL as necessary and submit the job. Expect a return code of zero.
   This step generates a member called KCISYPJB in the rhilev.rte.RKANSAMU and RKD2SAM libraries. This member lists the started tasks for all products configured in the runtime environment.
5. Edit the JCL if necessary and submit KCISYPJB from a user ID that has authority to the PROCLIB data set.
   This job copies all the required started tasks to the specified procedure library. Expect a return code of zero.
Copy the VTAM definitions to your system VTAMLST

During configuration, the Configuration Tool also creates VTAM definitions in the rhilev.rte.RKANSAMU data set (and in the rhilev.rte.RKD2SAM data set if the IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS or IBM Tivoli OMEGAMON XE for DB2 Performance Monitor on z/OS product or both are being configured). Under a user ID with write authority to the VTAMLST data set, copy the VTAM major nodes to your system VTAMLST. You can copy the definitions manually or use the Configuration Tool utility to copy them.

Note: This step is applicable only if you configured your runtime environment by using the Configuration Tool (ICAT).

About this task

To copy the definitions by using the utility:

1. On the Runtime Environments (RTEs) panel, type Z next to the name of the runtime environment whose nodes you want to copy and press Enter.

2. From the RTE Utility Menu, select option 7 (Generate sample system VTAMLST copy JCL) and then press Enter.

3. On the next panel, type the name of your VTAM node, and press Enter.

   The JCL to create the KCISYNJB member is displayed.

4. Edit the JCL and submit the job. Expect a return code of zero.

   The KCISYNJB member is created in the rhilev.rte.RKANSAMU and RKD2SAM libraries.

5. Edit the KCISYNJB member and submit the job from a user ID that has authority to the VTAMLST data set.

   This job copies all the required VTAM definitions to the specified VTAMLST. Expect a return code of zero.

   Be careful not to overwrite any members in the data set that have already been modified. If necessary, copy the definitions manually.

Tip: You can create a single VTAM major node that contains applids for all the products configured in the runtime environment. Creating a single node can simplify copying and activation. To create a single major node:

1. On the Runtime Environments (RTEs) panel, type Z next to the name of the runtime environment whose nodes you want to copy and press Enter.

2. On the RTE Utility Menu, type 4 (Create VTAM major node) and press Enter.

3. On the Create VTAM Major Node panel, type a name for the node and press Enter.

   The JCL for the job (CB#8ssss) that creates the single major node is displayed

4. Review the JCL, and then submit the job to create the node.

This procedure must be repeated if the runtime environment values change. If a new product is added to the runtime environment or an existing product is reconfigured to change any of the system variable applid values, rerun the CB#8ssss job.

Vary the VTAM major node active

After the VTAM major node has been copied to the system VTAMLST, the node must be varied active.

Note: This step is applicable only if you configured your runtime environment by using the Configuration Tool (ICAT).
About this task

Issue the following command to vary a VTAM major node active:

```
V NET,ACT,ID=nodeid
```

If you have not created a single VTAM major node for all the products, repeat the command for each node.

---

**APF-authorize the runtime load libraries**

The runtime load libraries created by the configuration software must be added to your list of APF-authorized libraries.

About this task

This step is applicable only if you configured your runtime environment by using the Configuration Tool, or if you configured your runtime environment by using the PARMGEN Workflow user interface and did not enable the optional RTE_X_STC_INAPF_INCLUDE_FLAG parameter to generate inline APF-authorize statements in the product started tasks.

Procedure

Add the following runtime load libraries to your list of APF-authorized libraries.

- `rhilev.rte.RKANMOD`
- `rhilev.rte.RKANMODU`
- `rhilev.rte.RKANMODL`
- `rhilev.rte.RKANMODP`

Any other runtime libraries concatenated in the STEPLIB DDNAME or in the RKANMODL DDNAME of started tasks must also be APF-authorized.

If the runtime environment shares target libraries that were installed by SMP/E, you must also APF-authorize the following libraries:

- `thilev.TKANMOD`
- `thilev.TKANMODL`
- `thilev.TKANMODP`

What to do next

Some monitoring agents require additional target libraries. Check the product-specific planning and configuration guides for a complete list of runtime libraries to include.

---

**Complete the configuration for the OMEGAMON subsystem**

Tasks described in this section are required to complete the configuration of the OMEGAMON subsystem. This component is not used by all OMEGAMON monitoring agents.

About this task

This subsystem is required if you have configured any of the following OMEGAMON monitoring agents in the runtime environment:

- IBM Tivoli OMEGAMON XE for CICS on z/OS (optional)
- IBM Tivoli OMEGAMON XE for IMS on z/OS
- IBM Tivoli OMEGAMON XE for DB2 Performance Expert/Performance Monitor on z/OS

---
Procedure

- Update the IEFSSNxx member of the SYS1.PARMLIB.
  - If you are creating a new runtime environment, the appropriate IEFSSNxx member of SYS1.PARMLIB must be updated to identify the OMEGAMON subsystem to the z/OS system. Member KCNDLSSI in the rhilev_rte.RKANSAMU data set contains a sample IEFSSNxx update. In addition to identifying the OMEGAMON subsystem to the z/OS system, this sample causes an automatic start of the subsystem address space.
  - If you have not already done so, copy the OMEGAMON subsystem started task (default: CANSCN) from the RKANSAMU library to PROCLIB. (If you used the PARMGEN configuration method, this step has already been done for you.)
    - You can change the name of this JCL procedure to any name that meets the installation standards of your site. However, do not use the name of the OMEGAMON subsystem (subsystem ID) as the name of your JCL procedure.
  - Copy the KCNDLINT load module to an appropriate library in the LINKLIST. Follow the installation standards at your site in making the decision about an appropriate library.

Use the RESTART parameter

The optional RESTART parameter forces the subsystem to complete initialization by passing checks designed to prevent the start of a second address space.

About this task

The RESTART parameter requires the FORCE operand, as in this example:

```
START CANSCN,RST=',RESTART=FORCE'
```

You can stop the subsystem by issuing the z/OS STOP command, as in this example:

```
STOP CANSCN
```

Important:

Use the RESTART parameter only if the subsystem address space ends abnormally and subsequent attempts to start the subsystem result in the CNDL018I message; this message indicates that the subsystem is already active. Verify that the subsystem address space named in the CNDL018I message is not active before using the RESTART parameter. If the RESTART parameter is used when the subsystem is already active, results are unpredictable. You must restart the subsystem to pick up any maintenance that was installed.

The OMEGAMON subsystem cannot be executed as a batch job; it must be invoked as a started task.

Verify the OMEGAMON subsystem installation

To verify a first-time installation, either issue the z/OS SETSSI command, or perform an IPL.

About this task

If an IPL is performed, the SYSLOG contains the CNDL184I message, informing you that the OMEGAMON subsystem initialization routine has completed. If the message is not displayed after an IPL, check your update to the IEFSSNxx member of SYS1.PARMLIB.

If you chose automatic startup of the OMEGAMON subsystem address space, the following messages are displayed: CNDL001, CNDL190I, CNDL034I, and CNDL027I. If the keyword parameter form of the
IEFSSN.xx PARMLIB member was used, you can choose to issue the SETSSI command to define the subsystem (rather than IPL). If you choose to use the SETSSI command, or if you did not choose automatic startup at IPL, you can issue the MVS START command to start the subsystem. The following messages are displayed: CNDL001I, CNDL190I, CNDL034I, and CNDL027I.

Complete the configuration for OMEGAMON monitoring agents

Before you verify the configuration of the monitoring agents in the runtime environment, consult the corresponding section within each product-specific planning and configuration guide to ensure that you have completed all the required steps.

Manually install application support

If you did not enable the self-describing agent feature, you must manually install application support by using the OMEGAMON Application Support DVD. Under this process, the application support is installed onto a distributed platform and then seeded to the hub monitoring server. To complete the seeding process, the hub monitoring server must be recycled. Application support for the Tivoli Enterprise Portal Server and Tivoli Enterprise Portal clients can also be added by using the OMEGAMON Application Support DVD.

About this task

For detailed instructions on installing application support to a monitoring server on z/OS systems, see IBM Tivoli Management Services on z/OS: Configuring the Tivoli Enterprise Monitoring Server on z/OS. If your hub monitoring server is on a distributed platform, see the IBM Tivoli Monitoring: Installation and Setup Guide.

What to do next

After you have installed application support, recycle the components on which the support was installed.

Install language support

If you want the Tivoli Enterprise Portal user interface, online help, and expert advice for your products to be displayed in a language other than English, you must also add language support to the portal server.

About this task

For instructions on adding language support, see the IBM Tivoli Monitoring: Installation and Setup Guide.

Verify the configuration

After completing the post configuration tasks described in this section, you are ready to confirm that the products and components can run and communicate with each other.

About this task

Procedure

1. If you have not already done so, run the configuration verification job in each runtime environment to confirm that all the required data sets, members, and configuration jobs were created, and that the jobs completed successfully.
   - With the PARMGEN configuration method, the configuration verification job is included in "Step 11. Submit batch jobs to complete the PARMGEN setup" on page 98.
   - With the Configuration Tool method, you can run the configuration verification job from the RTE Utility menu. See "Runtime environment utilities" on page 232.
If the configuration verification report shows that a required item is missing or that a job failed, repeat the necessary configuration steps.

2. After you have verified all runtime environments, validate the operation of all components and the communication between them as follows:

   Table 8. Validation of the runtime environment

<table>
<thead>
<tr>
<th>Configuration method</th>
<th>Steps</th>
</tr>
</thead>
</table>
| If you used the PARMGEN configuration method, a job containing all the tasks configured within the runtime environment is available to use. Complete the adjacent steps. | 1. If your hub is on a distributed system or another runtime environment, start it. For more information, see the IBM Tivoli Monitoring: Installation and Setup Guide.
2. Run the composite job to start all tasks within this runtime environment:
   `/s xxxxSTRT`
   where xxxx = %RTE_STC_PREFIX% (The default is CANS).
   **Note:** An equivalent xxxxSTOP job is generated for stopping all tasks within the runtime environment.
3. In the RKLVLOG for each monitoring agent address space, look for the messages that indicate successful startup. Also look for messages indicating that the monitoring agents or remote monitoring server have successfully connected to the hub, if applicable. Consult the product-specific Planning and Configuration guides for more information.
4. If you have enabled the self-describing agent feature, check the status of the self-describing agent application support at the hub monitoring server log, monitoring agent logs and finally any remote agent logs to ensure that the self-describing agent process has completed. For more information, see the IBM Tivoli Monitoring: Installation and Setup Guide.
### Table 8. Validation of the runtime environment (continued)

<table>
<thead>
<tr>
<th>Configuration method</th>
<th>Steps</th>
</tr>
</thead>
</table>
| If you prefer to start each job in turn or used the Configuration Tool configuration method, complete the adjacent steps. | 1. If IBM Tivoli OMEGAMON XE for Storage on z/OS is one of your installed products, start the OMEGAMON subsystem first: /S CANSCN  
2. Start the hub monitoring server. If your hub is on a z/OS system, start the monitoring server started task: /S CANSDSSST  
   If your hub is on a distributed system, see the IBM Tivoli Monitoring: Installation and Setup Guide.  
3. If you have configured a remote monitoring server on a z/OS system, start its started task: /S CANSDSSST  
   For agents that run inside the monitoring server address space, starting the monitoring server automatically starts the agents.  
4. In the RKLVLOG for each monitoring server address space, look for the messages that indicate successful startup. Also look for messages indicating that the remote monitoring server has successfully connected to the hub, if applicable.  
5. If OMEGAMON XE for Storage on z/OS is not one of your installed products and if any of your monitoring agents on z/OS require the OMEGAMON subsystem, start it now. See the configuration documentation for each monitoring agent to verify if the OMEGAMON subsystem is required.  
6. Start each monitoring agent on z/OS that runs within its own address space. Some monitoring agents may have specific instructions or a sequence in which tasks should be started. Consult the product-specific Planning and Configuration guides for information. If you have enabled the self-describing agent feature, check the status of the self-describing application support at the hub monitoring server log, monitoring agent logs and finally any remote agent logs to ensure that self-describing process has completed. For more information, see the IBM Tivoli Monitoring: Installation and Setup Guide.  
7. In the RKLVLOG member for each monitoring agent address space, look for the messages that indicate successful startup. Consult the product-specific Planning and Configuration guides for information. |

3. Ensure that data collected by the monitoring agents is available on the user interface that you have chosen to use as follows:
Table 9. Data validation on user interfaces

<table>
<thead>
<tr>
<th>User interface</th>
<th>Steps</th>
</tr>
</thead>
</table>
| Tivoli OMEGAMON enhanced 3270 user interface | 1. If you have not already done so, start the OMEGAMON enhanced 3270 user interface address space: `/S CANSTOM`
2. Check the SYSPRINT of the OMEGAMON enhanced 3270 user interface and look for the messages that indicate a successful startup. The following message accompanied by no error messages indicates the completion of the OMEGAMON enhanced 3270 user interface startup process: `KOBCM0058I: Registry Refresh statistics:`
3. Log on to the OMEGAMON enhanced 3270 user interface and check that the initial workspace panels contain data sourced from the monitoring agents within this runtime environment. |
| Tivoli Enterprise Portal                | 1. On a distributed system, start the Tivoli Enterprise Portal Server and client.
2. Log on to the Tivoli Enterprise Portal client. When the Tivoli Enterprise Portal starts, check for the managed system name that you have configured (listed under the z/OS Systems entry in the Physical Navigator tree). |

**Note:** If you encounter problems, see the IBM Tivoli Monitoring: Troubleshooting Guide and the troubleshooting guide for each of your monitoring agents. Also see “Support information” on page 275.

Enable security

After you have established that your new runtime environment is configured correctly, you can safely enable the required level of security on each of the components within your runtime environment.

**About this task**

If you are using the OMEGAMON enhanced 3270 user interface to view data, you can enable security by following the steps described in “Enable security” on page 140.

If you are using the Tivoli Enterprise Portal to view data, you can create user accounts that authorize users to view the monitored data and enable authentication of those accounts by enabling security through the hub monitoring server or through the portal server. For instructions on enabling authentication on a hub monitoring server on Windows, Linux and UNIX operating systems, see the IBM Tivoli Monitoring: Administrator’s Guide. For instructions on enabling authentication on a hub monitoring server on z/OS systems, see the IBM Tivoli Management Services on z/OS: Configuring the Tivoli Enterprise Monitoring Server on z/OS.

After you have validated the configuration and successfully logged on to the Tivoli Enterprise Portal, you can create user accounts that authorize users to view the monitored data and enable authentication of those accounts by enabling security through the hub monitoring server or through the portal server. For instructions on enabling authentication on a hub monitoring server on Windows and UNIX operating systems, see the IBM Tivoli Monitoring: Administrator’s Guide. For instructions on enabling authentication on a hub monitoring server on z/OS, see IBM Tivoli Management Services on z/OS: Configuring the Tivoli Enterprise Monitoring Server on z/OS.
If you intend to use the OMEGAMON (3270) or OMEGAMON II user interfaces for a monitoring agent, enable security for it now. For instructions on implementing security, see the product-specific planning and configuration guide for each agent.

Enabling SNMP V3 passwords for autonomous agents

If you intend to use autonomous agents on a z/OS system and the ICSF subsystem is available on the z/OS system, you can enable SNMP V3 passwords.

Procedure

1. From the distributed installation media for either IBM Tivoli Monitoring or IBM Tivoli Management Services on z/OS, install the Tivoli Enterprise Monitoring Agent Framework.

   When you select **Tivoli Enterprise Monitoring Agent Framework** during installation, the following components are installed:
   - Tivoli Enterprise Monitoring Agent Framework
   - IBM GSKit Security Interface
   - Tivoli Enterprise Services User Interface

   This software includes the **itmpwdsnmp** tool, which converts the passwords in SNMP trap configuration files from plain text into GSKit-encrypted passwords. For installation instructions, see *IBM Tivoli Monitoring: Installation and Setup Guide*.

2. Run the **itmpwdsnmp** tool on the distributed system to encrypt the passwords in an SNMP trap configuration file. For instructions, see the “SNMP PassKey encryption: itmpwdsnmp” topic in the *IBM Tivoli Monitoring: Administrator’s Guide*.

   If you must encrypt only a few passwords, you can run the **itmpwdsnmp** tool in interactive mode.

3. Upload the SNMP trap configuration file in text mode to the **rhilev.rte.RKANDATV** data set for the runtime environment in which the monitoring agent is configured.

4. Use one of the following methods to create a KAES256 member in the **rhilev.rte.RKANPARU** data set for the runtime environment in which the monitoring agent is configured.
   - Copy the KAES256 member from the **rhilev.rte.RKANPARU** data set for the runtime environment in which the monitoring server is configured to the **rhilev.rte.RKANPARU** data set for the runtime environment in which the monitoring agent is configured. For instructions on creating the KAES256 member, see *IBM Tivoli Management Services on z/OS: Configuring the Tivoli Enterprise Monitoring Server on z/OS*.
   - In binary mode, copy the KAES256.ser file from the keyfiles directory of the distributed system where you ran the itmpwdsnmp tool to the KAES256 member of the **rhilev.rte.RKANPARU** data set for the runtime environment in which the monitoring agent is configured. The KAES256.ser file contains 48 bytes on distributed systems and is padded with blanks in the KAES256 member of the **rhilev.rte.RKANPARU** data set.

5. Concatenate the ICSF modules to the existing startup PROC **RKANMODL DDNAME** of the monitoring agent. To do so, edit the monitoring agent startup PROC and add ICSF support to the **RKANMODL DDNAME** . In the following example, the **CSF.SCSFMO0** data set contains the ICSF decryption modules:

   ```
   //RKANMODL DD DISP=SHR,DSN=rhilev.rte.RKANMODL
   // DD DISP=SHR,DSN=TDOMPT.LWMLVL..MODL
   // DD DISP=SHR,DSN=TDOMPT.CMSLVL..MODL
   // DD DISP=SHR,DSN=CSF.SCSFMO0
   ```

6. Restart the monitoring agent and verify that the passwords are decrypted. Check RKLVLOG for error messages indicating failure of GSKit password decryption or failure to create SNMP V3 trap destinations.
Complete configuration of the OMEGAMON enhanced 3270 user interface

To complete configuration of the enhanced 3270 user interface you must create a site-specific log-on profile and finish setting up security. If one has not already been created, you need to create a UNIX user ID for the system ID used by the enhanced 3270 user interface address space.

Create a CUASITE profile

The CUASITE profile controls site-specific features of the interface, such as actions assigned to PFKeys and the initial workspace displayed. Most importantly, the profile identifies the hub monitoring server from which the interface receives information. You must create a CUASITE profile in order to see data in the interface. At a minimum, you must specify the hub name, address, and port.

About this task

IBM provides a KOBCUA profile that contains default values. To create a site-specific profile, you copy the KOBCUA profile from your target &thilev.TKOBDATF data set to your &rhilev.&rte.UKOBDATF data set, rename it to CUASITE, and modify it to reflect your site requirements.

To establish profile settings for a user session, the interface looks in the UKOBDATF data set for a profile member named &user_id (a profile customized for a specific user ID), then for one named CUASITE (a site-customized profile), and then for one named KOBCUA. Renaming the KOBCUA profile to CUASITE is not required. However, it is a good practice to keep the profiles distinct.

Procedure

1. Copy the &thilev.TKOBDATF(KOBCUA) member to &rhilev.&rte.UKOBDATF and rename it to CUASITE.

2. Edit CUASITE and locate the hub settings in the CUADATA section. By default, the settings are provided as commented statements:
   /* HUBNAME=HUBNAME
   /* HUBIPADDRESS=::ffff:HUBADDRESS
   /* HUBPORTNUMBER=HUBPORTNUMBER

3. Uncomment the parameters and move them to column 1.

4. Specify the values for the parameters. For example:
   HUBNAME=HUB1:CMS
   HUBIPADDRESS=::ffff:9.44.44.22
   HUBPORTNUMBER=55555

   HUBNAME
   The configured name of the hub monitoring server. The monitoring server might be configured to run on z/OS or a distributed system such as Linux. This name is the value of the CMS_NODEID parameter in the hub monitoring server's RKANPARU(KDSENV) data set. The hub name is case-sensitive: the case of the HUBNAME value must match the case of the configured value.

   HUBIPADDRESS
   The TCP/IP address of the host system where the hub monitoring server runs. The setting must be an TCP/IP address as shown in the preceding example. Do not specify a TCP/IP host name.

   HUBPORTNUMBER
   The TCP/IP port number of the configured hub monitoring server (by default, 1918).

   For information about the other keyword=value pairs in the interface profile, see the IBM Tivoli OMEGAMON XE and Tivoli Management Services on z/OS: OMEGAMON Enhanced 3270 User Interface Guide.
Enable security

Authentication and authorization for users of the enhanced 3270 user interface is provided using the system authorization facility (SAF) interface. Security is enabled by specifying the name of a general resource class in the LPAR configuration file for the runtime environment in which the enhanced 3270 user interface is configured. A security administrator must define the system authorization facility (SAF) general resource class if it does not already exist and define profiles to control access to the interface, the data queries issued by the interface, and the actions performed by the interface. Users or user groups must be given access to the profiles.

About this task

Security for the enhanced 3270 user interface is configured by specifying the name of a SAF general resource class for the RTE_SECURITY_CLASS parameter. (In the Configuration Tool, the class name is specified in the Global SAF class name field. In a PARMGEN configuration, the RTE_SECURITY_CLASS is set in the LPAR configuration profile.)

If the name of the global security class was specified during configuration of the runtime environment, no further configuration of the environment is required. If no security class was specified at the time the runtime environment was configured, modify the environment using one of the following methods:

**PARMGEN**

Edit the LPAR configuration profile to specify the resource class name for RTE_SECURITY_CLASS parameter, resubmit the $PARSE (or $PARSESv) job to recreate the profile, then submit the following jobs to update the runtime environment:

- KCIJcLOD
- KCIJcCPR (backs up the RKAN* user libraries)
- KCIJcW2R (copies WKAN* to RKAN)

where c=V if system variables are enabled, c=P if system variables are not enabled. (See scenario RTE03 in the IBM Tivoli OMEGAMON XE and Tivoli Management Services on z/OS: PARMGEN Reference for more information.)

**Configuration Tool**

From the Runtime Environments (RTEs) panel, select U to update the environment. On the Update Runtime Environment (1 of 3) panel, provide the resource class name for the Global SAF class name. Press Enter until you return to the Runtime Environments (RTEs) panel, then select L to load the updated libraries.

If more granular security definitions are required, you can override the global SAF class for logon, queries, or Take Action commands. To override the RTE_SECURITY_CLASS, add the following parameters to the rte_plib_hilev.rte.WCONFIG(KOB$PENV) member (PARMGEN) or use the Nonstandard Parameter Editor (Configuration Tool).

**KOB_SAF_LOGON_CLASS_NAME**

Specifies a specific security class name that is to be employed for interface log-on authentication. This parameter defaults to the RTE_SECURITY_CLASS parameter value. This parameter should only be specified if the RTE_SECURITY_CLASS is not being specified or a unique security class name is required for log-on authorization.

**KOB_SAF_QUERY_CLASS_NAME**

Specifies a specific security class name that is to be employed for authorization of an interface query (data retrieval). This parameter defaults to the RTE_SECURITY_CLASS parameter value. This parameter should only be specified if the RTE_SECURITY_CLASS is not being specified or a unique security class name is required for data retrieval authorization.

**KOB_SAF_ACTION_CLASS_NAME**

Specifies a specific security class name that is to be employed for Take Action authorization. This
parameter defaults to the RTE_SECURITY_CLASS parameter value. This parameter should only be specified if a unique security class name is required for take action authorization.

**KOB_SAF_LOGON_RESOURCE_PREFIX**

Authorization to log on to the enhanced 3270 user interface is verified by checking for access to an SAF resource named in the following pattern:

KOB.LOGON.<plexname>.<smfid>.<stcname>

where KOB.LOGON is the logon resource prefix. This prefix can be changed by setting the parameter to another value.

The enhanced 3270 user interface provides a pseudo security class named OMEGDEMO. This class name is used to implement Demo mode. In Demo mode, no authorization checks are performed. This mode should be used only at the instruction of IBM Software Support. To activate Demo mode, see "Using Demo mode" on page 144.

**What to do next**

To complete the security set up for the OMEGAMON enhanced 3270 user interface, the following tasks must be completed by a security administrator:

1. Define an SAF general resource class
2. Define logon profiles to control access to the OMEGAMON enhanced 3270 user interface
3. Define Take Action profiles to control access to OMEGAMON enhanced 3270 user interface data actions
4. Define Query profiles to control access to OMEGAMON enhanced 3270 user interface data sources
5. Permit access to the profiles by appropriate personnel

A CUASITE logon profile must be created so that the interface knows the address and port of the hub monitoring server from which it is to collect data. If no UNIX ID has been created for the address space, one must be created.

**Define an SAF general resource class**

The resource profiles that control access to the OMEGAMON enhanced 3270 user interface are defined within an SAF general resource class. If you already have a general resource class that you want to use that has the appropriate CDT characteristics, use that class. If you do not have an appropriate class, or you want to use a different class, one must be defined.

**Before you begin**

The SAF general resource class must have the following CDT characteristics (DCT entry values):

- CASE(UPPER)
- FIRST(ALPHA,NATIONAL)
- OTHER(ALPHA,SPECIAL,NUMERIC,NATIONAL)
- MAXLENGTH(246)
- MAXLENX(246)
- KEYQUALIFIERS(0)
- PROFILESALLOWED(YES)
- GENERIC(ALLOWED)
- RAclist(REQUIRED)
About this task

You can define an SAF general resource class dynamically or by updating the CDT. If you are using RACF, special authority is required. The following procedure documents the steps for using RACF. Consult the product documentation for other security applications.

Procedure

Take one of the following approaches:

- To define an SAF general resource class using the ICHERCDE macro (for RACF), code:
  ```
  ICHERCDE CLASS=classname, CASE=UPPER, DFTRETC=4, DFTUACC=NONE,
  FIRST=ANY, GENLIST=ALLOWED, GENERIC=ALLOWED, MAXLENX=247,
  OTHER=ANY, POSIT=n, PROFDEF=YES, RACLIST=ALLOWED
  ```
  Both the classname and POSIT number should be unique. POSIT values 0 - 18, 57 - 127, and 528 – 1023 are reserved for IBM use and should not be used for your dynamic class entries. To list the POSIT numbers currently in use, use the following command:
  ```
  RLIST CDT * NORACF CDTINFO
  ```

- To define a RACF class dynamically, use the following commands:
  ```
  SETROPTS CLASSACT(CDT) RACLIST(CDT)
  RDEFINE CDT classname UACC(NONE) CDTINFO( +
  CASE(UPPER) FIRST(ALPHA) OTHER(ALPHA,SPECIAL,NUMERIC) +
  MAXLENGTH(246) MAXLENX(246) KEYQUALIFIERS(0) +
  PROFILESALLOWED(YES) POSIT(nnn) GENERIC(ALLOWED) +
  RACLIST(REQUIRED) )
  SETROPTS RACLIST(CDT) REFRESH
  SETROPTS CLASSACT(classname) RACLIST(classname) GENERIC(classname) REFRESH
  ```

What to do next

Now you can define the profiles that control access to the interface itself, to data actions, and to data queries.

Define LOGON profiles to control access to the interface

Authorization to log on to the enhanced 3270 user interface is controlled by logon profiles. These profiles must be created by a security administrator. If no SAF profile exists to protect an enhanced 3270 user interface instance, unrestricted logging on to that instance is allowed.

About this task

The enhanced 3270 user interface verifies a user's authority to log on by checking for access to an SAF resource named in the following pattern:

```
KOB.LOGON.<plexname>.<smfid>.<stcname>
```

where

- **KOB.LOGON**
  - Is the logon resource prefix.
- **plexname**
  - Is the name of a z/OS Sysplex.
- **smfid**
  - Is the SMF ID that represents the Sysplex member system.
- **stcname**
  - Is the job name of the OMEGAMON enhanced 3270 user interface started task.
For example, suppose you wanted to define a profile to control access to an enhanced 3270 user interface started task named KOBE3270 running on Sysplex IBMTEST on Sysplex member TST, in an SAF class named $KOBSEC. In this case, define a profile named KOB.LOGON.IBMTEST.TSTA by using the following commands:

```
RDEFINE $KOBSEC KOB.LOGON.IBMTEST.TSTA.KOBE3270 UACC(NONE)
SETROPTS RACLIST($KOBSEC) REFRESH
```

More generally, you can define a profile to control all logons to enhanced user interfaces by using the following commands:

```
RDEFINE $KOBSEC KOB.LOGON.** UACC(NONE)
SETROPTS RACLIST($KOBSEC) REFRESH
```

The logon prefix can be changed by adding the KOB_SAF_LOGON_RESOURCE_PREFIX parameter to the RANPARU(KOBENV) member of the runtime environment in which the e3270ui is configured. For example:

```
KOB_SAF_LOGON_RESOURCE_PREFIX="E32701."
```

changes the resource name pattern for resource profiles used to control logon to the OMEGAMON enhanced 3270 user interface to

```
E32701.<plexname>.<smfid>.<stcname>
```

**Tip:** If you have a PARMGEN implementation, use the WCONFIG(KOBSPENV) imbed file to add the parameter to the KOBENV file. The contents of the KOBSPENV file are dynamically embedded in the KOBENV file. This prevents the parameter from being overwritten when updates or maintenance is applied. If you used a Configuration Tool implementation, add the parameter using the Nonstandard Parameters feature in the Configuration Tool.

**Define Take Action profiles to control access to data actions**

Authorization to transmit Take Action requests from the OMEGAMON enhanced 3270 user interface to a product agent instance is controlled by a Take Action profile named for the specific Take Action command.

**About this task**

The authority to transmit Take Action commands is verified by checking for access to an SAF resource named in the following pattern:

```
<Kpp>.<msn>.TAKEACTION
```

- **Kpp** Is the product code of the agent instance. (See [Appendix F, “Product codes,” on page 265](#))
- **msn** Is a managed system name. A managed system name typically identifies a unique Tivoli Enterprise Monitoring Server agent instance. Note that the form of managed system names differs from product to product. Check the agent-specific Configuration and Planning Guide for information about the form used for managed system names.

```
TAKEACTION
```

Is a literal.

You must create an SAF profile to match the resource. If there is no matching profile for a particular Take Action command, *any* request to transmit an action to the managed system name is denied. For example, suppose that you wanted to control the ability to issue a Take Action command to an OMEGAMON XE on z/OS agent running on Sysplex IBMTEST on Sysplex member TST, in an SAF class name $KOBSEC. In this case, you would define a profile named KM5.IBMTEST:TSTA:MVSSYS.TAKEACTION by entering these commands:

```
RDEFINE $KOBSEC KM5.IBMTEST:TSTA:MVSSYS.TAKEACTION UACC(NONE)
SETROPTS RACLIST($KOBSEC) REFRESH
```
More generally, you could define a profile to control all Take Action commands for OMEGAMON XE on z/OS:

```sql
RDEFINE $KOBSEC KM5.**.TAKEACTION UACC(NONE)
SETROPTS RACLIST($KOBSEC) REFRESH
```

**Define Query profiles to control access to data sources**

**About this task**

The authority to issue query requests from the OMEGAMON enhanced 3270 user interface to a product agent instance is verified by checking for access to an SAF resource named in this pattern:

`Kpp.msn.tablename`

where

- **Kpp** is the product code of the agent instance. For example, for OMEGAMON XE for CICS, the product code is KCP. For OMEGAMON XE on z/OS, the product code is KM5. See Appendix F, “Product codes,” on page 265 for other products.

- **msn** is a managed system name. A managed system name typically identifies a unique Tivoli Enterprise Monitoring Server agent instance. Note that the form of managed system names differs from product to product. Check the agent-specific Configuration and Planning Guide for information about the form used for managed system names.

- **tablename** is the name of the data source (attribute group, or table) defined within the product agent. Note that if a matching SAF profile does not exist to protect a given query, that query is allowed. For example, suppose you wanted to control the ability to issue a query to an OMEGAMON XE on z/OS agent running on Sysplex IBMTEST on Sysplex member TSTA, for table KM5xxxxx. Assuming that the SAF class name is $KOBSEC, you would define a profile named KM5.IBMTEST:TSTA:MVSSYS.KM5xxxxxx by entering these commands:

```sql
RDEFINE $KOBSEC KM5.IBMTEST:TSTA:MVSSYS.KM5xxxxxx UACC(NONE)
SETROPTS RACLIST($KOBSEC) REFRESH
```

More generally, you could define a profile to control all data queries for a specific product:

```sql
RDEFINE $KOBSEC KM5.** UACC(NONE)
SETROPTS RACLIST($KOBSEC) REFRESH
```

**Permit access to profiles**

To access the profiles defined to control access, a user or group must be permitted to access the profile instances. Your security administrator will know best how to grant permissions to access these resources.

**About this task**

As an example, assuming an SAF class name of $KOBSEC, you can allow access to a resource by entering this command:

```sql
PERMIT KOB.LOGON.** ID(userid) ACCESS(READ) CLASS($KOBSEC)
PERMIT Kpp.**.TAKEACTION ID(userid) ACCESS(UPDATE) CLASS($KOBSEC)
PERMIT Kpp.** ID(userid) ACCESS(READ) CLASS($KOBSEC)
```

Queries require permission for READ access. Take Actions require permission for UPDATE access. Note that if no matching SAF profile exists to protect a Take Action, that Take Action command is denied.

**Using Demo mode**

The OMEGAMON enhanced 3270 user interface provides a pseudo security class named OMEGDEMO. This class name is used to implement demo mode. In demo mode, no authorization checks are performed. Use this mode only if instructed by IBM Software Support.
Procedure

To activate demo mode, take one of the following steps:

- If you are using the PARMGEN method, set the RTE_SECURITY_CLASS parameter in the configuration profile for the runtime environment in which the enhanced 3270 user interface is configured to OMEGDEMO.
- If you are using the Configuration Tool interactive method, specify OMEGDEMO in the Global SAF class name field of the CREATE/UPDATE RUNTIME ENVIRONMENT panel.

If Demo mode is in effect at startup, the following messages are written to the DDNAME SYSPRINT log written to SYSOUT:
KOBCS0001A PLEASE REVIEW THE SECURITY SET UP INFORMATION
KOBCS0001A SAF SECURITY HAS NOT BEEN ENABLED/CONFIGURED
KOBCS0001A THIS PRODUCT IS RUNNING UNSECURED!!

Create a UNIX ID for the address space

The enhanced 3270 user interface address space uses the Unix System Services API for TCP/IP communications, so the user ID associated with the address space must be defined to your security product as a z/OS UNIX user. If this user ID is not already defined, you should define it now.

About this task

The user ID for the address space must be defined using the security authorization facility (RACF, for example) to contain what is known as an OMVS Segment. As part of defining the OMVS Segment for the user ID, you must also supply a UNIX UID. The UID you supply does not need to have superuser authority (that is, UID=0 is not required). The following is an example of a RACF ADDUSER command that defines an OMVS Segment and a UID for a new RACF user ID:
ADDUSER userid ... OMVS(UID(nnn) ...)

(Optional) Enable maintenance of the historical data store

If you intend to enable historical data collection and have allocated the historical data set and configured maintenance for data store, you must perform two additional tasks to enable the maintenance.

About this task

The KPDPROC1 maintenance procedure that is used to maintain the physical files must have the required authority to read, write, and update the files.

The KPDPROCC REXX procedure runs in a TSO environment and must be enabled to run as an authorized program under TSO. To authorize the KPDPROCC procedure, you must authorize the KPDDSCO module.

If you are upgrading an existing monitoring server, you must also refresh the KPDPROC1 maintenance procedure in your system procedure library. See the IBM Tivoli OMEGAMON XE and Tivoli Management Services on z/OS: Upgrade Guide.

Procedure

1. Ensure that KPDPROC1 procedure has the necessary authority to read, write, and update the persistent data store files.
2. Authorize the KPDDSCO module by adding KPDDSCO to the system PARMGEN(IKJTSOnn) under the AUTHPGM section and refresh the IKJTSOnn member by issuing the set command (SET IKJTSO=nn). You might request that authorized system programmers perform this step so it can be scheduled with the LPAR change control processes.
3. Verify that the configuration and authorization has been successful:
   a. Bring up the started task (for monitoring server or monitoring agent) that will collect historical data into the product's persistent data store libraries. In the RKPDLOG DDNAME started task, find any persistent data store libraries in a non-Offline status (for example, Partial or Full status).
   b. From a z/OS operator console, issue the following z/OS MODIFY command:
      
      `/F stcname,KPDCMD RECOVER FILE=DSN:pds_data set`

      where stcname is the name of the started task performing the persistent data store collection, and pds_data set is the persistent data store data set.
      
      For example, issue the following MODIFY command for the monitoring server:
      
      `/F CIDSSST,KPDCMD RECOVER FILE=+
       DSN:rhilev.rte.RGENHIS1`

   c. Wait 5 minutes.
   d. In the RKPDLOG DDNAME started task, find the following Command: and KPDDSTR: references as shown in the following monitoring server RKPDLOG DDNAME example:

   Command: RESUME FILE=DSN:rhilev.rte.RGENHIS1
   KPDDSTR: CONNECT processing started for DataStore file
   DSN:rhilev.rte.RGENHIS1
   KPDDSTR: CONNECT processing ended for DataStore file
   DSN:rhilev.rte.RGENHIS1

   e. If these references are not found, view the KPDPROC1 started task in SDSF and look for any obvious errors.

What to do next

Even though the historical data store is allocated and configured, no historical data is collected until collection is enabled using the Tivoli Enterprise Portal.

(Optional) Enable historical data collection

No historical data is collected unless the collection of historical data is enabled in the Tivoli Enterprise Portal.

About this task

The Tivoli Enterprise Portal History Collection dialog box lists the attribute groups, or tables, for each monitoring agent that are enabled for historical collection and reporting. To enable historical data collection, you must select and configure each attribute table for which you want to collect data, and then start collection for those tables. If you want to warehouse the data for long-term historical reporting, you must set the Warehouse Interval to the interval at which data is warehoused.

For detailed instructions on configuring history data collection, see the IBM Tivoli Monitoring: Administrator's Guide and the Tivoli Enterprise Portal online help.

Run the ITMSUPER Tools (optional)

Use the ITMSUPER Tools to learn about the health of your managed systems, situations, and environment configuration. The ITMSUPER Tools are included in the IBM Support Assistant (ISA), a free local software serviceability workbench that helps you resolve questions and problems with IBM software products.

About this task

To install the ISA software, go to [http://www.ibm.com/software/support/isa](http://www.ibm.com/software/support/isa)
Part 3. Replicating configured runtime environments

You can replicate configured runtime environments by cloning them and transporting them, if necessary, to other LPARs. The topics in this section help you replicate configured runtime environments and complete an enterprise-wide deployment.
Chapter 7. Using the PARMGEN method to replicate a configured runtime environment

Cloning runtime environments allows you to replicate environments with considerably less customization.

There are 8 main steps involved in cloning a runtime environment using the PARMGEN method:
1. Set up PARMGEN work environment for the runtime environment (KCIJPCFG).
2. Clone customized WCONFIG members to the WCONFIG library of the new runtime environment (KCIJPCCF).
3. Update interim libraries and create PARMGEN configuration profiles (KCIJPUP1).
4. Merge the configuration profile parameter values from the model RTE into the new runtime environment (KCIJPM2).
5. Customize the PARMGEN configuration profiles.
6. Validate PARMGEN profile parameter values (KCIJPVAL).
7. Create the runtime environment members and jobs ($PARSE/$PARSES).
8. Submit batch jobs to complete PARMGEN setup.

Where these steps are performed depends upon which transport scenario you are using to replicate the environment to another LPAR.

PARMGEN transport scenarios

The way in which you decide to share or transport work and runtime libraries determines where you run the configuration jobs required by the replication scenario you are using to create new runtime environments.

The following are all possible methods of sharing or transporting the runtime libraries between one runtime environment and another.

- Scenario 1: Defining a runtime environment on a z/OS image using shared DASD.
  This scenario employs a central site that defines and maintains the runtime environment configuration for both local and remote z/OS images. Each unique runtime environment that is defined represents a z/OS image. If a site has 10 shared-DASD z/OS systems (LPARs), the central site defines 10 unique runtime environments (either sharing-with-SMP/E targets, Full, or sharing-with-base) as part of the normal PARMGEN configuration process. These runtime environments are configured based on each remote site’s respective configuration requirements (data set naming conventions, VTAM nodes, and the like). The central site must allow for the necessary DASD to allocate the libraries needed for each runtime environment.
  Using this method, all jobs run on the same system and no libraries are copied.

- Scenario 2: Transporting a runtime environment from one z/OS image to another.
  This scenario is similar to method 1, except the 10 LPARs are on unshared DASD. During rollout of the initial runtime environment or during maintenance, you transport the product execution libraries (RK*) to each LPAR using unshared DASD.
  Using this method, all jobs are run on the same system. Only the RK* libraries are copied to the target system. If system variables are enabled, the GBL_USER_JCL library must also be copied.

- Scenario 3: Transporting PARMGEN WKANSAMU batch jobs from one z/OS image to another equipped with the SMP/E target libraries.
  In this method, a central site maintains the OMEGAMON and Tivoli Management Services CSI and the SMP/E maintenance of the target data sets. Initial build distribution of the PARMGEN-supported
OMEGAMON and Tivoli Management Services products to the remote z/OS images is done by copying the SMP/E target libraries and the PARMGEN batch jobs in the %RTE_HILEV%.%RTE_NAME%.WK* and WCONFIG libraries to the remote sites. If system variables are enabled, the GBL_USER_JCL library must also be copied.

All the required steps are performed at the central site, up to $PARSE or $PARSESV job. Then the WKANSAMU jobs and the SMP/E target libraries are transported to the remote LPARs. On the remote site, only the WKANSAMU jobs (such as the KCIJPLOD allocation job and the KCIJPHOD load job) are run to complete setup of the runtime environment (option 11 on the PARMGEN Workflow main panel). During maintenance, the latest fixes are applied to the central site, which then redistributes the updated target libraries to each remote site. The remote sites simply run the KCIJPHOD job to copy the new fixes from the updated or upgraded SMP/E target data sets to their product execution libraries (RK*).

- Scenario 4: Transporting PARMGEN runtime environment work libraries from one z/OS image to another that is equipped with the SMP/E target libraries.

  This scenario is similar to scenario 3, except in this scenario the PARMGEN setup is done at the remote LPAR. That is, the $PARSE or $PARSESV job is run at the remote site instead of the central site. That means that the runtime environment members and jobs are created on the remote LPAR, and you have the option of cloning from an existing set of PARMGEN work environments with already customized configuration profiles in WCONFIG, then transporting the PARMGEN work libraries (%RTE_PLIB_HILEV%.%RTE_NAME%.IK*, %RTE_PLIB_HILEV%.%RTE_NAME%.WCONFIG, %RTE_PLIB_HILEV%.%RTE_NAME%.WK*, GBL_USER_JCL) to the remote LPAR, and completing the remaining PARMGEN setup there.

Scenarios 1 and 2 support the use of system variables; methods 3 and 4 do not.

In scenarios 1 and 2, if system variables have been enabled for the runtime environment, there are several ways you can implement these symbolics as profile parameter values, depending on which type of system variables are being used (static symbols, KCIPARSE-extracted symbols, or user-defined symbols).

Symbolics for PARMGEN parameters like RTE_NAME and RTE_HILEV are used directly in the KCIJP* and KCIJV* batch jobs created by $PARSESV in the WKANSAMU library. If these symbolics are static system symbols (&SYSNAME., &SYSCLONE., for example) or KCIPARSE-extracted symbols (&SYSSIPHOSTNAME., &SYSVTAMNETID.) that resolve to different values on each LPAR, rather than user-defined symbols whose resolution values are controlled by the %GBL_USER_JCL%(%RTE_NAME%) LPAR system variables resolution member, the PARMGEN process assumes you plan to submit the WKANSAMU(KCIJV*) jobs on the system where the symbolics resolve. If you want to complete the setup in the local system (the central site) rather than the target system (the remote LPAR) where you intend to deploy the product execution RK* runtime libraries, enable the RTE_X_SYSV_OVERRIDE_SYMBOLS parameter in the configuration profile:
EDIT IBM.ITM62351.LPAR2.WCONFIG(LPAR2)
Command ===>
****** ************************************************ Top of Data ************************************************

000086 ** (Optional) z/OS System Variable setting:
000087 ** ---------------------------------------------------------------
000088 ** For TYPE:CE (CHAR extracted) and TYPE:IE (INTEGER extracted)
000089 ** KCIPARSE-extracted symbolics for System Variables use in the
000090 ** PARMGEN CONFIG parameter values. In the GLOBAL VARIABLE table
000091 ** below, "VARNAME" value may be used using the syntax in the
000092 ** "PARMGEN CONFIG Value" column on line#29 for example:
000093 ** Syntax: "&" + "VARNAME" + "." so if VARNAME = SYSIPHOSTNAME, then
000094 ** specify &SYSIPHOSTNAME. as the PARMGEN CONFIG Value as
000095 ** shown on line#29 below.
000096 ** 000021 RTE$ BEGIN *-------------- PARMGEN CONFIGURATION
000097 ** 000022 ** ============================ =====================
000098 ** 000023 ** PARMGEN CONFIG Parameter PARMGEN CONFIG Value
000099 ** 000024 ** ============================ =====================
000100 ** 000029 RTE_TCP_HOST &SYSIPHOSTNAME.
000101 ** 000029
000102 ** _______________GLOBAL VARIABLE TABLE SUMMARY_________________
000103 ** VARNAME TYPE SAMPLE VALUE
000104 ** ======= ==== ============
000105 ** SYSIP CE &SYSIP.
000106 ** SYSIPADDRESS CE &SYSIPADDRESS.
000107 ** SYSIPHOSTNAME CE &SYSIPHOSTNAME.
000108 ** SYSJOBNAME CE &SYSJOBNAME.
000109 ** SYSMMFID CE &SYSMMFID.
000110 ** SYSSMS CE &SYSSMS.
000111 ** SYSTCPPNAME CE &SYSTCPPNAME.
000112 ** SYSVTAMNETID CE &SYSVTAMNETID.
000113 ** SYSVTAMSSCP CE &SYSVTAMSSCP.
000114 ** SYSDATE CE &SYSDATE.
000115 ** SYSTIME CE &SYSTIME.
000116 ** SYSDAY CE &SYSDAY.
000117 ** SYSDAY IE &SYSDAY.
000118 ** SYSTIME IE &SYSTIME.
000119 ** ------------------------------------------------------------------
000120 ** Specify "Y" if you are using symbolics as parameter values.
000121 ** RTE_SYSV_SYSVAR_FLAG N
000122 RTE_SYSV_SYSVAR_FLAG
000123 ** If RTE_SYSV_SYSVAR_FLAG = Y:
000124 ** If you are using symbolics for PARMGEN parameters like RTE_NAME
000125 ** and RTE_HILEV, these symbols are used directly in PARMGEN batch
000126 ** jobs created by SPARSERV in the WKANSAMU libraries. If these are
000127 ** static system symbols (&SYSNAME., &SYSCONFIG., etc.) or KCIPARSE
000128 ** extracted symbols (&SYSIPHOSTNAME., &SYSVTAMNETID.) that resolve to
000129 ** different values on each LPAR, rather than user-defined symbols you
000130 ** can control via TDINTN.PARMGEN.JCL(LPAR2) LPAR System Variables
000131 ** resolution member, the PARMGEN process assumes you plan to submit
000132 ** the WKANSAMU(KCIJV*) jobs in the system where the symbolics
000133 ** resolve. You have the option to override the system symbols
000134 ** locally so you can complete the RTE set-up in the local system
000135 ** rather than the target system you intend to deploy the runtime
000136 ** libraries.
000137 ** Specify "Y" if you want to override the symbol values by defining
000138 ** the symbols and their resolved values in the
000139 ** TDINTN.PARMGEN.JCL(LPAR2) PARMGEN user JCL
000140 ** (similar process when defining user-defined symbols):
000141 ** RTE_X_SYSV_OVERRIDE_SYMBOLS N

Then ensure that the system variables member in the user JCL on the target LPAR properly defines the
static system symbols with your override values. Lines 12-14 in the following screen show an example
of KCIPARSE-extracted symbols and your override values for LPAR2 if you intend to run the LPAR2
KCIJV* jobs on LPAR1.
Required LPAR2 WKANSAMU jobs:

- KCJPALO
- KCJPUPV
- KCJVSUB
- KCJVALO
- KCJVLOD
- KCJVUPV

Optional WKANSAMU LPAR2 jobs (depending on the configuration):

- KCJVSEC
- KCJVUSP
- KCJVUSS
- KCJVLNK
- KCJVSYS
- KCJVVP
- KCJVCVPY
- KCJVW2R
- KCJVDEL
- KCJVMNT

For descriptions of these jobs, see “Step 11. Submit batch jobs to complete the PARMGEN setup” on page 98.

Clone an existing runtime environment

You can clone an existing PARMGEN runtime environment to create a new environment that runs on a different LPAR but uses the same values and has the same products configured. In this scenario, nothing changes, except LPAR-specific values.

Before you begin

This scenario uses the PARMGEN Workflow user interface to set up the work environment, customize profiles, and create and submit the jobs that create the runtime environment. For instructions on launching the interface, see “Start the Workflow user interface” on page 67.
About this task

In this scenario, you use the configuration values from one PARMGEN runtime environment to create another on a different LPAR. There are 8 main steps involved in the scenario:

1. Set up the PARMGEN work environment for the new runtime environment.
2. Clone customized WCONFIG library members from the model runtime environment to the WCONFIG library for the new runtime environment.
3. Update the interim libraries and create the PARMGEN configuration profiles.
4. Merge the configuration profile parameter values from the model runtime environment into the new one.
5. Customize the configuration profiles for the new environment.
6. Validate the profile parameter values.
7. Create the runtime members and jobs.
8. Submit the jobs to complete create the runtime environment.

If the LPAR on which the new runtime environment is being configured does not shared DASD with the LPAR on which the runtime environment will run, the PARMGEN libraries must be copied to the target LPAR. (See the “PARMGEN transport scenarios” topic in the IBM Tivoli OMEGAMON XE and Tivoli Management Services on z/OS: Common Planning and Configuration Guide for more information.) Where each of these steps is performed depends on the transport method being used to deploy the new environment.

Note: In this scenario, system variables are not enabled. If you intend to clone an environment which uses system variables, see “Clone an existing environment with system variables enabled to run on a different LPAR” on page 164.

Procedure

1. Set up the PARMGEN work environment for the converted runtime environment.
   a. On the main Workflow panel, specify the parameters to identify the new runtime environment you are configuring. Note that if you did not invoke the R (RESET) option, the values for GBL_USER_JCL, RTE_PLIB_HILEV, and RTE_NAME are carried forward from the last environment you created. If the values for the GBL_USER_JCL library and the high-level qualifier for the work libraries are correct, change the RTE_NAME to the name you want to use for the new environment.

   **GBL_USER_JCL**
   Specify the dataset name of the PARMGEN global user JCL library for the new runtime environment. If the dataset does not exist, you are prompted to correct the name or allocate the data set.

   **RTE_PLIB_HILEV**
   Specify the non-VSAM high-level qualifier you want to use for the PARMGEN work libraries for the new runtime environment.

   **RTE_NAME**
   Unique name for the runtime environment you are configuring. The name you specify is appended to the RTE_PLIB_HILEV values to make each set of PARMGEN and runtime libraries unique.

   b. Select option 1 from the Workflow main menu. The Set up PARMGEN Work Environment for an RTE (1 of 1) (KCIP@PG1) is displayed.

   c. Specify the required parameter values.

   **Profile library and member name**
   Specify the profile library and LPAR configuration profile member for the environment whose values you want to clone.
Install Job Generator (JOBGEN) output library

Specify the Install Job Generator (JOBGEN) output library if you want PARMGEN to reuse parameter values from the JOBGEN output library repository. These are values such as the jobcard, CALLLIBS override data, and other CSI values such as:

If the specified dataset does not exist, you are prompted to correct the name.

Note: For more information about the JOBGEN output library, refer to the Installation Job Generator Utility section in the product's Program Directory.

Jobcard data

If a customized jobcard is already available, it is retrieved from the ISPF user profile pool. If a jobcard is not available and you did not specify a JOBGEN output library, the jobcard is harvested from the &gbl_target_hilev.TKANSAM SMP/E target library where the initial PARMGEN sample jobcard default is supplied. If you specified a JOBGEN output library, the jobcard information is harvested from there. You can modify the retrieved data as needed. The customized jobcard is saved in the ISPF user profile pool and persists across ISPF sessions.

d. Press Enter to proceed to the next panel. If the GBL_USER_JCL data set does not exist, you are presented with the following verification screen:

![Verification Screen](image)

e. Perform one of the following steps:
   - Press Enter to allocate a new data set.
   - Specify the correct data set name, then press Enter.

   The Set Up PARMGEN Work Environment for an RTE (2 OF 3) panel (KCIP@PG2) is displayed.

f. After the necessary parameters have been specified, press Enter to proceed.
g. Review the values on the panel and override them as necessary. These values are read from the profile library and runtime member name you specified on the preceding panel.

h. Press Enter to proceed to the next panel. The third setup panel (KCIP@PG3) is displayed. Depending on your screen resolution, you may have to scroll down (PF8) to see all the parameters. (Note the More: + indicator.)
Typically, you would make the following changes on this panel. However, this scenario merges the customized values from the model runtime environment. Those values will take precedence over any values specified here, so you will change the values of the corresponding parameters in the LPAR configuration profile for the new runtime environment.

1) Change the type of runtime environment to Sharing-with-SMPE by setting RTE_TYPE to SHARING and RTE_SHARE to SMP.
2) Change RTE_TEMS_NAME_NODEID to PLB2SP23:CMS.
3) (Optionally) Change RTE_X_SECURITY_EXIT_LIB to TSTEST.userid.PLB2SP23.RKANSAMU. If you are sharing the same security settings and permission as the model runtime environment, you can use the same library.
4) Convert the hub to a remote by setting KDS_TEMS_TYPE to REMOTE.

i. Press Enter to proceed to the next panel. The Display PARMGEN Environment Analysis panel is displayed.

This panel lists the products installed and available for configuration. The "Components installed and configured in the RTE profile &rte_name" section lists products that will be configured. This list was inherited from the WCONFIG member for the runtime environment that is being cloned.

j. Review the message traffic, then press Enter to proceed to the next panel. The Exclude Products from PARMGEN Customization panel (KCIP@PG5) is displayed.
Select (X) products to EXCLUDE from PARMGEN customization in RTE=&rte_name.

When finished, change 'N' to 'Y' to confirm selections. Confirm ==> Y (Y, N)

<table>
<thead>
<tr>
<th>Kpp</th>
<th>Product Name/Version (Kpp* components configured in RTE profile)</th>
</tr>
</thead>
<tbody>
<tr>
<td>X ALL Exclude all not configured in RTE profile</td>
<td></td>
</tr>
<tr>
<td>_ KC5* IBM Tivoli OMEGAMON XE for CICS on z/OS V510</td>
<td></td>
</tr>
<tr>
<td>_ KD0 IBM Tivoli Decision Support for z/OS V101</td>
<td></td>
</tr>
<tr>
<td>_ KDS IBM Tivoli Enterprise Monitoring Server V630</td>
<td></td>
</tr>
<tr>
<td>_ KD4 IBM Tivoli Composite Application Manager for SOA V711</td>
<td></td>
</tr>
<tr>
<td>_ KDG IBM Tivoli OMEGAMON XE for DB2 PE/PM V511</td>
<td></td>
</tr>
<tr>
<td>_ KDM* IBM Tivoli OMEGAMON XE for CICS TG on z/OS V510</td>
<td></td>
</tr>
<tr>
<td>_ KI5 IBM Tivoli OMEGAMON XE for IMS on z/OS V510</td>
<td></td>
</tr>
<tr>
<td>_ KM5* IBM Tivoli OMEGAMON XE for Messaging - WebSphere MQ Config V710</td>
<td></td>
</tr>
<tr>
<td>_ KMC IBM Tivoli OMEGAMON XE for Messaging - WebSphere MQ Monitor V710</td>
<td></td>
</tr>
<tr>
<td>_ KMD IBM Tivoli OMEGAMON XE for Messaging - WebSphere MQ Monitor V710</td>
<td></td>
</tr>
<tr>
<td>_ KMG IBM Tivoli OMEGAMON XE on z/OS V510</td>
<td></td>
</tr>
<tr>
<td>_ KN3 IBM Tivoli OMEGAMON XE for Mainframe Networks V510</td>
<td></td>
</tr>
<tr>
<td>_ KOB* IBM Tivoli OMEGAMON Enhanced 3270 User Interface V700</td>
<td></td>
</tr>
<tr>
<td>_ KQI IBM Tivoli OMEGAMON XE for Messaging - WebSphere Message Broker V710</td>
<td></td>
</tr>
<tr>
<td>_ KR6 IBM Tivoli Advanced Audit for DFSMSdss V240</td>
<td></td>
</tr>
<tr>
<td>_ KRH IBM Tivoli Advanced Reporting and Management for DFSMSdss V240</td>
<td></td>
</tr>
<tr>
<td>_ KRJ IBM Tivoli Advanced Allocation Management for z/OS V320</td>
<td></td>
</tr>
<tr>
<td>_ KRR IBM Tivoli Automated Tape Allocation Manager for z/OS V320</td>
<td></td>
</tr>
<tr>
<td>_ KRM IBM Tivoli Advanced Catalog Management for z/OS V240</td>
<td></td>
</tr>
<tr>
<td>_ KRV IBM Tivoli Advanced Backup and Recovery for z/OS V230</td>
<td></td>
</tr>
<tr>
<td>_ KRW IBM Tivoli Tape Optimizer for z/OS V220</td>
<td></td>
</tr>
<tr>
<td>_ KS3 IBM Tivoli OMEGAMON XE for Storage on z/OS V510</td>
<td></td>
</tr>
<tr>
<td>_ KWD IBM Tivoli OMEGAMON DE on z/OS - OMEGAVIEW and OMEGAVIEW II V510</td>
<td></td>
</tr>
<tr>
<td>_ KYN ITCAM for Application Diagnostics on z/OS V710</td>
<td></td>
</tr>
</tbody>
</table>

This panel lists the products installed in the runtime environment and therefore available for configuration. They are listed by product code (Kpp) and product name and version. The products marked with an asterisk (*) are components read from the model environment, but the version represents the latest version available. In this example, a monitoring server, OMEGAMON XE on z/OS, OMEGAMON XE for CICS on z/OS, OMEGAMON XE for CICS TG on z/OS, and the OMEGAMON enhanced 3270 user interface are installed.

k. To include only those products and versions that were already configured in the runtime environment you are cloning, place an X before ALL Exclude all not configured in RTE profile. If you want to install additional products, or eliminate already installed products, include or exclude them as appropriate instead. Type Y in the Confirm ==> field, then press Enter to confirm your exclusion choices. The KCIJPCFG job is displayed, tailored with the values you have configured.

l. Review the notes and values and submit the job, then return to the Workflow main panel.

2. (Optional) Clone customized members from the model runtime environment (&clone_from).
   a. Select option 3 from the Workflow main menu.
Enter PARMGEN parameter values appropriate for your environment:

- **GBL_USER_JCL**: TSTEST.SYSPLEX.PARMGEN.JCL
- **RTE_GLIB_HILEV**: TSTEST.&userid
- **RTE_NAME**: &rte_name (Type ? for a list of configured RTEs)

Enter n (1-11) to perform tasks. Status Date
Enter ns (1s-11s) to display detailed status.

1. KCIJPCFG Set up PARMGEN work environment for an RTE. Submitted 2012/08/17
2. $JOBINDEX Review PARMGEN job index. Viewed 2012/08/17
3. KCIJPCCF Clone customized WCONFIG members. (COND)
4. KCIJPUP1 Update interim libraries and create profiles.
5. KCIJPCMC1 Merge profile from backup PLB4SPI3 (COND)
6. KCIJPCMC2 Merge profile from model RTE. (COND)
7. KCIJPCCN Convert an ICAT RTE Batch member. (COND)
8. Arte_name Customize PARMGEN configuration profiles.
9. KCIJPVAL Validate PARMGEN profile parameter values.
10. $PARSE Create the RTE members and jobs.
11. SUBMIT Submit batch jobs to complete PARMGEN setup.
U Utility Access PARMGEN utilities. (Optional)
R New RTE Reset RTE, Status and Date fields. (Optional)

The KCIJPCCF job is displayed. This job clones customized members ($KPP$SC*, $KPP$SP*/$KPP$P*, $KPP$SS*), $JOBCARD, and the $GBL$USR global profile) from an existing WCONFIG library to a new one.

b. Review the job to determine if you need to modify the library concatenated in the OLDWCNFG DDNAME.

By default, this DDNAME already points to the WCONFIG library that you specified in the RTE profile library field of the first work environment setup panel. If necessary, modify the job to point to an existing WCONFIG library that contains the customized members you want cloned over to the WCONFIG library for the new runtime environment.

c. Submit the job and return to the main Workflow panel.

The specified members are copied to the WCONFIG library for the new runtime environment.

3. Update the interim libraries and create the configuration profiles:
   
a. Select option 4.
The KCIJPUP1 job is displayed.

b. Review the KCIJPUP1 job and submit it, then return to the Workflow main menu.

The work libraries are updated with the configuration values you specified and the configuration profiles for the runtime environment are created.

4. Merge the configuration profile parameter values from the model runtime environment into the new one.

a. From the Workflow main menu, select option 6 (KCIJPMC2 Merge profile from model RTE).
b. Review the job and make any required changes. By default, the OLDMEM(&value) parameter is already set to the runtime environment you specified in the first setup panel (in this example, TSTEST.&userid.&clone_from.WCONFIG(&clone_from)).

c. Review and submit the job and return to the Workflow main menu.

5. Customize the configuration profiles for the new environment.
   a. From the Workflow menu, select option 8.
The Customize PARMGEN Configuration Profile Members panel is displayed.

b. Select option 1 to customize the LPAR configuration for the new runtime environment. Change any instances of &<clone_from> inherited from the model environment to &<rte_name>, and change any LPAR-specific values, such as host name.

c. After you complete any necessary changes, press F3 to return to the Customize PARMGEN Configuration Profile Members panel.

d. If necessary, edit the $GBLSUSR and &<rte_name> system variables configuration files, then return to the main Workflow panel.

6. Validate the configuration profile values that you supplied:

a. On the Workflow main panel, select option 9.
The KCIJPVAL job is displayed.

b. Review the notes and submit the job. The parameter validation report is written to &hilev.&rte.WCONFIG($VALRPT). Enter 9s option view the report.

c. If the job gets a condition code greater than 4 (COND CODE 0004), review the $VALRPT. Return to the previous step and correct the parameter values, then resubmit the validation job.

7. Create the WK* runtime members and the WKANSAMU jobs:

a. Select option 10 from the Workflow main menu.

The $PARSE/$PARSESV: Create the RTE Members and Jobs panel is displayed.

b. Select option 1 to submit the generated $PARSE composite job.
The $PARSE member is presented.

c. Review the member for any jobs that need to be submitted manually, then submit the job. Return to the main Workflow panel.

8. Submit the batch jobs to complete the setup of the runtime environment:

a. Select option 11 from the Workflow main menu.

The Submit Batch Jobs to Complete PARMGEN Setup (KCIP@SUB) panel is displayed.

b. Select option 1 if you want to submit the KCIJPSUB master SUBMIT job, or select the other options to submit each job individually. If you select option 1, review the KCIJPSUB job first, and edit the job according to which of the conditional jobs need be submitted automatically and which need to be submitted manually (for example, for reasons of authorization).
c. Submit the job.

Results
You have created a copy of an existing runtime environment.

What to do next
If necessary, copy the PARMGEN libraries to the target LPAR and start the products.

Clone an existing environment with system variables enabled to run on a different LPAR

This scenario clones an existing runtime environment, which currently runs on LPAR SYSG, to create a new runtime environment, which will run on LPAR SP14. Use of system variables is enabled in the environment, so system variables overrides must be enabled to allow the new runtime environment to run on a different LPAR.

Before you begin
This scenario uses the PARMGEN Workflow user interface to set up the work environment, customize profiles, and create and submit the jobs that create the runtime environment. For instructions on launching the interface, see “Start the Workflow user interface” on page 67.

About this task
This task involves 8 steps:
1. Create the PARMGEN work environment.
2. Clone the customized WCONFIG members from the model runtime environment.
3. Update the interim libraries and create the configuration profiles for the new environment.
4. Merge the LPAR profile from the model environment with the LPAR profile for the new environment.
5. Customize the profiles for the new environment.
6. Validate the profile parameter values.
7. Create the runtime members and jobs.
8. Submit the jobs to create the runtime environment.

Where these steps are run depend upon the transport scenario you are following (see “PARMGEN transport scenarios” on page 149).

Complete these steps to create sample runtime environment PLB3SP14.

Procedure
1. Create the PARMGEN work environment by completing the following steps.
   a. On the Workflow main panel, specify the GBL_USER_JCL, RTE_PLIB_HILEV, and RTE_NAME parameters, as shown in the following example, then select option 1 to set up the PARMGEN work libraries for the new PLB3SP14 environment. Note that if you did not invoke the R (RESET) option, the values for GBL_USER_JCL, RTE_PLIB_HILEV, and RTE_NAME are carried forward from the previous environment; in this case, simply change the RTE_NAME to PLB3SP14.

Since the runtime environment you specified is a new one, you are asked if you want to reset the definition values before you proceed:

b. Press Enter. You are presented with a verification screen:
c. Press Enter. You are returned to the main Workflow panel.

d. Select 1 again to begin setting up the work environment. The first set up panel (KCIP@PG1) is displayed.

e. In the first field, specify the profile library and member for the environment from which the new environment will inherit the product set and reuse common values in subsequent panels. If appropriate, specify the Install Job Generator output library. In this example, the model environment is PLB3SP13.

KCIP@PG1 ---- SET UP PARMGEN WORK ENVIRONMENT FOR AN RTE (1 OF 3) -------------

Specify the RTE profile library and member name that fits your scenario:

- TSTEST.&userid.PLB3SP13.WCONFIG(PLB3SP13) (ex: &dset(&rte))
  - If creating a brand new RTE, leave this field blank. **or**
  - If creating another new RTE and you want to clone a PARMGEN-created RTE's configured product set, specify the WCONFIG profile library and RTE member name to clone (ex.: &hlq.&rte.WCONFIG(&clone_from)). **or**
  - If reconfiguring or upgrading this existing PLB3SP14 RTE, specify its values (TSTEST.&userid.PLB3SP14.WCONFIG(PLB3SP14)) **or**
  - If converting an ICAT-created RTE to PARMGEN mode, specify the ICAT RTE Batch member location and RTE member (ex.: &hlq.INSTJOBS(PLB3SYSG))

Specify the Install Job Generator (JOBGEN) output library if you want PARMGEN to reuse CALLLIBS parameters from the JOBGEN repository:

- (Type ? for last referenced JOBGEN library discovered, if any.)

Enter Jobcard data:

- //&userid.J JOB (ACCT), 'SYSMEMBER %username% ,CLASS=A,____________________
- // MSGCLASS=X,MSGLEVEL=(1,1), NOTIFY=&SYSUID,,REGION=0M_________________
- //** RTE_NAME=%RTE_NAME%________________________________________________
- //** SYSJOBNAME=%SYSJOBNAME% SYSMEMBER=%SYSMEMBER%______________________

f. Press Enter to proceed to the next panel. The second set up panel is displayed:

KCIP@PG2 ---- SET UP PARMGEN WORK ENVIRONMENT FOR AN RTE (2 OF 3) -------------

GBL_USER_JCL: TSTEST.SYSPLEX.PARMGEN.JCL
RTE_PLIB_HILEV: TSTEST.&userid
RTE_NAME: PLB3SP14

Enter parameter values appropriate for your environment:

GBL_INST_HILEV:
- HLQ of ICAT INSTLIB/INSTJOBS installation datasets

GBL_TARGET_HILEV:
- TSTEST.TM62351
- HLQ of SMP/E target (TK*) datasets

GBL_SYSDA_UNIT:
- SYSDA
- Non-VSAM disk UNIT (global work datasets)

Enter=Next F1=Help F3=End/Cancel

g. Review the values on the panel and override them as necessary.
h. Press Enter to display the next panel. The third set up panel is displayed:

```plaintext
KCIP@PG3 ---- SET UP PARMGEN WORK ENVIRONMENT FOR AN RTE (3 OF 3) ---------
Command ===> Scroll ===> PAGE

Enter parameter values appropriate for your RTE=PLB1SP22.
Note: See F1=Help for SMS-related and RTE HLQ-related considerations
when VOLUME, UNIT, STORCLAS, and MGMTCLAS parameters are required for
the global RTE_* parameters and the Kpp_* product-specific parameters.

More: +

RTE_SMS_PDSE_FLAG: Y  (PDSE flag (Y, N))
RTE_SMS_UNIT: ________  (Non-VSAM disk UNIT type)
RTE_SMS_VOLUME: ________  (Non-VSAM disk VOLSER)
RTE_SMS_MGMTCLAS: ________  (Non-VSAM disk MGMTCLAS)
RTE_SMS_STORCLAS: ________  (Non-VSAM disk STORCLAS)
RTE_SMS_VSAM_VOLUME: ______  (VSAM disk VOLSER)
RTE_SMS_VSAM_MGMTCLAS: ________  (VSAM disk MGMTCLAS)
RTE_SMS_VSAM_STORCLAS: ________  (VSAM disk STORCLAS)

RTE_HILEV: TTEST.&userid____________  (Non-VSAM prod. RK* HLQ)
RTE_VSAM_HILEV: TTEST.&userid____________  (VSAM production RK* HLQ)
RTE_TYPE: SHARING_  (FULL, SHARING)
RTE_SHARE: BASE&SYSALVL._____________  (SMP or shared RTE name)
RTE_X_HILEV_SHARING: TTEST__________  (HLQ of shared RTE)
RTE_TEMS_CONFIGURED_FLAG: Y  (Y, N)
RTE_TEMS_NAME_NODEID: PLB3&SYSNAME.:CMS______________  (e.g., RTE1:cms)
RTE_STC_PREFIX: TSS3_______________  (Started task prefix)
RTE_VTAM_APPLID_PREFIX: TS&SYSCLONE._____________  (VTAM APPLID prefix)
RTE_SYSV_SYSVAR_FLAG: Y  (System variable flag)
RTE_SECURITY_USER_LOGON: NONE____  (RACF, ACF2, TSS, NAM, SAF, None)
RTE_SECURITY_FOLD_PASSWORD_FLAG: Y  (Fold to upper case (Y, N))
RTE_SECURITY_CLASS: ________
RTE_X_SECURITY_EXIT_LIB: TSTEST.&userid.PLB3&SYSNAME..RKANSAMU_______________
GBL_DSN_ACF2_MACLIB: _______________________________________ (If ACF2)
KDS_TEMS_TYPE: REMOTE_  (Hub, Remote)
KDS_TEMS_HA_TYPE: ________  (Hub TEMS type (HA))

Note: Type BACK to go back one panel.
```

These values are read from the model environment's profile library and member
(TSTEST.&userid.PLB3SP13.WCONFIG(PLB3SP13), in this case). You do not need to make any
changes to this panel, because any changes you make will be overwritten when you merge
the LPAR profile from the existing runtime environment. Any changes you need to make, such as
enabling system variable overrides, you will make to the corresponding parameters in the LPAR
configuration profile.

i. Press Enter to proceed to the next panel. The Display PARMGEN Environment Analysis
(KCIP@PG4) panel is displayed.
Review message traffic before proceeding. These Kpp components are installed into the TSTEST.ITM62351.TKANCUS SMP/E target library and are available for (re)configuration and/or upgrade into the RTE=PLB3SP14. Refer to the next panel ("EXCLUDE PRODUCTS FROM PARMGEN CUSTOMIZATION") for a legend of the Kpp components' product names.

34 Components installed in TSTEST.ITM62351.TKANCUS:
KAH KC2 KC5 KDF KDO KDS KD2 KD4 KD5 KGW
KHL KI2 KI5 KMC KMQ KMV KM2 KM5 KNA KN3
KOB KON KQI KRG KRJ KRK KRN KRV KRW
KS3 KT1 KWO KYN

21 Components installed and configured in the RTE profile PLB3SP13:
KC2 KC5 KDF KDO KD2 KD5 KGW KHL KI2 KI5
KMC KMQ KMV KM2 KM5 KN3 KOB KON KQI KS3
KWO

13 Components installed but not configured in RTE profile PLB3SP13:
KAH KDO KD4 KNA KRG KRJ KRK KRN
KRV KRW KT1 KYN
End of data

This panel lists the products installed and available for configuration. The "Components installed and configured in RTE profile PLB3SP14" section lists products that will be configured; this list was inherited from the WCONFIG(PLB3SP13) member specified on the first setup panel (KCIP@PG1).

j. Review the message traffic to determine if there are additional products you want to configure, then press Enter to proceed to the next panel. The Exclude Products from PARMGEN Customization (KCIP@PG5) panel is displayed.

k. Place an X before the ALL entry to configure the same product set as in the runtime environment (PLB3SP13) that you are cloning. Notice the asterisk ("*") next to the products that will be configured into the new PLB3SP14 environment. Specify Y in the "Confirm ==> Y (Y, N) " field.
Select (X) products to EXCLUDE from PARMGEN customization in RTE=PLB3SP14.

When finished, change "N" to "Y" to confirm selections. Confirm ==> Y (Y, N)

<table>
<thead>
<tr>
<th>Kpp</th>
<th>Product Name/Version (Kpp* components configured in RTE profile)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALL</td>
<td>Exclude all not configured in RTE profile</td>
</tr>
<tr>
<td>X</td>
<td></td>
</tr>
<tr>
<td>_</td>
<td></td>
</tr>
<tr>
<td>KCS+</td>
<td>IBM Tivoli OMEGAMON XE for CICS on z/OS V510</td>
</tr>
<tr>
<td>KD0</td>
<td>IBM Tivoli Decision Support for z/OS V181</td>
</tr>
<tr>
<td>KD4</td>
<td>IBM Tivoli Composite Application Manager for SOA V711</td>
</tr>
<tr>
<td>KD5+</td>
<td>IBM Tivoli OMEGAMON XE for DB2 PE/PM V511</td>
</tr>
<tr>
<td>KGU+</td>
<td>IBM Tivoli OMEGAMON XE for CICS TG on z/OS V510</td>
</tr>
<tr>
<td>KI5+</td>
<td>IBM Tivoli OMEGAMON XE for IMS on z/OS V510</td>
</tr>
<tr>
<td>KM+</td>
<td>IBM Tivoli OMEGAMON XE for Messaging - WebSphere MQ Configuration V710</td>
</tr>
<tr>
<td>KMQ+</td>
<td>IBM Tivoli OMEGAMON XE for Messaging - WebSphere MQ Monitoring V710</td>
</tr>
<tr>
<td>KMS+</td>
<td>IBM Tivoli OMEGAMON XE on z/OS V510</td>
</tr>
<tr>
<td>KRS+</td>
<td>IBM Tivoli OMEGAMON XE for Mainframe Networks V510</td>
</tr>
<tr>
<td>KOB+</td>
<td>Tivoli OMEGAMON Enhanced 3270 User Interface V700</td>
</tr>
<tr>
<td>KQI+</td>
<td>IBM Tivoli OMEGAMON XE for Messaging - WebSphere Message Broker V710</td>
</tr>
<tr>
<td>KRG</td>
<td>IBM Tivoli Advanced Audit for DFSMSHsm V240</td>
</tr>
<tr>
<td>KRH</td>
<td>IBM Tivoli Advanced Reporting and Management for DFSMSHsm V240</td>
</tr>
<tr>
<td>KRJ</td>
<td>IBM Tivoli Advanced Allocation Management for z/OS V320</td>
</tr>
<tr>
<td>KRK</td>
<td>IBM Tivoli Automated Tape Allocation Manager for z/OS V320</td>
</tr>
<tr>
<td>KRN</td>
<td>IBM Tivoli Advanced Catalog Management for z/OS V240</td>
</tr>
<tr>
<td>KRV</td>
<td>IBM Tivoli Advanced Backup and Recovery for z/OS V230</td>
</tr>
<tr>
<td>KRW</td>
<td>IBM Tivoli Tape Optimizer for z/OS V220</td>
</tr>
<tr>
<td>KS3+</td>
<td>IBM Tivoli OMEGAMON XE for Storage on z/OS V510</td>
</tr>
<tr>
<td>KWO+</td>
<td>IBM Tivoli OMEGAMON DE on z/OS - OMEGAVIEW and OMEGAVIEW II V510</td>
</tr>
<tr>
<td>KYN</td>
<td>ITCAM for Application Diagnostics on z/OS V710</td>
</tr>
</tbody>
</table>

This panel lists the products installed in the runtime environment and therefore available for configuration (see Appendix F, “Product codes,” on page 265). They are listed by product code (Kpp) and product name and version. To exclude a product from the runtime environment being configured, place an X before its product code.

1. Press Enter. The GBL_USER_JCL(KCIJPCFG) job is displayed.
2. Clone the customized members from the model environment.
   a. Select option 3 from the Workflow menu.
Enter PARMGEN parameter values appropriate for your environment:

- **GBL_USER_JCL**: TSTEST.SYSPLEX.PARMGEN.JCL
- **RTPLIB_HILEV**: TSTEST.&userid
- **RTE_NAME**: PLB3SP14 (Type ? for a list of configured RTEs)

Enter n (1-11) to perform tasks. Status Date

1. **KCIJPCFG** Set up PARMGEN work environment for an RTE. Submitted 2012/08/17
2. **$JOBINDX** Review PARMGEN job index. Viewed 2012/08/17
3. **KCIJPCCF** Clone customized WCONFIG members. (COND)
4. **KCIJPUP1** Update interim libraries and create profiles.
5. **KCIJMC1** Merge profile from backup PLB3SP14 (COND)
6. **KCIJMC2** Merge profile from model RTE. (COND)
7. **KCIJPCNV** Convert an ICAT RTE Batch member. (COND)
8. **PLB3SPI4** Customize PARMGEN configuration profiles.
9. **KCIJPVAL** Validate PARMGEN profile parameter values.
10. **$PARSE** Create the RTE members and jobs.
11. **SUBMIT** Submit batch jobs to complete PARMGEN setup.
12. **U** Utility Access PARMGEN utilities. (Optional)
13. **R** New RTE Reset RTE, Status and Date fields. (Optional)

The KCIJPCCF job is displayed.

The WCONFIG(KCIJPCCF) job clones your user-customized members from an existing WCONFIG library to a new one. If you made any customizations to the WCONFIG override imbeds ($KpsC$, $KpsP*/$Kps@P*, $KpsS$), $JOBCARD$, and common PARMGEN CONFIG profile members such as $GBLSUSR$ and you would like to reuse the same customized members for creating the new runtime environment, use this job to clone the already customized WCONFIG members after you run the KCIJPCFG setup job for the new environment.

b. Review the job to determine if you need to further modify the library concatenated in the OLDWCNFG DDNAME. By default, this DDNAME already points to the WCONFIG library that you specified in the RTE profile library field of the first set up panel. Modify as needed to point to an existing WCONFIG library that contains the customized members you want cloned to the WCONFIG library of the new runtime environment.

c. Submit the job. You are returned to the main Workflow panel.

3. To update the interim libraries and create profiles, select option 4, then submit the WCONFIG(KCIJPUP1) job.
4. After you return to the main Workflow panel, merge the customized values from the model runtime environment into the new environment:

a. Select option 6 to merge the profile from model PLB3SP13.

b. Review and customize the KCIIJPMC2 job. The WCONFIG(KCIIJPMC2) job merges or clones another version of the WCONFIG profiles (the %RTE_NAME% RTE LPAR profile and $GBL$USR) into a refreshed copy or cloned copy. In this case, the PLB3SP14 runtime environment has already been fully customized; its PARMGEN profile already uses symbolics. So to save time, clone its values into PLB3SP14 by editing the OLDMEM() parameter to point to the WCONFIG(PLB3SP13) value.
EDIT TSTEST.&sysuid.PLB3SP14.WCONFIG(KCIJPMC2)
Command ===> Scroll ===> CSR
000068 // MERGECMD Step:
000069 // Merge the changes in %OLDMEM% into %NEWMEM% member.
000071 // ******************************************************
000072 //MERGECMD EXEC PGM=IKJEFT01,DYNAMNBR=99,REGION=4M
000073 //SYSEXEC DD DISP=SHR,
000074 //SYSPRINT DD SYSOUT=*
000075 //SYSTSIN DD *
000078 //PLB3SP14 +
000079 //BATCH +
000080 //OLDMEM(TSTEST.&userid.PLB3SP13.WCONFIG(PLB3SP13)) +
000081 //NEWMEM(TSTEST.&userid.PLB3SP14.WCONFIG(PLB3SP14)) +
000082 /*

The Customize PARMGEN Configuration Profile Members panel is displayed.

b. Customize the LPAR configuration profile:
1) Select option 1 to display the LPAR profile:
KCIP@PG6 ------ CUSTOMIZE PARMGEN CONFIGURATION PROFILE MEMBERS ---------------

Option ==> 1

(Required)* Select option 1 to customize the PLB3SPI4 RTE LPAR profile:

1. PLB3SPI4 RTE LPAR CONFIG profile in WCONFIG

(Conditional)* Select option 2 and/or 3 if applicable to this RTE:

2. $GBL$USR Global parameters CONFIG profile in WCONFIG
   (Required if this is not an ICAT-to-PARMGEN conversion
   or if this is a product upgrade - see F1=Help)

3. PLB3SPI4 System Variables CONFIG profile in GBL_USER_JCL
   (TSTEST.SYSPLEX.PARMGEN.JCL)
   (Required if using user-defined symbols or overriding
   system symbols' resolved values - see F1=Help)

*Note: The PARMGEN configuration profiles above are preserved
(initially created by KCIJPUP1 or KCIJPUP2 job).

(Reference) IBM-supplied default profiles (refreshed by KCIJPUP1 job):

4. $CFG$IBM IBM default RTE LPAR CONFIG profile in WCONFIG

5. $GBL$IBM IBM default Global parameters CONFIG profile in WCONFIG

6. $SYSIN $PARSE/$PARSESV SYSIN controls

(Optional) Select option 7 for member list of the WCONFIG library:

7. WCONFIG TSTEST.&userid.PLB3SPI4.WCONFIG

---écran---

2) Change all references to PLB3SPI3 (the model environment) to PLB3SPI4 (the current environment).

**Important:** In a system variables scenario, the only parameters that must have a nonsymbolic value are RTE_NAMESV, RTE_HILEVSV, and RTE_VSAM_HILEVSV parameters, as these are used directly in the WKANSAMU(KCIJV) jobs. In the RTE_NAMESV parameter below, the value must remain PLB3SPI4.

```shell
ISREDDE2 TSTEST.&userid.PLB3SPI4.WCONFIG(PLB3SPI14)
Command ===> C PLB3SPI13 PLB3SPI14 ALL Scroll ===> CSR
***** ***************************** Top of Data *****************************
aira. - - - - - - - - - - - - 352 Line(s) not Displayed
470 RTE_NAMESV PLB3SPI14
- - - - - - - - - - - - - - - - - - - - - - 904 Line(s) not Displayed
***** ***************************** Bottom of Data *****************************
```

3) You must also change any DB2, IMS subsystem names, CICS regions specific to the LPAR, MQ manager and subsystem names, and their related load libraries, inherited from the model runtime environment.

c. Customize the system variables configuration profile to define all user symbols used in the LPAR profile.

1) From the Customize PARMGEN Configuration Profile Members, select option 3.

The TSTEST.SYSPLEX.PARMGEN.JCL(PLB3SPI14) system variables profile is displayed.

2) Because the PLB3SPI14 LPAR profile inherited its symbolics from the cloned PLB3SPI3 LPAR profile, simply replace the contents of TSTEST.SYSPLEX.PARMGEN.JCL(PLB3SPI14) with the TSTEST.SYSPLEX.PARMGEN.JCL(PLB3SPI13) system variables profile parameters, then change the LPAR-specific symbolic values accordingly.
For example, the PLB3SP14 environment sets the RTE_X_SYSV_OVERRIDE_SYMBOLS parameter set to Y, which means override the local system symbols. Therefore, system symbols intended for the SP13 LPAR (like SYSNAME, SYSCLONE and SYSHOSTNAME) must be changed accordingly. The other system variables still resolve to the same values as both the remote Tivoli Enterprise Monitoring Server and the agents in the PLB3SP13 runtime environment. PLB3SP13 and this cloned PLB3SP14 environment now point to the same hub monitoring server; the respective stand-alone agents in each environment continue to report to their primary local monitoring server.
Since a secondary monitoring server was enabled for these agents, their backup (secondary) monitoring servers now both point to the same hub monitoring server (the static hub monitoring server, started task TSS0DSH1, in the PLB3SP14 environment).

If you need the PLB3SP13 remote monitoring server and agents to use other values, make further edits to the TSTEST.SYSPLEX.PARMGEN.JCL(PLB3SP14)'s System Variables profile.

3) Save the changes and exit the profile. Press F3 to return to the main Workflow panel.

6. Validate the profile parameter values you specified.
   a. On the Workflow main panel, select option 9.

   The KCIJPVAL job is displayed.
   b. Review the notes and submit the job. The parameter validation report is written to \&hilev\&rte.WCONFIG($VALRPT). Enter option 9s view the report.
   c. If the job gets a condition code greater than 4 (COND CODE 0004), review the $VALRPT. Return to the previous step and correct the parameter values, then resubmit the validation job.

7. Create the WK* runtime members and the WKANSAMU jobs:
   a. From the main Workflow menu, select option 10.
The $PARSE/$PARSESV: Create the RTE Members and Jobs (KCIP@PRS) panel is displayed.

b. Select option 1 to submit the generated $PARSESV composite job.

tip: In cases where you need to create or recreate only certain runtime members, review the other library-specific $PARSExx utility jobs.

c. Review the job, then submit it.

8. Complete setup of the environment.
   a. From the Workflow menu select option 11.
The Submit Batch Jobs to Complete PARMGEN Setup (KCIP@SUB) panel is displayed.

b. Select option 1 if you want to submit the KCIJVSUB master SUBMIT job, or select the other options to submit each job individually. If you select option 1, review the KCIJVSUB job first, and edit the job according to which the conditional jobs need be submitted automatically.

If you select option 1, you are prompted with the following message:
c. Press Enter. The following message is displayed:

```
  KCIP0PGP ------------------ PARMGEN MESSAGES ---------------------
  Command ===> 

In a system variable environment, job KCIJVSUB runs $PARSESINV
to customize then submit the KCIJPSUV job. EDIT and modify
the KCIJPSUV job as appropriate. See comments in the JCL for
further information.
DO NOT MANUALLY SUBMIT THE KCIJPSUV JOB!
```


d. Press ENTER to edit the KCIJPSUV job, then press F3, and you will automatically be presented
with the KCIJVSUB job for submission.

**Results**

You have created runtime environment PLB3SP14.

**What to do next**

Perform the applicable “Complete the configuration” steps in your planning and configuration guides.
Chapter 8. Using the Configuration Tool batch mode to replicate a configured environment

After you have used the Configuration Tool in interactive mode to configure a runtime environment, you can use the tool in batch mode to replicate that environment to other z/OS images. Using batch mode, you can install a complete runtime environment by submitting a single batch job, rather than configuring it panel by panel.

About this task

This section focuses on the use of batch mode to create new runtime environments based on a model environment. However, the Configuration Tool batch mode facilities can be used for several different purposes:

- Backing up the configuration parameters of an existing runtime environment.
- Restoring a runtime environment based on a saved configuration.
- Reconfiguring an existing runtime environment when something in the environment has changed (such as a port or subsystem name).
- Creating a new runtime environment based on an existing one.

The principal components of batch mode processing are the batch mode job and the batch parameter member. The batch mode job validates the input configuration parameters and generates the jobs required to build, configure, and load a runtime environment. The job uses a single parameter member containing all the configuration parameters as input for the jobs it generates. The parameter member is derived from an existing runtime environment using a Configuration Tool utility. You can adjust the parameters before using the parameter member to create a new runtime environment or to reconfigure or restore an existing one.

The topics in this section describe the tasks to be performed for batch mode processing in order.

- **“Step 1. Create the batch mode job” on page 180**
  The batch mode job, CICATB, is used to create and submit the jobs that build, configure and load a runtime environment using parameters exported from an existing, or model, runtime environment. It can also be used to validate the parameters. The batch mode job does not exist by default; you must generate it by using the Configuration Tool’s Create Batch Mode Job service. Creating the batch mode job also creates the KCISETUP job, which you invoke to set up the environment required by the batch utilities.

- **“Step 2. Create and clone a batch mode parameter member” on page 183**
  A batch mode parameter member contains the configuration values for all the products configured in an existing runtime environment. The member is created by using the Create/Clone Batch Mode Parameters utility to export the configuration values into a library member. You clone the member and modify the values as required to create a new runtime environment.

- **“Step 3. Adjust and validate the parameters” on page 186**
  If you do not want to create an exact copy of an existing runtime environment, you adjust the configuration parameters in the cloned parameter member. You validate those changes by using the SCAN function of the CICATB job.

- **“Step 4. Generate sample transport JCL” on page 188**
  The Configuration Tool provides several sample DFDSS tape jobs that you can use to transport the runtime environment or the configuration jobs or parameter member required for creating the runtime environment to the target system. It also provides sample scenarios for using the jobs.

- **“Step 5. Create a new runtime environment” on page 190**

© Copyright IBM Corp. 1994, 2013
You can create new runtime environments on target z/OS images using one of several methods.

Tip: "Step 6. Complete the configuration of a new runtime environment" on page 196

You must complete steps outside the Configuration Tool on the target image just as you did on the model image. This section explains how to find information about the required steps.

Tips:
- Read this entire section before you begin performing the tasks described.
- If you intend to use batch mode processing to replicate your runtime environments, enable system variable support when you configure the model runtime environment. System variable support enables runtime environments to inherit the system values for the systems where they are defined. See Appendix A, "Enabling system variable support," on page 199 for detailed instructions.

Step 1. Create the batch mode job

The CICATB batch mode job generates the jobs that create the replicated runtime environment. CICATB can also be used to validate the parameters in a parameter member.

About this task

There is no default batch mode job. You create one by running the Create Batch Mode Job service. Running Create Batch Mode Job also generates KCISETUP, which is invoked to set up the Configuration Tool environment required by the batch utilities. (See "KCISETUP: Set up the environment for batch utilities" on page 236 for more information.)

Tip: After KCISETUP and CICATB are created, they can be used for all subsequent batch parameter member processing unless the ISPF environment changes. If the ISPF environment changes, recreate KCISETUP and CICATB before you do any further batch processing.

You must exit the Configuration Tool before the CICATB job can run.

Tips:
- "Creating the batch mode job"
- "Batch mode job submission options" on page 181
- "Batch mode configuration jobs" on page 182
- "Batch mode job report" on page 182

Creating the batch mode job

About this task

Follow this procedure to create the batch mode job.

Procedure

1. Start the Configuration Tool by entering the following command in a TSO command panel: ===>
   EXEC 'shilev.INSTLIB'
2. From the Main Menu, select Option 1 (Set up work environment) and press Enter.
   The Set up work environment panel is displayed.
3. Select option 3 (Create batch mode job).
   The Create batch mode job option generates the CICATB JCL, which runs the configuration process in batch mode. This option also creates the KCISETUP REXX exec. When invoked, this exec enables ISPF to use macros supplied by the Configuration Tool.
   The CICATB job is created in the shilev.INSTJOBS data set. You use this job to create new runtime environments with the configuration values from the original production runtime environments.

Tips:
To see the complete path of the library the batch mode job was created in, press F1. A text box at the bottom of the panel shows the complete path. The text box is displayed only if F1 is pressed immediately after Create batch mode job is selected. Otherwise, the general help for the panel is displayed.

When you create the CICATB batch job member in the shilev.INSTJOBS data set, the Configuration Tool also creates the batch jobcard ISPF table KCITPIG1 in the shilev.INSTDATA data set. The contents of the KCITPIG1 table are used as the standard jobcard in batch configuration jobs. If you modify the jobcard on the Specify JCL Options panel (Figure 27 on page 106), or if the ISPF environment changes in some other manner, recreate the CICATB batch job to refresh the KCITPIG1 jobcard table before running CICATB. When the ISPF environment changes, CICATB must be recreated to modify the DDNAME concatenation accordingly, so that the job no longer points to the old ISPF libraries.

4. Wait until you see the CICATB JOB CREATED message in the upper-right corner of the panel.
5. Open the CICATB JCL and examine it. The SYSTSIN DD statement in the JCL contains several parameters that you must customize before you submit the job.

**BATCHLIB**
- The library that contains the batch runtime environment parameter member. The default is &shilev.INSTJOBS.

**BATCHMEM**
- The member name of the batch runtime environment parameter member (the name must be the same as the name of the runtime environment you intend to create). You create this member in the following step.

**SUBMIT**
- The type of processing. You specify different dispositions depending on the step you are performing and the requirements of the environment. See "Batch mode job submission options."

### What to do next

Do not submit the CICATB job yet. Read the rest of the topics in this section first. Your choice of a transport method ("Step 5. Create a new runtime environment" on page 190) and related decisions affect the other tasks to be performed.

### Batch mode job submission options

#### About this task

The SUBMIT parameter in the CICATB job specifies a disposition for runtime environment configuration jobs generated by CICATB. The valid values are:

**YES**
- Create and submit the runtime environment configuration jobs. If the jobs already exist, such as when you are reconfiguring the runtime environment, the jobs are regenerated but are not submitted for execution. If you want to submit the jobs in such an environment, use the RERUN option.

If your site uses a batch job utility that analyzes the submitted JCL before the job is sent to the execution queue, use the YES option only if the jobs are single-threaded through the analyzer. Otherwise, SUBMIT(YES) might cause CICATB-generated jobs to be executed out of sequence.

**Important:**

If the runtime environment contains any user-modified elements that will be modified by any configuration job generated by the CICATB job, running CICATB creates reports identifying those elements. Each report is named for the job that will modify user-modified elements. If a job named pp@nssss will modify user-modified elements, a report is stored in pp@nssss. Examine the reports and...
the user-modified elements they identify before running the configuration jobs. The first time you run the CICATB job, it is best to specify SUBMIT(NO) and examine the report on user-modified elements before submitting the configuration jobs. If the runtime environment contains any user-modified elements and if you intend to run the CICATB job with SUBMIT(RERUN) or SUBMIT(YES) specified, back up the runtime libraries before running CICATB.

**RERUN**
Create and submit the runtime environment configuration jobs, regardless of whether the jobs already exist. Do not use this option if the runtime environment contains any user-modified elements. Otherwise, use it to avoid deleting the configuration jobs in the `shilev.INSTJOBS` data set if the job names already exist, before reconfiguring the runtime environment in batch mode.

**NO**
Create but do not submit the runtime environment configuration jobs. Used when the runtime environment will be created on a different z/OS image, or when manual control over job submission is desired.

**SCAN**
Validate the input parameter member and produce a report. The scan validates that the required variables are defined, and that the defined values meet the syntax requirements for the respective variables.

**SCAND**
Scan the input parameter member and produce a detailed data dictionary of all the runtime environment parameters. The dictionary provides a description and syntax rules for each parameter.

For more information on the SCAN and SCAND dispositions, see [“Step 3. Adjust and validate the parameters” on page 186](#). See the CICATB job itself for information on other input control parameters.

**Batch mode configuration jobs**
**About this task**
If there are no parameter errors, running CICATB with either SUBMIT(YES), SUBMIT(RERUN), or SUBMIT(NO) produces a set of runtime environment configuration jobs named 

```
pp#nssss
```

where:

- `pp` is the 2-character product code (DS is the product code for the Tivoli Enterprise Monitoring Server).
- `n` is the number of the job
- `ssss` is the JCL suffix for the runtime environment

The configuration jobs are saved in the `shilev.INSTJOBS` data set.

**Batch mode job report**
**About this task**
Part of the CICATB job output is a report that has several sections, depending on the disposition and outcome. The report is in the KCIPMRPT DDNAME of the CICATB job output. In addition to KCIPMRPT, the CICATB report is also saved as a member in the `shilev.INSTJOBS` library. The name of the report member depends on the disposition of the SUBMIT command in CICATB job, as shown in [Table 10 on page 183](#).
Table 10. CICATB job reports

<table>
<thead>
<tr>
<th>Disposition</th>
<th>Result</th>
<th>Report name</th>
<th>Report contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUBMIT(SCAN)</td>
<td>No errors</td>
<td>CB#sSSS</td>
<td>Two sections:</td>
</tr>
<tr>
<td></td>
<td>Errors</td>
<td>CB#S</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1. Parameter errors</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2. Parameters you have changed</td>
</tr>
<tr>
<td>SUBMIT(SCAND)</td>
<td>Any</td>
<td>CB#SDICT</td>
<td>Detailed parameter dictionary with four sections:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1. List of parameters and descriptions.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2. List of parameters, variable names, and default values.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3. List of parameters, lengths, types, and subtypes, with indication of whether CAPS(ON) or CAPS(OFF) is set.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>4. List of numeric parameters, with minimum, maximum, and default values.</td>
</tr>
<tr>
<td>SUBMIT(NO), SUBMIT(YES), SUBMIT(RERUN)</td>
<td>No errors</td>
<td>CB#RsSSS</td>
<td>Five sections:</td>
</tr>
<tr>
<td></td>
<td>Errors</td>
<td>CB#R</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1. Parameter errors.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2. Parameters you have changed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3. Jobs sorted by generation sequence.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>4. Jobs sorted by member name.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>5. Components configured in the runtime environment.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>When there are errors, sections 3, 4, and 5 are empty.</td>
</tr>
</tbody>
</table>

**Step 2. Create and clone a batch mode parameter member**

**About this task**

A batch mode parameter member contains configuration values extracted from an existing runtime environment. The Configuration Tool creates, or regenerates, a parameter member each time the runtime environment Load function runs. This member can be used as a backup from which the current runtime environment can be recreated. The batch mode parameter member can also be cloned, modified if necessary, and used with the batch mode job to create a new runtime environment.

You clone a batch mode parameter member by using the Create/Clone batch mode parameters utility. The name you give to the new member becomes the mid-level qualifier for the new runtime environment you create by using the member.

The Create/Clone batch mode parameters utility offers three options:

1. **Create batch mode parameter deck.**
   
   Use this option to create or recreate a batch mode parameter deck for the selected runtime environment. You might use this option, for example, if you want to create a parameter deck without loading the runtime libraries.

2. **Clone batch mode parameter deck, unmodified.**
Use this option to create a clone of a batch mode parameter deck that includes all the products and components in the runtime environment. If a parameter deck does not already exist, the utility creates one and then creates the clone.

3. **Clone batch mode parameter deck, with modifications.**
   Use this option if you want to create a clone of a batch mode parameter deck, but you do not want to include all of the products configured in the runtime environment in the deck. During the cloning process, you are presented with a panel displaying a table of all of the components configured in the RTE, from which you can select components to be excluded from the cloning process.

Every parameter deck requires a unique 1- to 4-character runtime environment JCL suffix. During the cloning process, you are prompted to enter one for the clone member. The clone member will contain this new value.

Complete the following steps to generate and clone a parameter member:

**Procedure**

1. Start the Configuration Tool by entering the following command in a TSO command panel:
   ```
   ====> EXEC 'shilev.INSTLIB'
   ```
2. Navigate to the Runtime Environments (RTEs) panel:
   a. On the Main Menu panel select option 3 (Configure products).
      The Configure Products panel is displayed.
   b. Select option 1 (Select product to configure).
      The Product Selection Menu panel is displayed.
   c. Select a product that is configured in the runtime environment from which you want to export the parameters.
      The Runtime Environments (RTEs) panel is displayed.
   
   **Tip:** Enter `panelid` in the `OPTION` `=>` line to see the panel ID in the upper left corner of the panel.
3. Type Z next to the runtime environment you want to replicate and press Enter.
   The RTE Utility Menu is displayed.
4. Select option 1 (Create/Clone batch mode parameters) and press Enter.
   The Create/Clone Parameter Deck Menu is displayed (see Figure 44 on page 219).
5. Select option 2 (Clone batch mode parameter deck, unmodified) to create a cloned parameter member that contains all the same products as the model runtime environment, or select option 3 (Clone batch mode parameter deck, modified) to create a clone parameter member that excludes some of the products in the model runtime environment.
6. Press Enter. The Create/Clone Batch Mode Parameters panel is displayed (see Figure 38 on page 185).
The panel contains default values for the input parameter member for the existing runtime environment. By default, the name of the parameter member is the name of the runtime environment and it is created in the `shilev.INSTJOBS` library. You can edit these values if you want. If the parameter deck does not already exist, the utility creates it.

Tip: The cloned parameter member (Output parameter deck member) is a copy of the original (Input parameter deck member). If you provide a different value for the output Parameter library or for the Clone member name, creating the clone does not destroy the original.

7. Specify appropriate values for the Output parameter deck member section of the panel:

   **Parameter library**
   The library where the clone parameter member will be created. By default, the library is `shilev.INSTJOBS`. You can change the library name, but the library that you specify must already exist. If the library does not exist, the Configuration Tool does not create the parameter member and display the message LIBRARY ERROR in the upper right corner of the panel.

   **Clone member name**
   The name of the member that contains the parameters. This name becomes the mid-level qualifier for the new runtime environment.

   **Edit after create**
   Y or N, to specify whether the clone parameter member is to be opened for editing after it is created.

8. Press Enter.

   If the input parameter deck member does not already exist, the following message is displayed:
   The parameter deck member `rte` does not exist.
   Press Enter to continue and create `rte` parameters or F3 to cancel.

9. Press Enter to create the parameter deck.

   After parameter member for the model runtime environment is created, the following message is displayed:
   BATCH PARMS CREATED

10. Press Enter again to generate the clone parameter deck.

    The following message is displayed:

---

**Figure 38. Create/Clone Batch Mode Parameters panel**

```plaintext
KCIPCLO1 ------ CREATE/CLONE BATCH MODE PARAMETERS / RTE: rte -----------
COMMAND >>>

Input parameter deck member:
Parameter library ==> shilev.INSTJOBS
Parms member name ==> rte

Output parameter deck member:
Parameter library ==> shilev.INSTJOBS
Clone member name ==>
Edit after create ==> Y

Enter=Next  F1=Help  F3=Back
```
11. Type the JCL suffix for the new runtime environment and press Enter. If you specified Y for Edit after create, the clone member is displayed.

12. Edit the new library member to reflect the settings specific to the z/OS image where the new runtime environment will be created, as discussed in "Step 3. Adjust and validate the parameters."

Tip:

Use the KCICGPHP utility to display help for the parameters while editing the batch parameter member. Use either of the following methods:

- From an ISPF Edit command-line, enter the command KCICPGHP, position the cursor on the row that contains a batch parameter, and then press Enter.
- Position your cursor on the row that contains a batch parameter and then select the PF key assigned by the Configuration Tool PF Key Manager.

For more information on setting up and using the help, see “KCICPGHP: Display help for configuration parameters” on page 238.

13. Press F3 to save the changes and close the parameter member. After the clone parameter member for the target runtime environment is created, the following message is displayed:

```
rte CLONED
```

---

**Step 3. Adjust and validate the parameters**

The parameters in the clone batch mode parameter member describe the configuration of the model runtime environment from which they were exported. You might want to adjust the parameter values for the new environment.

**About this task**

Compare the values that were used for the model runtime environment with the values of the new runtime environment. (You can use a scan report on the parameter member for the model environment to review the values.) If a variable value must be changed for the new runtime environment (for example, the DB2 subsystem name or z/OS hostname), open the clone parameter member in edit mode. Adjust the parameters as necessary and save the changes.

After modifying the values, validate the parameters in the parameter member before you create the new runtime environment by running the CICATB job with SUBMIT(SCAN). The job checks, for example, that a numeric parameter is in the allowed range, or that the length of a character parameter does not exceed the maximum number of allowed characters. Some validations performed by the Configuration Tool in interactive mode are not performed in batch mode.

To validate the parameters:
**Procedure**

1. If the CICATB job is not already open in the Configuration Tool, open it from the library in edit mode.

2. Scroll down until you come to the SYSTSIN section.

3. Replace the default value for BATCHMEM with the name of your new parameter member as shown in the following example. Set the disposition for SUBMIT to SCAN (the default).

   //SYSTSIN DD *,DCB=BLKSIZE=80
   PROFILE PREFIX(TST)
   ISPSTART +
   CMD(EX 'shilev.INSTLIB(KINSTALL)' +
       'INSTLIB(shilev.INSTLIB) +
       BATCHLIB(shilev.INSTJOBS) +
       BATCHMEM(SYSB) + Replace name of current with name of new parameter member
       SUBMIT(SCAN'))
   /*
   */
   //

4. Save the changes.

5. Submit the job.

6. Press F3 to leave the CICATB job. If you are editing the job in INSTJOBS in the Configuration Tool, exit the tool. Otherwise the job can not continue.

7. Switch to SDSF and check the status of your job. If it has finished, use ? to select it and see the separate logs. Check SYSTSPRT first for error messages.

   It is not sufficient to look at the return code of the job. This will be 0 even if errors occurred. Check SYSTSPRT. If the scan did not find any error in the parameters, the scan is successful and the
   SYSTSPRT looks similar to the one in

   ![Figure 39](image_url)

   **Figure 39. SYSTSPRT for successful scan**

**Results**

The scan report is in KCIPMRPT. Additionally, if the scan runs with no errors, a report named CB#ssss is generated in the INSTJOBS library (where ssss is the JCL suffix for the runtime environment). If errors are

If you do not get the success message in SYSTSPRT, the report KCIPMRPT is not generated. Check for one of the following errors:

**BATCHMEM Parameter Value State Error**

The parameter member was not found. Most likely the default value was not changed in the CICATB job. Open CICATB and check that the data set in BATCHMEM exists in the library that is specified in BATCHLIB.

**INSTLIB Parameter Value State Error**

The INSTLIB library was not found. Open CICATB and correct the path name.
KCICCSTR NO PRODUCTS
The target libraries that were installed by SMP/E were not found. Navigate to the Set Up Configuration Environment panel (from the Main Menu select Set up work environment, and then select Set up configuration environment) and verify that the value for SMP/E data sets Target is set correctly.

SCANNER ERROR cccc FROM KCICRTE
A SCANNER ERROR indicates that a parameter did not pass validation. Open the report KCIPMRPT and look at Section 1. The parameter that causes the error is listed there. For example:

************************************************************************
* SECTION 1: PARAMETER ERRORS *
************************************************************************
PARAMETER INPUT ERROR
NAME LINE# DESCRIPTION
------------------------- ----- ----------------------------------------
KDS_CMS_TCP_PIPE_PORT 151 Invalid Parameter Value: Value greater-than allowable maximum of 65535

Edit the batch parameter member in INSTJOBS to change the value to an allowed value. Save the changes and run the CICATB job again.

If you are not sure what value a variable can have (for example, the allowed range for a number or the allowed number of characters for a string) you can use the detailed data dictionary report of CICATB.

To generate the detailed data dictionary report:
1. Open the CICATB job in edit mode.
2. Change the disposition for SUBMIT to SCAND.
3. Save the changes and submit the job.
4. If CICATB is in the INSTJOBS library, press F3 to leave the CICATB job and press F3 again to exit the browse mode of the INSTJOBS library. Otherwise the job can not continue.
5. Start SDSF and check the status of your job.
   If it has finished, select it with ? to see the separate logs.

The detailed data dictionary report is in KCIPMRPT. Additionally, the report is saved in the CB#SDICT member of the shilev.INSTJOBS data set.

For questions on allowed characters per string go to

SECTION 3: *****************************************************************
* SECTION 3: LENGTHS, TYPE, SUBTYPE, CAPS *
************************************************************************

For questions on allowed value range for numbers go to SECTION 4:

************************************************************************
* SECTION 4: NUMERIC PARAMETERS WITH MIN, MAX, AND DEFAULTS
classification _____________________________
************************************************************************ and search for your variable.

After you have fixed the problem, rerun the CICATB job with SUBMIT(SCAN). Before you move on to the next step, make sure that the validation does not report any more errors.

Step 4. Generate sample transport JCL
The Configuration Tool provides sample configuration scenarios and DFDSS tape jobs for transporting the runtime environment or the components required for creating the runtime environment to the target system.
About this task

Take these steps to generate the samples:

Procedure

1. If necessary, start the Configuration Tool by entering the following command in a TSO command panel:
   ```
   ===> EXEC 'shilev.INSTLIB'
   ```

2. Navigate to the Runtime Environments (RTEs) panel:
   a. On the Main Menu panel select option 3 (Configure products).
      The Configure Products panel is displayed.
   b. Select option 1 (Select product to configure). The Product Selection Menu panel opens.
   c. Select a product that is configured in the runtime environment from which you want to export the parameters. The Runtime Environments (RTEs) panel is displayed.

3. Type Z next to the runtime environment you want to generate the samples for and press Enter.
   The RTE Utility Menu panel opens.

4. Select 5 (Generate sample transport JCL) and press Enter. This action causes several sample transport jobs to be generated in the rhilev.rte.RKANSAMU library. The generated members include:
   $XPRTNDX
   Provides descriptions of all generated members.

   XCNFG#01
   Describes a typical transport scenario for one system and one runtime environment.

   XCNFG#02
   Describes a remote installation scenario by using the Configuration Tool on the target image.

   XDFDMP01
   This sample DFDSS tape job executes a data set DUMP from a runtime environment at a model system to a target site at initial build distribution using configuration scenario XCNFG#01.

   XDFRST01
   This sample DFDSS tape job executes a data set RESTORE of a runtime environment at a model system to a target site at initial build distribution using configuration scenario XCNFG#01.

   XDFDMP02
   This sample DFDSS tape job executes a data set DUMP from a runtime environment at a model system to a target site at maintenance distribution using Configuration Scenario XCNFG#01.

   XDFRST01
   This sample DFDSS tape job executes a data set RESTORE of the target libraries and Configuration Tool data set from the model system to the target site at initial build distribution using configuration scenario XCNFG#01.

   XDFDMP03
   This sample DFDSS tape job executes a data set DUMP of the target libraries and Configuration Tool data set from the model system to the target site at initial build distribution using configuration scenario XCNFG#02.

   XDFRST03
   This sample DFDSS tape job executes a data set RESTORE of the target libraries and Configuration Tool data set to the remote site at initial build distribution using configuration scenario XCNFG#02.
This sample DFDSS tape job executes a data set DUMP of the target libraries from the model system to the remote site at maintenance distribution using configuration scenario XCNFG#02.

**Step 5. Create a new runtime environment**

You can use any of several methods to create a new runtime environment.

**Before you begin**

Before using the procedures in the following sections, make sure that sufficient space and library security authorizations exist on the target systems. These procedures assume that you have already set up the batch processing environment and created the batch processing job on the first image, as described in “Step 1. Create the batch mode job” on page 180, and generated the batch mode parameter member, as described in “Step 2. Create and clone a batch mode parameter member” on page 183.

**About this task**

These methods and their advantages and disadvantages are summarized in Table 11.

<table>
<thead>
<tr>
<th>Method</th>
<th>This method uses</th>
<th>Advantages and disadvantages</th>
</tr>
</thead>
</table>
| Defining a runtime environment on a z/OS image using shared DASD | • Interactive and batch modes  
• The Configuration Tool on the master z/OS image to create a runtime environment accessible to the target image. | Advantages of this method:  
• The Configuration Tool located on the master z/OS image contains the configuration information for all images.  
• Only one copy of the runtime libraries is created.  
• Only one batch job is submitted.  
The disadvantage of this method is that it is applicable only to z/OS images with shared DASD. |
| Transporting a runtime environment from one image to another | • Interactive and batch modes  
• The Configuration Tool on the first image to create a runtime environment. After the runtime environment has been defined, you use sample transport jobs to ship the runtime libraries and parameters to other images. | Advantages of this method:  
• The Configuration Tool on the first image contains the configuration information for all images.  
• Only one batch job is submitted.  
• This method can be used for configuring the other z/OS images that do not share DASD.  
The disadvantage of using this method is that multiple copies of the runtime libraries are created. |
<table>
<thead>
<tr>
<th>Method</th>
<th>This method uses</th>
<th>Advantages and disadvantages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transporting runtime environment batch jobs from one z/OS image to another</td>
<td>• Interactive and batch modes</td>
<td>Advantages of this method:</td>
</tr>
<tr>
<td></td>
<td>• The Configuration Tool on the first image to create a set of batch jobs that can build a runtime environment. After the jobs are created, you use sample transport jobs to ship the batch jobs to other images. The jobs are manually submitted on the other images to create the runtime libraries and parameters.</td>
<td>• The Configuration Tool on the first image contains the configuration information for all images.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• This method can be used for configuring the other z/OS images that do not share DASD.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Only one copy of the runtime libraries is created.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The disadvantage of using this method is that you must manually submit a series of batch jobs, or run a product-provided CLIST to submit the jobs that will create the runtime environment.</td>
</tr>
<tr>
<td>Transporting runtime environment batch mode parameters from one z/OS image to another equipped with the Configuration Tool</td>
<td>• Interactive mode</td>
<td>Advantages of this method:</td>
</tr>
<tr>
<td></td>
<td>• The Configuration Tool in interactive mode on the first image to export an existing runtime environment. After the runtime environment parameters have been collected, you use the sample transport jobs to ship the batch mode parameters to other images. Then you run the Configuration Tool in batch mode on the other images to create the runtime libraries and parameters.</td>
<td>• This method can be used for configuring the other z/OS images that do not share DASD.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Only one copy of the runtime libraries is created.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Only one batch job is submitted.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The disadvantage of using this method is that the Configuration Tool located on the first image does not contain the configuration information for all images.</td>
</tr>
</tbody>
</table>

The scenarios in this section use the sample DFDSS tape jobs created by the Configuration Tool, but tape is not the only mode of transport you can use between systems that do not share DASD. If the target system is connected to the network, you can modify the samples and dump the files to data sets, from which the files (and the jobs to restore them) can be sent to the target system through the network by FTP or TSO SEND/RECEIVE.

Consult the configuration guides for the products configured in the runtime environment for post-configuration steps that might be required on the target system.

**Scenario A: Defining a runtime environment on a z/OS image using shared DASD**

**About this task**

Follow these steps to define a runtime environment on a z/OS image using shared DASD.

**Procedure**

1. Exit the Configuration Tool.
2. If you have not already done so, perform a scan on your runtime environment parameters:
a. Edit CICATB, updating the BATCHLIB and BATCHMEM parameters as necessary, and setting the SUBMIT parameter to SCAN.
b. Submit the CICATB job to scan your runtime environment parameters.
c. Exit the shilev.INSTJOBS data set so that CICATB can run.
d. Verify that the job completes successfully, review the parameter report, correct any errors in the parameter member, and repeat the scan until a clean report is generated.

3. Create a new runtime environment that is accessible to the target image:
a. Edit CICATB again, setting the SUBMIT parameter to YES. Setting SUBMIT(YES) automatically submits the configuration jobs that allocate and populate runtime libraries, unless the jobs already exist in the shilev.INSTJOBS data set. If you are using batch processing in a cloned environment, you can specify SUBMIT(RERUN) to submit the configuration jobs automatically even if the jobs already exist.
   If the runtime environment contains user-modified elements, back up the runtime libraries before submitting CICATB with SUBMIT(RERUN) or SUBMIT(YES) specified.
b. Submit the CICATB job to create the runtime environment.
c. Exit the shilev.INSTJOBS data set so that CICATB can run.
d. Verify that the job completes successfully.

What to do next

Perform any manual configuration steps on the target image, such as:
- Copying procedures to PROCLIB
- Copying VTAM definitions to VTAMLST
- APF-authorizing libraries

See “Step 6. Complete the configuration of a new runtime environment” on page 196 for more information.

Scenario B: Transporting a runtime environment from one z/OS image to another

About this task

Follow these steps to create a runtime environment and transport it from one z/OS image to another:

Procedure

1. Exit the Configuration Tool.
2. If you have not already done so, perform a scan on the parameters for the new runtime environment:
   a. Edit CICATB, updating the BATCHLIB and BATCHMEM parameters as necessary, and setting the SUBMIT parameter to SCAN.
   b. Submit the CICATB job to scan your runtime environment parameters.
   c. Exit the shilev.INSTJOBS data set so that CICATB can run.
   d. Verify that the job completes successfully, review the parameter report, correct any errors in the parameter member, and repeat the scan until a clean report is generated.
3. Create a new runtime environment that is accessible to the target image runtime environment:
   a. Edit CICATB again, setting the SUBMIT parameter to YES. This submits the runtime environment configuration jobs that allocate and populate runtime libraries.
   b. Submit the CICATB job to create the runtime environment.
   c. Verify that the job completes successfully.
4. Start the Configuration Tool again.
5. Navigate to the Runtime Environments (RTEs) panel:
   a. On the Main Menu panel select option 3 (Configure products).
      The Configure Products panel is displayed.
   b. Select option 1 (Select product to configure). The Product Selection Menu panel is displayed.
   c. Select a product that is configured in the model runtime environment that you are going to replicate. The Runtime Environments (RTEs) panel is displayed.

6. Generate the sample transport JCL:
   a. Type Z next to the runtime environment you want to transport and press Enter.
      The RTE Utility Menu panel is displayed.
   b. Select 5 (Generate sample transport JCL) and press Enter.
      This action causes several sample transport jobs to be generated in the rhlev.rte.RKANSAMU library. For more information about these jobs, see “Step 4. Generate sample transport JCL” on page 188.

7. Use the sample jobs appropriate to this scenario:
   • Use sample job XDFDMP01 on the first image to dump the runtime libraries
   • Use sample job XDFRST01 on the target image to restore the runtime libraries

What to do next

After you have transported the runtime environment to the target images, perform any required manual configuration steps on the target image, such as:
   • Copying procedures to PROCLIB
   • Copying VTAM definitions to VTAMLST
   • APF-authorizing libraries

See “Step 6. Complete the configuration of a new runtime environment” on page 196 for more information.

Scenario C: Transporting runtime environment batch jobs from one z/OS image to another equipped with the Configuration Tool

About this task

Follow these steps to transport runtime environment batch jobs from one z/OS image to another equipped with the Configuration Tool.

Procedure

1. Exit the Configuration Tool.
2. If you have not already done so, perform a scan on your runtime environment parameters:
   a. Edit CICATB, updating the BATCHLIB and BATCHMEM parameters as necessary, and setting the SUBMIT parameter to SCAN.
   b. Submit the CICATB job to scan your runtime environment parameters.
   c. Verify that the job completes successfully, review the parameter report, correct any errors in the parameter member, and repeat the scan until a clean report is generated.
3. Create the runtime environment generation jobs:
   a. Edit CICATB again, setting the SUBMIT parameter to NO.
      This creates the runtime environment configuration jobs that allocate and populate runtime libraries, but does not run the jobs.
   b. Submit the CICATB job
   c. Verify that the job completes successfully.
You have now successfully created a set of runtime environment configuration batch jobs that must be transported to the target image.

4. Start the Configuration Tool again.

5. Navigate to the Runtime Environments (RTEs) panel:
   a. On the Main Menu panel select option 3 (Configure products). The Configure Products panel is displayed.
   b. Select option 1 (Select product to configure).
      The Product Selection Menu panel opens.
   c. Select a product that is configured in the runtime environment that you want to replicate. The Runtime Environments (RTEs) panel opens.

6. Generate the sample transport JCL:
   a. Type Z next to the runtime environment you want to transport and press Enter. The RTE Utility Menu panel is displayed.
   b. Select 5 (Generate sample transport JCL) and press Enter.
      This action causes several sample transport jobs to be generated in the &shilev.&rte.RKANSAMU library. For more information about these jobs, see “Step 4. Generate sample transport JCL” on page 188.

7. Use sample job XDFDMP03 on the first image to dump the runtime environment configuration batch jobs.

8. Use sample job XDFRST03 on the target image to restore the batch jobs on the target image.

9. On the target image, submit the batch jobs in the order listed in the Jobs Sorted By Generation Sequence section of the batch mode job report.
   You can submit each job manually or run a product-provided CLIST to submit the Configuration Tool jobs on the target image. To use the CLIST to submit the jobs:
   a. Verify that the target libraries that were installed by SMP/E are available on the image where the CLIST will be executed.
   b. Edit the member named SUB#ssss in the shilev.INSTJOBS data set, where ssss identifies the JCL suffix for the new runtime environment.
   c. Run the CLIST to submit the Configuration Tool jobs that will create the runtime environment.
   d. Exit the shilev.INSTJOBS data set so that CICATB can run.

Results

You have successfully created a new runtime environment on the target image.

What to do next

Perform any manual configuration steps on the target image, such as:

• Copying procedures to PROCLIB
• Copying VTAM definitions to VTAMLST
• APF-authorizing libraries

See “Step 6. Complete the configuration of a new runtime environment” on page 196 for more information.
Scenario D: Transporting runtime environment batch mode parameters from one z/OS image to another

About this task

Follow these steps to transport runtime environment batch mode parameters from one z/OS image to another.

Procedure

1. Start the Configuration Tool on your first image using this command:
   EX 'shilev.INSTLIB'
2. Navigate to the Runtime Environments (RTEs) panel:
   a. On the Main Menu panel select option 3 (Configure products). The Configure Products panel is displayed.
   b. Select option 1 (Select product to configure).
      The Product Selection Menu panel opens.
   c. Select a product that is configured in the runtime environment that you want to replicate. The Runtime Environments (RTEs) panel opens.
3. Generate the sample transport JCL:
   a. Type Z next to the runtime environment you want to transport and press Enter. The RTE Utility Menu panel is displayed.
   b. Select 5 (Generate sample transport JCL) and press Enter.
      This action causes several sample transport jobs to be generated in the &shilev.&rte.RKANSAMU library. For more information about these jobs, see “Step 4. Generate sample transport JCL” on page 188.
4. On the master image, use sample job XDFDMP03 to dump the batch jobs.
5. On the target image, use XDFRST03 to restore the batch jobs.
6. On the target image, create the Configuration Tool batch mode job:
   a. Start the Configuration Tool by entering the following command in a TSO command panel:
      EXEC 'shilev.INSTLIB'
   b. From the Main Menu, select Option 1 (Set up work environment) and press Enter.
      The Set up work environment panel is displayed.
   c. Select option 3 (Create batch mode job).
   d. Wait until you see the CICATB JOB CREATED message in the upper-right corner of the panel.
7. Exit the Configuration Tool.
8. Perform a scan on your runtime environment parameters:
   a. Edit CICATB, updating the BATCHLIB and BATCHMEM parameters as necessary, and setting the SUBMIT parameter to SCAN.
   b. Submit the CICATB job to scan your runtime environment parameters.
   c. Verify that the job completes successfully, review the parameter report, correct any errors in the parameter member, and repeat the scan until a clean report is generated.
9. Create the runtime environment on the target image:
   a. Edit CICATB again, setting the SUBMIT parameter to YES. This option submits the runtime environment configuration jobs that allocate and populate runtime libraries.
   b. Submit the CICATB job to create the runtime environment.
   c. Verify that the job completes successfully.
What to do next

Perform any manual configuration steps on the target image, such as:
- Copying procedures to PROCLIB
- Copying VTAM definitions to VTAMLST
- APF-authorizing libraries

See “Step 6. Complete the configuration of a new runtime environment” for more information.

Step 6. Complete the configuration of a new runtime environment

Batch mode configuration requires manual steps after the configuration is complete. These steps are listed in a Complete the Configuration file, which can be accessed only from within the Configuration Tool. This file is created specifically for each configuration and its content depends on the configured components.

About this task

Follow this procedure to generate the file:

Procedure

1. From the Configuration Tool Main Menu (Figure 25 on page 105), enter 3 (Configure products).
2. On the Product Selection Menu, select any product.
3. On the Runtime Environments (RTEs) panel, enter Z (Utilities) next to the runtime environment in which you want to complete product configuration.
4. On the RTE Utility Menu (Figure 50 on page 233), enter 9 (Create Consolidated Complete the Configuration instructions).
   This option creates JCL containing a consolidated list of Complete the Configuration instructions for all products configured in the runtime environment. The list is stored in the shlev.INSTJOBS data set, in a member named $DFI@ssss$ (where $ssss$ is the unique JCL suffix for the runtime environment).
5. Perform the tasks listed in the Complete the Configuration job. The list is generated dynamically and tailored specifically to your configuration.

The most common tasks for configuring a monitoring server are covered in Chapter 6, “Completing the configuration,” on page 129. For detailed information about completing the configuration of monitoring agents, see the configuration documentation for each monitoring agent in the Tivoli Monitoring and OMEGAMON XE Information Center at http://publib.boulder.ibm.com/infocenter/tivihelp/v15r1/index.jsp?toc=

Tips:
- You do not have to configure the monitoring server in a runtime environment more than once, even though the monitoring server shows up on the Product Component Selection Menu for every monitoring agent.
- The Complete the Configuration list for every monitoring agent on z/OS includes the task of registering the monitoring agent with the monitoring server in its runtime environment. See IBM Tivoli Management Services on z/OS: Configuring the Tivoli Enterprise Monitoring Server on z/OS.

Results

See the “Complete the configuration” section of the configuration guide for each product installed in the runtime environment for more information about required post-Configuration Tool configuration steps specific to that product.
Part 4. Appendixes
Appendix A. Enabling system variable support

System variables are elements that allow systems to share parameter definitions while retaining unique values in those definitions. System variables act like variables in a program; they can take on different values, based on the input to the program. When you specify a system variable in a shared parameter definition, the system variable acts as a placeholder. Each system that shares the definition replaces the system variable with a unique value during initialization. With system variables, the software becomes z/OS system-independent. The software can then be ported and started on any z/OS system without extensive reconfiguration.

About this task

If you use system variables, the components inherit the system values of the system on which they are started (the host z/OS system). These system-specific values are then automatically loaded into temporary data sets that exist only while the component runs. The result is that the software runs correctly by using the system-specific parameter values for the host z/OS system.

Tips:
• You cannot use system variables in the runtime environment of a high-availability hub. See IBM Tivoli Management Services on z/OS: Configuring the Tivoli Enterprise Monitoring Server on z/OS for details.
• Product started tasks contain a preprocessing step that resolves all system variable specifications in the product parameter members.
• If you enable system variables, product parameter members contain many variables whose values are resolved during startup of the started task.

See the z/OS MVS Initialization and Tuning Reference (SA22-7592) for basic information on system variables.

Both the PARMGEN configuration method and the Configuration Tool method support system variables.

The topics in this appendix provides instructions for enabling system variable support and using it to run your installed products on any z/OS system.

• For the PARMGEN configuration method, follow the instructions in Chapter 4, “Configuring products using the PARMGEN method,” on page 63.
• For the Configuration Tool (ICAT) method, follow the instructions in Chapter 5, “Configuring products with the Configuration Tool,” on page 103.

Using the PARMGEN Workflow user interface to set up system variables

Complete these basic steps to enable and use system variable support within the runtime environment by using the PARMGEN Workflow user interface.

For more information about using system variables in your PARMGEN configuration, see the scenarios described in the IBM Tivoli OMEGAMON XE and IBM Tivoli Management Services on z/OS: PARMGEN Reference.

Prerequisites

A number of PARMGEN configuration profile parameters support symbolics as their parameter values. Care should be used when specifying user-defined symbols.
The following PARMGEN profile parameters support symbolics. The parameter values are referenced in the STC PROC symbols.

**RTE_NAME**
- Used as the STC PROC SYS= symbol parameter.

**RTE_HILEV**
- Used as the STC PROC RHILEV= symbol parameter.

**RTE_VSAM_HILEV**
- Used as the STC PROC RVHILEV= symbol parameter.

**RTE_X_HILEV_SHARING**
- Used as part of the STC PROC BASEHLEV= symbol parameter.

**RTE_SHARE**
- Used as part of the STC PROC BASEHLEV= symbol parameter.

Any PARMGEN profile parameters that have VTAM APPLs (*pp_* _VTAM_APPL_*) are applicable to &stc_prefix.M2RC and &stc_prefix.TOM, as well as the STC PROC APPL= symbol parameter.

### Additional symbolics for OMEGAMON XE for IMS on z/OS sites

OMEGAMON XE for IMS on z/OS supports additional parameters that can use symbolics.

**KI2_I1nn_CLASSIC_IMSID**
- Used in the OMEGAMON for IMS (3270) STC PROC IMSID= symbol parameter.

**KI2_I1nn_CLASSIC_MPREFIX**
- Used in the OMEGAMON for IMS (3270) STC PROC MPREFIX= symbol parameter.

**Note:** Exercise caution when specifying user-defined symbols (as opposed to static symbols) for any of these configuration profile parameters because the JCL rules that govern the use of system symbols state that most system symbols cannot be used in batch JCL. In started-task procedures, you will generate a startup JCL error if you issue the /STARTstc command, as user symbols (SYSCLONE, for example) referenced as part of the STC PROC statement are not recognized by the system (unlike SYSNAME or SYSCON). Because these are system-wide defined static symbols).

Use one of the following ways to work around the JCL rule:
1. By overriding the value of these symbols by using the startup command
2. By using the PARMLIB sample composite STC startup member (CANSAPF by default) and then defining your own PROC symbol overrides

For VTAM_APPL-related parameters, no exceptions are allowed to the JCL rule. You must not use user-defined symbols or both the started tasks and the VTAM major node members will not work.

### Enable system variable support

Support for system variables must be enabled in each runtime environment.

In “Step 8. Customize PARMGEN configuration profiles” on page 84, select option 1 to produce the list of parameters for this runtime environment. To enable system variable support, you must change the value of the RTE_SYSV_SYSVAR_FLAG parameter to Y as part of your customization of the parameter list. Optionally, if you are configuring a runtime environment that will use and resolve LPAR system variables, also edit the value of the RTE_X_SYSV_OVERRIDE_SYMBOLS parameter to Y. Editing this parameter causes LPAR-specific values specified in parameters (such as system names) to be resolved correctly in the runtime members.

You can then customize the other parameters within the member to use the system variables and user-defined variables. For example, you can modify the runtime environment name by using a system variable:
You can also customize parameters to employ user-defined symbols and variables. In the following example, the USS directory used can be represented by a variable:

```
RTE_USS_RTEDIR "&RTE_USS_DIR."
```

On completion of the parameter customization, save the changes and return to the Customize PARMGEN Configuration Profile Members panel.

**Specify user-defined symbols and variables**

On the Customize PARMGEN Configuration Profile Members (KCIP@PG6) panel, select option 3 to customize the System Variables profile to define all the user-defined symbols added in the previous step. Figure 40 shows an example of the member where the resolved symbols are defined.

Figure 40. Pre-defined and user-defined symbolics

On completion of the parameter customization, save the changes and return to CUSTOMIZE PARMGEN CONFIGURATION PROFILE MEMBERS panel. You can continue configuring the runtime environment as described in Chapter 4, “Configuring products using the PARMGEN method,” on page 63.
**Using the Configuration Tool (ICAT) to set up system variables**

These are the basic steps of enabling and using system variable support with the Configuration Tool:

1. Define and configure a runtime environment. During runtime environment configuration, specify the values to enable system variable support (Use z/OS system variables? on the second Add or Update Runtime Environment panel: see “ Enable system variable support”).

   **Note:** Multiple runtime environments might be required depending on the runtime environment structure (full, sharing, base, or SMP/E), the type of monitoring server (hub or remote), and variations in product mixtures.

2. Create the system variable parameter member.
   After configuring a runtime environment, you can create the system variable parameter member. Each runtime environment contains one user system variable parameter member named RKANPARU(rte), which contains all system-specific values.

   See “Create the system variable parameter member” on page 203 for information on creating the rte system variable parameter member in the rhilev.rte.RKANPARU library.

3. Create the VTAM major node rename job.
   After configuring a runtime environment, you can create the VTAM major node rename job. This job creates VTAM major nodes on remote systems with names that are resolved from your system variable specification.

   See “Create the VTAM major node rename job” on page 204 for information on creating a VTAM major node rename job.

4. Create the runtime environment transport job by using the RTE Utility option Generate Sample Transport JCL.

5. Copy the runtime environment to a remote system by using the runtime environment transport job (or any other copy utility).

6. After the copy completes, edit the system variable parameter member RKANPARU (rte). If necessary, set values for components running on other systems. For example, set values for a hub monitoring server running on a different LPAR.

7. Perform other remote system setup tasks as required.
   - Copy the new started tasks to your system procedure library. These started tasks have been enabled for system variables.
   - If you are using VTAM system variable support, you must run the VTAM major node rename job. This job creates new major nodes that are named according to your system variable specifications. After the new nodes are created, copy them to SYSL.VTAMLST.
   - VSAM file allocation and seeding are required on every system.

8. Start the components.

**Enable system variable support**

You enable system variable support from the Add Runtime Environment panel. In an existing runtime environment, use the Update (U) command to enable system variable support.

Complete the following steps to enable system variable support.

1. From the Configuration Tool Main Menu (shown in Figure 25 on page 105), enter 3 (Configure products).
2. On the Configure Products panel, enter 1 (Select product to configure).
3. Select the product that you want to configure.
   The Runtime Environments (RTEs) panel is displayed.
4. If you are adding a new runtime environment, type A in the Action field and specify all other required information and press Enter. If you are updating a runtime environment, type U in the Action field and press Enter.
   The Add Runtime Environment or the Update Runtime Environment panel is displayed.
5. Specify any required values or accept the existing ones and press Enter.
   A second Add or Update Runtime Environment panel is displayed.
6. Specify the following values to enable system variable support:
   **Use z/OS system variables?**
   Specify Y to enable support. The default is N.
   **RTE name specification**
   Specify the system variable name by which the runtime environment is identified in the SYS1.PARMLIB LPAR system definition member. The default is SYSNAME. This value becomes the value of the SYS parameter in all started tasks (for example, SYS=‘SYSNAME’).
   **Note:** Resolved system variable values cannot exceed the length of the variable name (maximum length of 8 characters).
   
   **Important:** If you change the status of system variable support in an existing runtime environment (on to off or vice versa), you must reconfigure all products in that runtime environment. This includes specifying VTAM values and creating runtime members.
7. (Optional) Specify the following values on the second Add Runtime Environment panel to enable other functions:
   **RTE base alias specification**
   If this runtime environment is sharing with a base runtime environment, specify an optional system variable specification for the base runtime environment. This value is inserted into the base runtime environment library references in all started tasks. The resolved name must be a valid library name qualifier. This field is commonly used to switch between base runtime environments at different maintenance levels. Use the runtime environment base alias to obtain the following benefits:
   • An easy way to switch runtime environment bases.
   • An alternative way to refer to an existing base.
   **Note:** A label of n/a might be displayed next to this field if the current runtime environment is not sharing with a base runtime environment.
   **Applid prefix specification**
   Specify the VTAM applid prefix that contains system variables. Be sure to place a period after the last symbol in the specification. The resolved prefix can be a maximum of four characters. The default is KSYSCLONE.
   **Use VTAM model applids?**
   If you want to use model applids (wildcards), specify Y. Model applids generate VTAM nodes that contain applids with wildcard suffixes wherever possible. Then you can use any applids that match the pattern within the VTAM node. The default is N.
8. When you have finished specifying the values to enable system variable support, press F3 until you return to the Main Menu.

**Create the system variable parameter member**
You enable system variable support from the Add Runtime Environment panel. In an existing runtime environment, use the Update (U) command to enable system variable support.
After configuring a runtime environment, you can create the system variable parameter member.

Each runtime environment contains one user system variable parameter member named RKANPARU(rte).
All system-specific values are contained in this member.

Complete the following steps to create the system variable parameter member, rte, in the rhilev.rte.RKANPARU library.

1. From the Configuration Tool Main Menu (shown in Figure 25 on page 105), enter 3 (Configure products).
2. On the Configure Products panel, enter 1 (Select product to configure).
3. Select the product that you want to configure.
The Runtime Environments (RTEs) panel is displayed.
4. Type Z (Utilities) next to the name of the runtime environment you have configured and press Enter.
5. On the RTE Utility Menu, select Create System Variable Parameter Member and press Enter.
The JCL that creates the system variable parameter member (CB#Vssss) job is displayed.

Important: The KCIDSV41 step in the CB#Vssss job requires exclusive enqueue on the runtime libraries allocated in each DDNAME. Make sure that the job has the access it requires.
6. Review the JCL and submit the job. Verify that the job completes successfully and that all return codes are zero.
7. Press F3 to return to the Runtime Environments (RTEs) panel.
8. Edit the rhilev.rte.RKANPARU(rte) parameter member. Follow the directions to ensure correct resolution of cross-system variables.

Create the VTAM major node rename job

After configuring a runtime environment, you can create the VTAM major node rename job. This job creates VTAM major nodes on remote systems with names that are resolved from your system variable specification.

Complete the following steps to create a VTAM major node rename job in the INSTJOBS library.

1. From the Configuration Tool Main Menu (shown in Figure 25 on page 105), enter 3 (Configure products).
2. On the Configure Products panel, enter 1 (Select product to configure).
3. Select the product that you want to configure.
The Runtime Environments (RTEs) panel is displayed.
4. Type Z (Utilities) next to the name of the runtime environment you have configured and press Enter.
5. On the RTE Utility Menu, select Create System Variable VTAM Major Node Rename Job, and then press Enter.
The JCL that renames the VTAM major node (CB#7ssss) job is displayed.
6. Review the JCL. Do not submit the job yet.

Note: You must submit this job on each remote system where the monitoring software is to run. The job is in the rhilev.rte.RKANSAMU library.
7. Press F3 to return to the Runtime Environments (RTEs) panel.

Create one VTAM major node for all products in the runtime environment

A single VTAM major node can contain all the VTAM applids for all of the TMS:Engine-based products you have configured in the runtime environment. This single major node is then used in place of the individual product major nodes.
If you choose to use a single VTAM major node, you must create it after all products have been configured in the runtime environment. After the node is created and copied to your system VTAM system library (SYS1.VTAMLST), you vary it active and then start all of the components (started tasks).

Complete the following steps to create a single VTAM major node in the rhilev.rte.RKANSAMU library.

1. From the Configuration Tool Main Menu (shown in Figure 25 on page 105), enter 3 (Configure products).
2. On the Configure Products panel, enter 1 (Select product to configure).
3. Select the product that you want to configure.
   The Runtime Environments (RTEs) panel is displayed.
4. Type Z (Utilities) next to the name of the runtime environment you have configured and press Enter.
5. On the RTE Utility Menu, select Create VTAM Major Node (one node for all products) and press Enter.
6. On the Create VTAM Major Node panel, type the name you want to use for the single node and press Enter.
   The JCL that creates the single node is displayed.
7. Review the JCL and submit the job. Verify that the job completed successfully and that all return codes are zero.
8. Press F3 to return to the Runtime Environments (RTEs) panel.
Appendix B. Configuration Tool deployment scenarios

This section contains a series of thumbnail scenarios that illustrate various methods for using the Configuration Tool to replicate a configuration of Tivoli Management Services components and monitoring agents across an enterprise. These scenarios assume that you are using system variables (IBM symbolics) and consistent naming conventions for data set names, started tasks, and VTAM applids on all your LPARs. For detailed configuration instructions, see Part 2, “Configuring components on z/OS,” on page 61. For information about using system variables, see Appendix A, “Enabling system variable support,” on page 199.

The following sections cover several enterprise-wide deployment scenarios:

- “Scenario 1: Standard replication method”
- “Scenario 2: Using common RKANPARU and RKANCMDU libraries” on page 209
- “Scenario 3: Using batch mode to replicate a runtime environment to another LPAR” on page 211
- “Scenario 4: Using batch auto-submit to replicate a runtime environment to another LPAR” on page 212
- “Scenario 5: Using batch mode to replicate a runtime environment on the same LPAR” on page 213
- “Scenario 6: Using interactive copy to replicate a runtime environment” on page 215
- “Scenario 7: Copying configuration values from one instance to another of the Configuration Tool” on page 215
- “Scenario 8: Configuring a remote monitoring server and monitoring agents in batch mode” on page 216

Scenario 1: Standard replication method

This scenario describes a standard method for replicating a configuration of products across a z/OS enterprise. You install and configure all products on your main LPAR, and then copy items from the main LPAR to other LPARs throughout your enterprise.

Scenario 1 has the following advantages:

- You install and configure only one instance of the Configuration Tool.
- You configure a maximum of three unique runtime environments.
- Products use symbolic values defined on the LPAR they run on.
- Replication across the enterprise is quick.
- Runtime environments are consistent.
- You save storage space.

The disadvantage of this scenario is that you cannot use the Configuration Tool to update the runtime environments on other LPARs.

<table>
<thead>
<tr>
<th>CPC A</th>
<th>CPC B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shared DASD in CPC A</td>
<td>Shared DASD in CPC B</td>
</tr>
<tr>
<td>1 Configuration Tool and SMP/E environment</td>
<td>1 copy of base runtime environment from LPAR A1</td>
</tr>
<tr>
<td>1 base runtime environment</td>
<td>3 copies of sharing runtime environment A3_remote on LPAR A4</td>
</tr>
<tr>
<td>2 sharing runtime environments: A1_hub and A3_remote</td>
<td></td>
</tr>
<tr>
<td>1 copy of sharing runtime environment A3_remote on LPAR A4</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>LPAR A1 (main)</th>
<th>LPAR A3</th>
<th>LPAR A4</th>
<th>LPAR B1 (target)</th>
<th>LPAR B4</th>
<th>LPAR B7</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

© Copyright IBM Corp. 1994, 2013
To use scenario 1, perform the following steps:

1. Install all your products on central processor complex (CPC) A, LPAR A1 (main).
   This sets up one Configuration Tool and one SMP/E environment to support all LPARs. See the program directories for your products for detailed installation instructions.

2. Create a base runtime environment on LPAR A1 (main).

3. Create a sharing-with-base runtime environment on LPAR A1 (main), to contain the hub monitoring server and a set of monitoring agents. Be sure to enable system variables. This runtime environment becomes your template for hub sharing-with-base runtime environments on other LPARs.

4. Configure the hub monitoring server and monitoring agents.

5. Create a second sharing-with-base runtime environment on LPAR A3. Be sure to enable system variables. This runtime environment becomes your template for remote sharing-with-base runtime environments on other LPARs.

6. In this runtime environment, configure a remote monitoring server to report to the hub in LPAR A1, and a set of monitoring agents to report to the remote monitoring server.

7. Create the system variable parameter member. See “Create the system variable parameter member” on page 203.

8. (Optional) Create a single VTAM major node. See “Create one VTAM major node for all products in the runtime environment” on page 204.

9. Copy the base runtime libraries from LPAR A1 (main) to LPAR B1 (target), by using any copy-and-rename method or by using the runtime environment transport batch jobs provided by the Configuration Tool. Keep the same runtime library names, if possible. See “Step 4. Generate sample transport JCL” on page 188.

10. Copy and rename the remote sharing-with-base runtime libraries from LPAR A3 to LPAR B1, by using any copy-and-rename method or by using the runtime environment transport batch jobs provided by the Configuration Tool. As the files are copied, rename them to match the LPAR name.

11. In each rhilev.rte.RKANPARU library, create a new system variable member that matches the name of the LPAR, and copy into it the contents of the system variable member you created earlier.
    In the new member, change LPAR-specific values as required.

12. Repeat the preceding three steps for all other LPARs in your enterprise.

13. Complete the configuration on each system. At a minimum you must complete these steps:
a. Copy the started tasks and VTAM major nodes to the system libraries.
b. APF-authorize the data sets on the appropriate LPARs.

14. Start each monitoring server and monitoring agent.

Scenario 2: Using common RKANPARU and RKANCMDU libraries

This scenario describes another method for replicating a configuration of products across your z/OS enterprise. In the scenario, you create a new common RKANPARU library and a new common RKANCMDU library to contain the configuration values required for runtime environments in all of your LPARs. You install and configure all products on your main LPAR, and then copy items from the main LPAR to other LPARs throughout your enterprise.

Scenario 2 has the following advantages:
• You use one common set of RKANPARU and RKANCMDU libraries for all LPARs.
• You install and configure only one instance of the Configuration Tool.
• You configure a maximum of three unique runtime environments.
• Runtime environments are copied to other LPARs.
• Products use symbolic values defined on the LPAR they run on.
• Replication across the enterprise is quick.
• Runtime environments are consistent.
• You save storage space.

The disadvantage of this scenario is that you cannot use the Configuration Tool to update the runtime environments on other LPARs.

<table>
<thead>
<tr>
<th>CPC A</th>
<th>CPC B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shared DASD in CPC A</td>
<td>Shared DASD in CPC B</td>
</tr>
<tr>
<td>1 Configuration Tool and 1 SMP/E environment</td>
<td>1 copy of base runtime environment from LPAR A1</td>
</tr>
<tr>
<td>1 base runtime environment</td>
<td>3 copies of sharing runtime environment A3_remote</td>
</tr>
<tr>
<td>2 sharing runtime environments: A1_hub and A3_remote</td>
<td></td>
</tr>
<tr>
<td>1 copy of sharing runtime environment A3_remote on LPAR A4</td>
<td></td>
</tr>
<tr>
<td>1 common RKANPARU data set</td>
<td>1 common RKANPARU data set</td>
</tr>
<tr>
<td>1 common RKANCMDU data set</td>
<td>1 common RKANCMDU data set</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>LPAR A1 (main)</th>
<th>LPAR A3</th>
<th>LPAR A4</th>
<th>LPAR B1 (target)</th>
<th>LPAR B4</th>
<th>LPAR B7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration Tool, target libraries that were installed by SMP/E</td>
<td>Copy of sharing runtime environment A3_remote, renamed to A4</td>
<td>Copy of sharing runtime environment A3_remote, renamed to B1</td>
<td>Copy of sharing runtime environment A3_remote, renamed to B1</td>
<td>Copy of sharing runtime environment A3_remote, renamed to B4</td>
<td>Copy of sharing runtime environment A3_remote, renamed to B7</td>
</tr>
</tbody>
</table>
To use scenario 2, perform the following steps:

1. Install all your products on CPC A, LPAR A1 (main).
   This sets up one Configuration Tool and one SMP/E environment to support all LPARs. See the program directories for your products for detailed installation instructions.

2. Create a base runtime environment on LPAR A1 (main).

3. Create a sharing-with-base runtime environment on LPAR A1 (main), to contain the hub monitoring server and a set of monitoring agents. Be sure to enable system variables.

4. Configure the hub monitoring server and monitoring agents in the sharing-with-base runtime environment. This runtime environment becomes your template for hub sharing-with-base runtime environments on other LPARs.

5. Create a second sharing-with-base runtime environment on LPAR A3. Be sure to enable system variables. This runtime environment becomes your template for remote sharing-with-base runtime environments on other LPARs.

6. In this runtime environment, configure a remote monitoring server to report to the hub in LPAR A1, and a set of monitoring agents to report to the remote monitoring server.

7. Create the system variable parameter member. See "Create the system variable parameter member" on page 203.

8. Create common RKANPARU and RKANCMDU data sets, and copy the corresponding template runtime environment data sets into these common data sets.

9. In the common RKANPARU, create additional system variable members as necessary.
   The names of these members must match the other LPAR names. For each member, change LPAR-specific values as necessary. Review cross-system values for components on different LPARs.

10. Edit RKANSAMU started tasks, changing all occurrences of the RKANPARU and RKANCMDU data set names to the common data set names.
    For example:
    `C 'rhilev.rte.RKANPARU' 'common.RKANPARU' all`
    `C 'rhilev.rte.RKANCMDU' 'common.RKANCMDU' all`

11. (Optional) Create a single VTAM major node. See "Create one VTAM major node for all products in the runtime environment" on page 204.

12. Copy the following data sets to LPAR B1 (target), by using any copy-and-rename method or by using the runtime environment transport batch jobs provided by the Configuration Tool:
    - Base runtime environment data sets
    - Common RKANPARU and RKANCMDU data sets
    Keep the same data set names, if possible.

13. Copy and rename the remote sharing-with-base runtime libraries from LPAR A1 to LPAR B1, by using any copy-and-rename method or by using the runtime environment transport batch jobs provided by the Configuration Tool. As the files are copied, rename them to match the LPAR name.

14. Repeat the preceding two steps for all other LPARs in your enterprise.

15. Complete the configuration on each system. At a minimum you must complete these steps:
    a. Copy the started tasks and VTAM major nodes to the system libraries.
    b. APF-authorize the data sets on the appropriate LPARs.

16. Start each monitoring server and monitoring agent.
Scenario 3: Using batch mode to replicate a runtime environment to another LPAR

This scenario describes a method for copying a runtime environment to another LPAR, using batch mode in the Configuration Tool.

Scenario 3 has the following advantages:
• You have less work to do in interactive mode in the Configuration Tool.
• You create each runtime environment after the first with a single batch job.
• Detailed reports help you troubleshoot and edit the batch job.

The only disadvantage is that you must install and set up an instance of the Configuration Tool on each LPAR.

<table>
<thead>
<tr>
<th>CPC A</th>
<th>CPC B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shared DASD in CPC A</td>
<td>Shared DASD in CPC B</td>
</tr>
<tr>
<td>1 Configuration Tool</td>
<td>1 Configuration Tool</td>
</tr>
<tr>
<td>1 SMP/E environment</td>
<td>1 copy of target libraries from SMP/E</td>
</tr>
<tr>
<td>1 base runtime environment</td>
<td>1 copy of base runtime environment</td>
</tr>
<tr>
<td>3 unique sharing runtime environments</td>
<td>3 unique sharing runtime environments</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>LPAR A1 (main)</th>
<th>LPAR A3</th>
<th>LPAR A4</th>
<th>LPAR B1 (target)</th>
<th>LPAR B4</th>
<th>LPAR B7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration Tool, target libraries that were installed by SMP/E</td>
<td>Sharing runtime environment A1, created in interactive mode</td>
<td>Sharing runtime environment A4, created in batch mode</td>
<td>Sharing runtime environment A1, created in interactive mode</td>
<td>Sharing runtime environment A4, created in batch mode</td>
<td>Sharing runtime environment B7, created in batch mode</td>
</tr>
<tr>
<td>Base runtime environment</td>
<td>Sharing runtime environment A3, created in batch mode</td>
<td>Sharing runtime environment A4, created in batch mode</td>
<td>Base runtime environment copied from A1 and renamed to B1</td>
<td>Sharing runtime environment B4, created in batch mode</td>
<td>Sharing runtime environment B7, created in batch mode</td>
</tr>
</tbody>
</table>

To use scenario 3, perform the following steps:
1. Install all your products on CPC A, LPAR A1 (main). See the program directories for your products for detailed installation instructions.
2. Copy the Configuration Tool libraries and the target libraries that were installed by SMP/E to CPC B, by using any copy-and-rename method or by using the runtime environment transport batch jobs provided by the Configuration Tool.
3. Create a base runtime environment on LPAR A1 (main).
   See “Step 2. Create and clone a batch mode parameter member” on page 183 for detailed procedure.
6. Create a new batch parameter member for each of the other LPARs (A3, A4, B1, B4, and B7, by copying the contents of the sharing-with-base runtime environment A1 parameter member and renaming it to the name of the target runtime environment.
7. In each new batch parameter member, review and edit the values.
8. Submit the Configuration Tool batch job on CPC A for each LPAR in CPC A (A3 and A4). This creates the required runtime environments for CPC A.
9. Copy the batch parameter members for each LPAR in CPC B (B1, B4, B7) to the Configuration Tool on CPC B.
10. Submit the Configuration Tool batch job on CPC B for each LPAR in CPC B (B1, B4, B7). This creates the required runtime environments for CPC B.
11. Copy the base runtime environment A1 data sets to LPAR B1. Keep the same data set names, if possible.
12. Complete the configuration on each system. At a minimum you must complete these steps:
   a. Copy the started tasks and VTAM major nodes to the system libraries.
   b. APF-authorize the data sets on the appropriate LPARs.
13. Start each monitoring server and monitoring agent.

Scenario 4: Using batch auto-submit to replicate a runtime environment to another LPAR

This scenario describes a method for copying a runtime environment to another LPAR by using the auto-submit feature of batch mode processing in the Configuration Tool. This scenario is similar to scenario 3, except that the new runtime environments are created by JCL that is built on the main LPAR, using Configuration Tool batch mode processing. You copy the JCL to the target LPAR and run a product-provided CLIST to submit the necessary jobs.

Scenario 4 has the following advantages:
• You install and configure only one instance of the Configuration Tool.
• The Configuration Tool contains all runtime environment configuration values.
• All JCL is built on the main LPAR.
• You have less work to do in interactive mode in the Configuration Tool.
• You create each runtime environment after the first with a single batch job.
• Detailed reports help you troubleshoot and edit the batch job.

This scenario has no disadvantages.

<table>
<thead>
<tr>
<th>CPC A</th>
<th>CPC B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shared DASD in CPC A</td>
<td>Shared DASD in CPC B</td>
</tr>
<tr>
<td>1 Configuration Tool</td>
<td>1 copy of target libraries from SMP/E</td>
</tr>
<tr>
<td>1 SMP/E environment</td>
<td></td>
</tr>
<tr>
<td>1 base runtime environment</td>
<td>1 copy of base runtime environment</td>
</tr>
<tr>
<td>3 unique sharing runtime environments</td>
<td>3 unique sharing runtime environments</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>LPAR A1 (main)</th>
<th>LPAR A3</th>
<th>LPAR A4</th>
<th>LPAR B1 (target)</th>
<th>LPAR B4</th>
<th>LPAR B7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration Tool, target libraries that were installed by SMP/E</td>
<td>Target libraries copied from A1</td>
<td>Base runtime environment copied from A1 and renamed to B1</td>
<td>Sharing runtime environment B1, created in batch mode</td>
<td>Sharing runtime environment B4, created in batch mode</td>
<td>Sharing runtime environment B7, created in batch mode</td>
</tr>
<tr>
<td>Base runtime environment</td>
<td>Sharing runtime environment A3, created in batch mode</td>
<td>Sharing runtime environment A4, created in batch mode</td>
<td>Sharing runtime environment A1, created in interactive mode</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
To use scenario 4, perform the following steps:

1. Install all your products on CPC A, LPAR A1 (main). See the program directories for your products for detailed installation instructions.

2. Copy the target libraries that were installed by SMP/E to CPC B, by using any copy-and-rename method or by using the runtime environment transport batch jobs provided by the Configuration Tool.

3. Create a base runtime environment on LPAR A1 (main).


   See “Step 2. Create and clone a batch mode parameter member” on page 183 for detailed procedure.

6. Create a new batch parameter member for each of the other LPARs (A3, A4, B1, B4, and B7, by copying the contents of the sharing-with-base runtime environment A1 parameter member and renaming it to the name of the target runtime environment.

7. In each new batch parameter member, review and edit the values.

8. On LPAR A1, submit the Configuration Tool batch jobs for all LPARs in CPC A (A3 and A4) and for all LPARs in CPC B (B1, B4, B7). Set the SUBMIT parameter to YES in the batch job options for LPARs A3 and A4, and to NO for all CPC B LPARs.

   These batch jobs create the required runtime environments for CPC A and generate the JCL necessary for creating the runtime environments on CPC B.

9. Copy the JCL members from shilev.INSTJOBS to each LPAR in CPC B.

   The names of these members end in the 4-character suffix that you specified in the batch parameter member.

10. On CPC B, run the product-provided CLIST named SUB#ssss to submit the batch jobs required to create runtime environments B1, B4, and B7.

11. Copy the base runtime environment A1 data sets to LPAR B1. Keep the same data set names, if possible.

12. Complete the configuration on each system. At a minimum you must complete these steps:

   a. Copy the started tasks and VTAM major nodes to the system libraries.

   b. APF-authorize the data sets on the appropriate LPARs.

13. Start each monitoring server and monitoring agent.

---

**Scenario 5: Using batch mode to replicate a runtime environment on the same LPAR**

This scenario describes a method for replicating a runtime environment on the same LPAR, using batch mode processing in the Configuration Tool. You install and configure all products on your main LPAR, and then copy the runtime environments to the appropriate LPARs.

Scenario 5 has the following advantages:

- You install and configure only one instance of the Configuration Tool.
- You have less work to do in interactive mode in the Configuration Tool.
- You create each runtime environment after the first with a single batch job.
- Detailed reports help you troubleshoot and edit the batch job.
• Making copies of a runtime environment without moving to a different LPAR is the easiest way to replicate a runtime environment in a large enterprise.

The disadvantage is that duplicated runtime environments temporarily use more storage space.

<table>
<thead>
<tr>
<th>CPC A</th>
<th>CPC B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shared DASD in CPC A</td>
<td>Shared DASD in CPC B</td>
</tr>
<tr>
<td>1 Configuration Tool</td>
<td>1 copy of base runtime environment</td>
</tr>
<tr>
<td>1 SMP/E environment</td>
<td>3 copies of sharing runtime environments</td>
</tr>
<tr>
<td>1 base runtime environment</td>
<td></td>
</tr>
<tr>
<td>3 unique sharing runtime environments</td>
<td></td>
</tr>
</tbody>
</table>

**LPAR A1 (main)**
- Configuration Tool, target libraries that were installed by SMP/E
- Base runtime environment
- Sharing runtime environment A1, created in interactive mode
- Sharing runtime environments A3, A4, B1, B4, and B7, created in batch mode

**LPAR A3**
- Sharing runtime environment A3, copied from A1

**LPAR A4**
- Sharing runtime environment A4, copied from A1

**LPAR A1 (target)**
- Base runtime environment copied from A1 and renamed to B1
- Sharing runtime environment B1, copied from A1
- Sharing runtime environment B4, copied from A1
- Sharing runtime environment B7, copied from A1

**LPAR B4**
- Sharing runtime environment B4, copied from A1

**LPAR B7**
- Sharing runtime environment B7, copied from A1

To use scenario 5, perform the following steps:

1. Install all your products on CPC A, LPAR A1 (main).
   - This sets up one Configuration Tool and one SMP/E environment to support all LPARs. See the program directories for your products for detailed installation instructions.
2. Create a base runtime environment on LPAR A1.
   - See “Step 2. Create and clone a batch mode parameter member” on page 183 for detailed procedure.
5. Create a new batch parameter member for each of the other LPARs (A3, A4, B1, B4, and B7, by copying the contents of the sharing-with-base runtime environment A1 parameter member and renaming it to the name of the target runtime environment.
6. In each new batch parameter member, review and edit the values.
7. Submit the Configuration Tool batch job for each LPAR, to create the required runtime environments.
8. Copy the base runtime environment A1 data sets to LPAR B1, by using any copy-and-rename method or by using the runtime environment transport batch jobs provided by the Configuration Tool. Keep the same data set names, if possible.
9. Copy the appropriate sharing-with-base runtime environment to each LPAR, by using any copy-and-rename method or by using the runtime environment transport batch jobs provided by the Configuration Tool. As the files are copied, rename them to match the destination LPAR name.
10. Complete the configuration on each system. At a minimum you must complete these steps:
   a. Copy the started tasks and VTAM major nodes to the system libraries.
   b. APF-authorize the data sets on the appropriate LPARs.
11. Start each monitoring server and monitoring agent.

**Scenario 6: Using interactive copy to replicate a runtime environment**

This scenario describes an interactive method for creating a runtime environment based on values specified in an existing runtime environment.

The advantage of this method is that it saves duplication of effort in interactive mode. The disadvantage is that you must navigate through all the configuration panels in the Configuration Tool to review and edit values for the new runtime environment and to submit the configuration jobs.

<table>
<thead>
<tr>
<th>LPAR A1 (main)</th>
<th>LPAR A3 (target)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration Tool, target libraries that were installed by SMP/E</td>
<td>Runtime environment A3</td>
</tr>
<tr>
<td>Runtime environment A1</td>
<td>Runtime environment A3</td>
</tr>
</tbody>
</table>

To use scenario 6, perform the following steps:
2. Create runtime environment A3, specifying the name of runtime environment A1 in this field on the Add Runtime Environment panel:
   Copy configuration values from RTE =>
3. Complete all configuration steps and jobs for runtime environment A3.

**Scenario 7: Copying configuration values from one instance to another of the Configuration Tool**

This scenario describes a method for copying configuration values, one runtime environment at a time, from one instance to another of the Configuration Tool.

The advantage of this method is that it reduces duplication of effort in configuring products. The disadvantage is that you must install and set up more than one instance of the Configuration Tool.

<table>
<thead>
<tr>
<th>Configuration Tool #1</th>
<th>Configuration Tool #2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Batch parameter member for runtime environment A</td>
<td>Batch mode job using the parameter member from runtime environment A</td>
</tr>
<tr>
<td>ISPF tables storing the runtime environment values</td>
<td>ISPF tables storing the runtime environment values</td>
</tr>
</tbody>
</table>

To use scenario 7, perform the following steps:
1. In interactive mode in Configuration Tool #1, create and configure runtime environment A.
2. Create a batch parameter member for runtime environment A.
3. In Configuration Tool #2, run batch mode job CICATB, by using the batch parameter member from runtime environment A. Specify SUBMIT(NO) if you want to populate the Configuration Tool ISPF tables and generate JCL in the shilev.INSTJOBS data set without submitting the jobs to create and configure a second runtime environment.
Scenario 8: Configuring a remote monitoring server and monitoring agents in batch mode

When you use batch mode, you create new runtime environments by submitting batch jobs, rather than by configuring them panel by panel. You use the configuration values from an existing runtime environment as the model for the new runtime environment and modify the values as required to define the new environment.

This scenario shows how to use batch mode processing to create a new runtime environment that contains a remote z/OS monitoring server and monitoring agents.

In the scenario shown in Figure 41, an existing runtime environment (here called SYSA) is used as the model for the new runtime environment. The SYSA runtime environment has the following characteristics:

- It was created by using the Configuration Tool ISPF screens.
- It contains a hub monitoring server and several monitoring agents.
- It uses system variables.
- The sysplex proxy is defined to run in the hub. (The sysplex proxy is a component of OMEGAMON XE on z/OS, which is one of the monitoring agents in the runtime environment.)
- Several OMEGAMON XE products are configured in the runtime environment. All agents defined in the runtime environment run in stand-alone address spaces, with the exception of OMEGAMON XE on z/OS and OMEGAMON XE for Storage on z/OS, which must run in the monitoring server address space.

The new runtime environment created with batch mode (here called SYSB) has the following characteristics:
It contains a remote monitoring server, reporting to the hub in SYSA.

It uses system variables.

It has the same agents as SYSA, but the agents connect to the monitoring server in SYSB.

SYSB is in the same sysplex as SYSA.

Complete the following steps in order. For examples of other configuration scenarios that use batch mode processing, see Appendix B, “Configuration Tool deployment scenarios,” on page 207. See Chapter 8, “Using the Configuration Tool batch mode to replicate a configured environment,” on page 179 for more information on batch mode processing.

**Step 1. Create the batch mode job (CICATB)**

The CICATB job generates the configuration jobs required to build, configure and load the products configured in a runtime environment. CICATB can also be used to validate the configuration values used by the configuration jobs. CICATB does not exist by default. You generate it by using the Configuration Tool.

Complete the following steps to create the job:

1. Start the Configuration Tool by entering the following command in a TSO command panel:
   ```
   ===> EXEC 'shilev.INSTLIB'
   ```

2. From the Main Menu, select Option 1 (Set up work environment) and press Enter.
   The Set up work environment panel is displayed.

3. Select option 3 (Create batch mode job).

   **Tip:**

   The default location for the CICATB job is the *shilev*.INSTJOBS library. To see the complete path of the library the batch mode job was created in, press F1. A text box at the bottom of the panel shows the complete path. The text box is displayed only if F1 is pressed immediately after Create batch mode job is selected. Otherwise, the general help for the panel is displayed.

4. Wait until you see the CICATB JOB CREATED message in the upper-right corner of the panel.

The SYSTSIN DD statement in the JCL contains several parameters that you must customize before you submit the job.

**BATCHLIB**

The library that contains the batch runtime environment parameter member. The default is &shilev.INSTJOBS.

**BATCHMEM**

The member name of the batch runtime environment parameter member (the name must be the same as the name of the runtime environment you intend to create). You create this member in the following step.

**SUBMIT**

The type of processing. You specify different dispositions depending on the step you are performing and the requirements of the environment.
The options for this parameter are:

**YES**
Creates and submits the runtime environment configuration jobs that create the runtime environment, if the jobs do not already exist. If the jobs already exist, such as when you are reconfiguring a runtime environment, the jobs are regenerated but are not submitted for execution.

**RERUN**
Creates and submits the runtime environment configuration jobs, regardless of whether the jobs already exist.

**NO** Creates but does not submit the runtime environment configuration jobs.

**SCAN**
Validates the input parameter member and produces a report.

**SCAND**
Scans the input parameter member and produces a detailed data dictionary of all the runtime environment parameters.

**Important:**
When you create the CICATB batch job member in the `shilev.INSTJOBS` data set, the Configuration Tool also creates the batch jobcard ISPF table KCIITPIG1 in the `shilev.INSTDATA` data set. The contents of the KCIITPIG1 table are used as the standard jobcard in batch configuration jobs. If you modify the jobcard on the Specify JCL Options panel [Figure 27 on page 106] or if the ISPF environment changes in some other manner, you must recreate the CICATB batch job to refresh the KCIITPIG1 jobcard table before running CICATB.

**Step 2. Clone the runtime environment batch parameter member**
A batch mode parameter member contains configuration values for all the products in an existing runtime environment. The member created, or recreated, every time the runtime environment Load job is run. It can also be generated by using the “Create/Clone batch mode parameters” utility. To create a new runtime environment using batch processing, you clone a copy of the member for an existing runtime environment and modify the values as required for the target image. The name you give to the cloned member becomes the mid-level qualifier for the new runtime environment you create by using the member.

Take the following steps to clone a parameter member:
1. Start the Configuration Tool by entering the following command in a TSO command panel:
   ```
   ===> EXEC 'shilev.INSTLIB'
   ```
2. On the Main Menu panel, select option 3 (Configure products).
   The Configure Products panel is displayed (as shown in Figure 29 on page 110).
3. Select 1 (Select product to configure) and press Enter. The Product Selection Menu is displayed (as shown in Figure 30 on page 110).
4. Select any product to bring up the Runtime Environments (RTEs) panel [Figure 42 on page 219].
The existing runtime environment is named SYSA. It is a sharing runtime environment that uses the SMP/E libraries and contains a hub monitoring server.

5. Enter Z (Utilities) next to SYSA and press Enter to display the RTE Utility Menu (Figure 43).

6. Select option 1 (Create/Clone batch mode parameters) and press Enter. The Create/Clone Parameter Deck Menu is displayed (Figure 44).

---

**Figure 42. Runtime Environments (RTEs) panel**

The existing runtime environment is named SYSA. It is a sharing runtime environment that uses the SMP/E libraries and contains a hub monitoring server.

5. Enter Z (Utilities) next to SYSA and press Enter to display the RTE Utility Menu (Figure 43).

**Figure 43. RTE Utility Menu panel**

6. Select option 1 (Create/Clone batch mode parameters) and press Enter. The Create/Clone Parameter Deck Menu is displayed (Figure 44).

**Figure 44. Create/Clone Parameter Deck Menu panel**
7. Because this scenario involves changing a hub monitoring server in the model runtime environment into a remote monitoring server in the replicated runtime environment, select option 3 (Create batch mode parameter deck, with modifications) and press Enter. This option also allows you the ability to exclude products defined in the model environment from the target environment.

The Create/Clone Batch Mode Parameters panel is displayed (Figure 45).

```
 KCIPBATO ------ CREATE/CLONE BATCH MODE PARAMETERS / RTE: SYSA-------------
 COMMAND ===> 

 Create parameter deck member:
 Parameter library ==> TEST.V621.INSTJOBS
 Parms member name ==> SYSA
 Edit after create ==> Y

 Clone parameter deck member:
 Parameter library ==> TEST.V621.INSTJOBS
 Clone member name ==> SYSB
 Edit after create ==> 

 Enter=Next  F1=Help  F3=Back
```

**Figure 45. Create/Clone Batch Mode Parameters panel**

The first section of the panel, Create Parameter Deck Member, contains default values for the parameter member with values from the existing runtime environment. If a parameter deck does not already exist, the Configuration Tool will create it. By default, the name of the parameter member is the name of the runtime environment and it is created in the shilev.INSTJOBS library. If you want to examine the parameter member after it is created, leave Y as the value for **Edit after create**. Otherwise, type N.

For Clone Parameter Deck Member, specify appropriate values for the following fields:

**Parameter library**

The library where the clone parameter member is to be created. By default, the library is shilev.INSTJOBS. You can change the library name, but the library that you specify must already exist. If the library does not exist, the Configuration Tool does not create the parameter member and display the message "LIBRARY ERROR" in the upper right corner of the panel.

**Clone member name**

The name of the clone member. This name becomes the mid-level qualifier for the new runtime environment. For this scenario, the name is SYSB.

**Edit after create**

Specify Y to have the Configuration Tool open the member in ISPF edit mode so you can review or edit the values. If you want to edit the member later, specify N.

8. Press Enter.

If the input parameter deck member does not already exist, the following message is displayed:

```
The parameter deck member SYSA does not exist.
Press Enter to continue and create RTE parameters
or F3 to cancel.
```

Press Enter to create the parameter deck.

After parameter member for the model runtime environment is created, the following message is displayed:

```
BATCH PARMS CREATED
```

9. If necessary, press Enter again to clone the parameter deck.

The following message is displayed:
The cloned batch parameter deck includes a value for the RTE JCL suffix.

Every RTE should have a unique one to four character RTE_JCL_SUFF value.

The RTE_JCL_SUFF value for the original RTE
SYSA = SYSA

Specify a new unique RTE_JCL_SUFF for
RTE SYSB ==> ____

ENTER=Next F3=Abort Cloning

10. Type the JCL suffix for the new runtime environment and press Enter.

If you specified Y for **Edit after create**, the clone member is displayed.

**Step 3. Review and update the parameter member for the new runtime environment**

If you specified Y for **Edit after create**, the parameter member is already displayed in ISPF edit mode. If you did not specify Y, locate the member in the specified library and open it.

You see something similar to **Figure 46**

```
EDIT  TEST.V621.INSTJOBS(SYSB) - 01.00  Columns 00001 00072
Command ===> Scroll ===> PAGE
***** *************** Top of Data **********************
000001
000002 RTE$310 BEGIN *********** CONFIGURATION TOOL V310 ***********
000003 RTE_DESC "420 and 621 RUNTIME ENV SYSA HUB"
000004 RTE_TYP SHARING * FULL, SHARING or BASE *
000005
000006 ** If RTE_TYP is SHARING:
000007 RTE_SHARE SMP
000008 **
000009 ** RTE global defaults:
000010 RTE_HILEV TEST.V621
000011 RTE_VSAM_HILEV TEST.V621
000012
000013 ** (Opt) Only valid when RTE_TYP is BASE:
000014 **RTE_BASE_MIDLEV
000015
000016 RTE_VOL CANDL1
000017 RTE_VSAM_VOL CANDL1
000018 RTE_UNIT 3390
000019 RTE_SMS_STOR_CLAS
000020 RTE_SMS_VSAM_STOR_CLAS
000021 RTE_SMS_MGMT_CLAS
000022 RTE_SMS_VSAM_MGMT_CLAS
000023 RTE_PDSE N
000024 RTE_LOAD_OPTIMIZE Y
000025 RTE_DEBUG_SYSOUT "X"
000026 RTE_Debug_SYSOUT "X"
000027 RTE_JCL_SUFF SAH
F13=PFK Help F14=ParmHelp F15=PFK Off F16= F17= F18= F19= F20= F21=PFKShow F22= F23= F24=
```

**Figure 46. Batch parameter member for SYSB**

The contents of the parameter member are structured as follows:

- Variable names and values are defined within sections.
The first variable name section is bounded by $RTE\$

BEGIN and $RTE\$ END lines, where $vvv$ is the
version of the Configuration Tool. Between these lines are variable names and values for the runtime
environment variables.

Following the runtime environment variable name section are sections for all products that are
configured in the runtime environment. These sections are bounded by $Kpp$ END lines, where $pp$ is the 2-character product code, and $vvv$ is the product version.

For example, the variable name section containing the OMEGAMON XE on z/OS V4.2.0 (product code
M5) variable definitions begins with

$KMS$420 BEGIN   *-------- OMEGAMON XE on z/OS V420 -------------*

Comment lines begin with an asterisk.

The first value on a line is the variable name. The second value is the variable value. Each variable
value matches a value supplied on a configuration panel for the runtime environment or for the
product. In some cases, the value is enclosed by double quotation marks.

Table 12 lists the variables that you must update to create a runtime environment with the desired
characteristics.

Table 12. Actions required to update the runtime environment parameters

<table>
<thead>
<tr>
<th>Required actions</th>
<th>Variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>Change the JCL suffix for jobs generated for this runtime environment.</td>
<td>RTE_JCL_SUFF</td>
</tr>
<tr>
<td>You must change the suffix because the CICATB job does not submit the jobs it creates if they already exist in INSTJOBS.</td>
<td></td>
</tr>
<tr>
<td>Set the prefix used in generating started task names.</td>
<td>RTE_STC_PREF</td>
</tr>
<tr>
<td>Note: Some variables, most ending in _STC, already have started task names. Update the existing names with the new prefix.</td>
<td></td>
</tr>
<tr>
<td>Change the prefix used in generating VTAM APPLIDs</td>
<td>RTE_VTAM_APPL_PREF</td>
</tr>
<tr>
<td>Note: Other variables, containing the string <em>APPL</em>, have VTAM APPLIDs. Search the member for instances of the VTAM APPLID prefix, and update the prefix wherever it occurs.</td>
<td></td>
</tr>
<tr>
<td>Set the default TCP/IP communications values:</td>
<td></td>
</tr>
<tr>
<td>• Host name</td>
<td>RTE_TCP_HOST</td>
</tr>
<tr>
<td>• TCP/IP started task name</td>
<td>RTE_TCP_STC</td>
</tr>
<tr>
<td>• IP port for communications</td>
<td>RTE_TCP_PORT</td>
</tr>
<tr>
<td>Note: If the remote monitoring server in the new runtime environment is to report to the hub in the model runtime environment, the IP port number must be the same for both runtime environments.</td>
<td></td>
</tr>
<tr>
<td>(Optional) Update the runtime environment data set high-level qualifiers.</td>
<td></td>
</tr>
<tr>
<td>This actions is not required, because the new runtime environment name is appended to the high-level qualifier to make the data sets unique. Perform this action only if different high-level qualifiers are desired for this runtime environment.</td>
<td>RTE_HILEV</td>
</tr>
<tr>
<td>Scan the member for other occurrences of the existing high-level qualifiers to determine whether those variable values must be updated.</td>
<td>RTE_VSAM_HILEV</td>
</tr>
<tr>
<td>Change the monitoring server from hub to remote.</td>
<td>KDS_CMS_TYPE</td>
</tr>
<tr>
<td>Change the monitoring server name.</td>
<td>RTE_CMS_NAME</td>
</tr>
</tbody>
</table>
### Table 12. Actions required to update the runtime environment parameters (continued)

<table>
<thead>
<tr>
<th>Required actions</th>
<th>Variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>Update the monitoring server TCP/IP communications information:</td>
<td>• KDS_CMS_TCP_HOST&lt;br&gt; • KDS_CMS_TCP_STC&lt;br&gt; • KDS_CMS_TCP_KDEBLST&lt;br&gt; • KDS_CMS_TCP_zzzz_PORT</td>
</tr>
<tr>
<td>• Host name</td>
<td></td>
</tr>
<tr>
<td>• TCP/IP started task name</td>
<td></td>
</tr>
<tr>
<td>• Bind to a specific interface</td>
<td></td>
</tr>
<tr>
<td>• IP port for communications</td>
<td></td>
</tr>
<tr>
<td>• Update the IP port for the appropriate zzzz protocol</td>
<td></td>
</tr>
<tr>
<td>Have this remote monitoring server report to the SYSA hub monitoring server</td>
<td>KDS_HUB_ variables.</td>
</tr>
<tr>
<td>These variables are commented out when the source batch parameter member has a</td>
<td></td>
</tr>
<tr>
<td>hub monitoring server defined. Uncomment them to add information about the</td>
<td></td>
</tr>
<tr>
<td>hub name and the protocols and ports used to connect to it.</td>
<td></td>
</tr>
<tr>
<td>Have all defined agents report to this remote monitoring server.</td>
<td>• Kpp_AGT_CMS_NAME (for agents that run in the monitoring server address</td>
</tr>
<tr>
<td></td>
<td>space)</td>
</tr>
<tr>
<td></td>
<td>• Kpp_CMS_NAM (for agents that run in separate address spaces)</td>
</tr>
<tr>
<td>where pp is the 2-character product code.</td>
<td></td>
</tr>
<tr>
<td>Update the remote monitoring server TCP/IP communication information for agents</td>
<td>• Kpp_CMS_TCP_HOST&lt;br&gt; • Kpp_CMS_zzzz_PORT</td>
</tr>
<tr>
<td>that run in separate address spaces.</td>
<td></td>
</tr>
<tr>
<td>• Host name</td>
<td></td>
</tr>
<tr>
<td>• IP port</td>
<td></td>
</tr>
<tr>
<td>Update the agent TCP/IP communication information for agents that run in</td>
<td>• Kpp_CMS_AGT_HOST&lt;br&gt; • Kpp_CMS_AGT_zzzz_PORT</td>
</tr>
<tr>
<td>separate address spaces.</td>
<td></td>
</tr>
<tr>
<td>• Host name</td>
<td></td>
</tr>
<tr>
<td>• IP port</td>
<td></td>
</tr>
</tbody>
</table>

To see definitions and valid values for all the variables (also called batch parameters), generate a parameter map. See “Obtaining parameter reports” on page 122.

### Step 4. Run the CICATB job to validate the changes

When you specify the SCAN option for the SUBMIT parameter, CICATB validates that the required variables are defined and that the defined values meet the syntax requirements for the respective variables.

To validate the changes you have made to the parameter member:

1. Make the following changes to the CICATB JCL:
   - Modify the BATCHLIB parameter to specify the library where the SYSB member was generated.
   - Modify the BATCHMEM parameter to specify SYSB.
2. Submit the job.
If you submit CICATB from ISPF Edit mode, exit this mode for the INSTJOBS data set. The job does not run until the INSTJOBS data set is free.

A return code of 0 means the job executed successfully. It does not mean that the validation found no errors. The CICATB job output and the CB#rssss member of the shilev.INSTJOBS data set contain a report of the results. The heading of the report is similar to the following screen:

************************************************************************
* DATE: 2008/09/22                                             *
* TIME: 16:06                                                 *
*                                                            *
* RTE NAME:        RUN: 00/22/2008 16:06:39  *                 *
*                                                            *
* INSTLIB: TEST.V621.INSTLIB                                 *
* BATCHLIB: TEST.V621.INSTJOBS                               *
* BATCHMEM: SYSB                                            *
* SUBMIT: SCAN                                              *
*                                                            *
*                                                            *
* THIS REPORT CONTAINS THE FOLLOWING SECTIONS:               *
* 1. PARAMETER ERRORS                                       *
* 2. PARAMETERS YOU HAVE CHANGED                            *
* 3. JOBS SORTED BY GENERATION SEQUENCE                     *
* 4. JOBS SORTED BY MEMBER NAME                             *
* 5. COMPONENTS CONFIGURED IN THIS RTE                      *
************************************************************************

For the SCAN option, only the first two sections are created.

- **Parameter Errors**
  
  This section contains errors the scan detected. Here is an example of errors that might be detected when a hub monitoring server is changed to a remote:

<table>
<thead>
<tr>
<th>PARAMETER NAME</th>
<th>INPUT ERROR</th>
<th>LINE# DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>KDS_CMS_HTTP_PORT</td>
<td>431</td>
<td>This parameter may only be enabled by a Hub TEMS.</td>
</tr>
<tr>
<td>KDS_CMS_CTSOAP</td>
<td>431</td>
<td>This parameter may only be enabled by a Hub TEMS.</td>
</tr>
<tr>
<td>KDS_HUB_GLAPL</td>
<td>431</td>
<td>Missing required parameter</td>
</tr>
</tbody>
</table>

  The first two errors occur because these parameters do not apply and must be removed. The third error occurs because a parameter that is now required has not been defined.

- **Parameters You Have Changed**

  This section contains all parameters whose values have changed from the default values. It is a listing of all the parameters that have been specified, not just the parameters that were changed when the member was updated. Here is a portion of the section:

<table>
<thead>
<tr>
<th>PARAMETER NAME</th>
<th>YOUR VALUE</th>
<th>DEFAULT VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>RTE_DESC</td>
<td>420 and 621 RUNTIME ENV SYSB (none)</td>
<td></td>
</tr>
<tr>
<td>RTE_TYP</td>
<td>SHARING FULL</td>
<td></td>
</tr>
<tr>
<td>RTE_SHARE</td>
<td>SMP (blank)</td>
<td></td>
</tr>
<tr>
<td>RTE_HILEV</td>
<td>PROD.V621 (none)</td>
<td></td>
</tr>
<tr>
<td>RTE_VSAM_HILEV</td>
<td>PROD.V621 (none)</td>
<td></td>
</tr>
</tbody>
</table>
Review the report and correct any errors, and then run the CICATB SCAN again. When there are no errors in the Parameter Errors section, the parameter member can be used to create the new runtime environment successfully.

**Step 5. Create and (optionally) execute the batch jobs to create the runtime environment**

To create the batch jobs that create the runtime environment, but not submit them for execution, submit the CICATB job again with the SUBMIT(NO) option. In this scenario, the systems share DASD, so you can execute CICATB to create the jobs from either the current system (SYSA) or the target system (SYSB).

Depending on the outcome, expect one of the following results:

If you did not correct all of the errors identified in the CICATB(SCAN) report, the jobs are not created and the job output contains the following message:

```
*** BATCH: SCANNER ERROR 10100 FROM KCICRTE
*** NOTE: JOB CREATION CANNOT PROCEED WHEN THERE ARE ERRORS IN THE PARAMETER DECK.
*** YOU MUST MAKE THE PARM DECK ERROR FREE BEFORE THE RTE JOBS CAN BE CREATED.

15:10:25 JOB: CB#R DESC: PARAMETER SCAN REPORT
***
```

If there are no errors when the job is run, the job output contains a listing of the jobs created in the INSTJOBS data set and any warnings generated. The number varies based on the number of products and the product configuration options selected. For this scenario, 29 jobs were created, resulting in the following output:

```
*** BATCH: PRODUCT LIST IS DF550 DS621 M2420 M5420 S3420
***
15:20:43 JOB: CB#ISBR DESC: ALLOCATE RUNTIME LIBRARIES
***
15:20:45 JOB: DS#LSBR DESC: ASM, LINK VTAM LOGMODE TABLE
***
15:20:46 JOB: DS#3SBR DESC: CREATE RUNTIME MBRS
***
15:20:48 JOB: PD#PSBR DESC: CREATE PDS MBRS
***
15:20:49 JOB: PD#QSBR DESC: ALLOC AND INIT PDS
***
15:20:52 JOB: DF#3SBR DESC: CREATE RUNTIME MBRS
***
15:20:54 JOB: DF#4SBR DESC: REGISTER PRODUCT TO THE TEMS
***
15:20:56 WRN: KPD61CPX GENHIST LIBS MISSING
  REASON: data set NOT FOUND. You have enabled the OMEGAMON II for SMS product to collect historical data at the TEMS.
  The TEST.V621.SYSB.RGENHISx libraries are required to store the Data Warehousing data. These libraries are typically allocated and initialized during TEMS configuration. Default PDS
```
control members (KPDPG and KPDAL) will be provided. Please refer to the DF#PSBR job for more information.

15:20:56 JOB: DF#PSBR DESC: CREATE PDS MBRs
15:20:57 JOB: DF#QSBR DESC: ALLOC AND INIT PDS
15:20:59 JOB: M2#5SBR DESC: ALLOCATE ADDITIONAL data setS
15:21:00 JOB: M2#3SBR DESC: CREATE RUNTIME MBRs
15:21:01 JOB: M2#4SBR DESC: REGISTER PRODUCT TO THE TEMS
15:21:03 JOB: M2#05SBR DESC: MODIFY MENU SYSTEM SECURITY
15:21:06 JOB: M2#5SBR DESC: REGISTER PRODUCT TO THE TEMS
15:21:09 JOB: M5#ISBR DESC: CREATE RUNTIME MBRs, AGT TEMS
15:21:14 WRN: KPD61CPX GENHIST LIBS MISSING
   REASON: data set NOT FOUND. You have enabled the IBM Tivoli OMEGAMON XE on z/OS product to collect historical data at the TEMS.
   The PROD.V621.SYSB.RGENHISx libraries are required to store the Data Warehousing data. These libraries are typically allocated and initialized during TEMS configuration. Default PDS control members (KPDPG and KPDAL) will be provided. Please refer to the M5#PSBR job for more information.
15:21:14 JOB: M5#5SBR DESC: CREATE PDS MBRs
15:21:15 JOB: M5#QSBR DESC: ALLOC AND INIT PDS
15:21:17 JOB: S3#4SBR DESC: REGISTER PRODUCT TO THE TEMS
15:21:20 JOB: S3#ISBR DESC: CREATE RUNTIME MBRs, AGT TEMS
15:21:24 WRN: KPD61CPX GENHIST LIBS MISSING
   REASON: data set NOT FOUND. You have enabled the IBM Tivoli OMEGAMON XE for Storage on z/OS product to collect historical data at the TEMS.
   The PROD.V621.SYSB.RGENHISx libraries are required to store the Data Warehousing data. These libraries are typically allocated and initialized during TEMS configuration. Default PDS control members (KPDPG and KPDAL) will be provided. Please refer to the S3#PSBR job for more information.
15:21:24 JOB: S3#PSBR DESC: CREATE PDS MBRs
15:21:25 JOB: S3#3SBR DESC: INSTALL CANDLE SUBSYSTEM
15:21:26 JOB: CB#3SBR DESC: CREATE SINGLE VTAM NODE
15:21:26 JOB: CB#PSBR DESC: SYSTEM PROCEDURE COPY JOB
15:21:28 JOB: CB#NSBR DESC: SYSTEM VTAMLST COPY JOB
15:21:30 JOB: CB#2SBR DESC: LOAD RUNTIME LIBRARIES
15:21:32 JOB: STA4SBR DESC: BUILD USER MODIFIED ELEMENTS
15:21:32 JOB: SUB#5SBR DESC: CLIST TO DRIVE JOB SUBMISSION
The first line in the listing identifies the products that are being configured in the runtime environment by their 2-character product code and version.release.modification level.

The output contains several warnings, all related to missing data sets required for historical data collection. These data sets are created, if they do not already exist, by the first job that runs and produces a warning.

The CB#PSBR and CB#NSBR jobs create sample JCL members in the rhilev.rte.RKANSAMU data set for copying the created started tasks procedures and VTAM nodes to PROCLIB and VTAMLST, respectively. No actual copying is performed. The sample jobs can be edited and used if desired. The SUB#ssss job (where ssss is the JCL suffix for the runtime environment) is a CLIST created to control the job submissions. The CLIST submits the jobs in the correct order. It also contains optional jobs that are commented out, as shown in this partial listing of the CLIST contents:

```
PROC 0 CANJOBS(TEST.V621.INSTJOBS)
CONTROL NOMSG NOFLUSH ASIS

/* ALLOCATE RUNTIME LIBRARIES
/* ** THIS JOB ALLOCATES EVERY
/* LIB REQUIRED IN THIS RTE
SUBMIT('&CANJOBS(CB#1SBR)')

/* OPTIONAL: ASM, LINK VTAM LOGMODE TABLE
/* ** OPTIONAL JOB;
/* RUN IF REQUIRED
/*SUBMIT('&CANJOBS(DS#LSBR)')

/* CREATE RUNTIME MBRS
SUBMIT('&CANJOBS(DS#3SBR)')

/* CREATE PDS MBRS
SUBMIT('&CANJOBS(PD#PSBR)')

/* ALLOC AND INIT PDS
SUBMIT('&CANJOBS(PD#QSBR)')

/* CREATE RUNTIME MBRS
SUBMIT('&CANJOBS(DF#3SBR)')

/* REGISTER PRODUCT TO THE TEMS
SUBMIT('&CANJOBS(DF#4SBR)')
```

Executing the CLIST creates the SYSB runtime environment and configures the products according to the contents of the SYSB batch parameter member. You can also submit the configuration jobs manually in the order specified in the CLIST.

To submit the jobs automatically after the CICATB job runs, instead of executing the CLIST or submitting them manually, specify SUBMIT(YES) or SUBMIT(RERUN) in the CICATB parameters. Both of these options create the same jobs as the SUBMIT(NO) option, and write each job to the INTRDR JES2 destination and to the shilev.INSTJOBS data set.

- If you specify SUBMIT(YES), automatic submission occurs only if the jobs do not already exist in the shilev.INSTJOBS data set.
- If you specify SUBMIT(RERUN), automatic submission occurs even if the jobs already exist in the shilev.INSTJOBS data set. You can use this option to avoid deleting the existing configuration jobs before reconfiguring the runtime environment in batch mode.
Important: If the runtime environment contains any user-modified elements and if you intend to run the CICATB job with SUBMIT(RERUN) or SUBMIT(YES) specified, back up the runtime libraries before running CICATB.

When the jobs are submitted automatically, the CICATB output contains a record of their submission, as shown here:

READY
ISPSTART CMD(EX 'TEST.V621.INSTLIB(KINSTALL)' 'INSTLIB(TEST.V621.INSTLIB) BATCHLIB(TEST.V621.INSTJOBS) BATCHMEM(SYB) SUBMIT(YES)')
IDC0549I MEMBER KCICTEMP DELETED
***
*** BATCH: PRODUCT LIST IS DF550 DS621 M2420 M5420 S3420
***
*** 11:41:12 JOB: CB#ISBR DESC: ALLOCATE RUNTIME LIBRARIES
***
*** 11:41:14 JOB: DS#LSBR DESC: ASM, LINK VTAM LOGMODE TABLE
***
*** 11:41:16 JOB: DS#3SBR DESC: CREATE RUNTIME MBRS
***
*** 11:41:19 JOB: PD#PSBR DESC: CREATE PDS MBRS
***
*** 11:41:20 JOB: PD#QSBR DESC: ALLOC AND INIT PDS
***
*** 11:41:25 JOB: DF#3SBR DESC: CREATE RUNTIME MBRS
***
*** 11:41:28 JOB: DF#4SBR DESC: REGISTER PRODUCT TO THE TEMS
***
*** 11:41:31 WRN: KP61CPX GENHIST LIBS MISSING
*** REASON: data set NOT FOUND. You have enabled the OMEGAMON II for SMS product to collect historical data at the TEMS.
***
*** The TEST.V621.SYSB.RGENHISx libraries are required to store the Data Warehousing data. These libraries are typically allocated and initialized during TEMS configuration. Default PDS control members (KPDPG and KPDAL) will be provided.
*** Please refer to the DF#PSBR job for more information.
*** 11:41:31 JOB: DF#PSBR DESC: CREATE PDS MBRS
***
*** 11:41:34 JOB: DF#QSBR DESC: ALLOC AND INIT PDS
***
*** 11:41:37 JOB: M2#5SBR DESC: ALLOCATE ADDITIONAL data setS
***
*** 11:41:39 JOB: M2#3SBR DESC: CREATE RUNTIME MBRS
***
*** 11:41:42 JOB: M2#4SBR DESC: REGISTER PRODUCT TO THE TEMS
***
*** 11:41:44 JOB: M2#OSBR DESC: MODIFY MENU SYSTEM SECURITY
***
*** 11:41:49 JOB: M5#4SBR DESC: REGISTER PRODUCT TO THE TEMS
***
*** 11:41:53 JOB: M5#ISBR DESC: CREATE RUNTIME MBRS, AGT TEMS
***
*** 11:41:59 JOB: M5#PSBR DESC: CREATE PDS MBRS
***
*** 11:42:01 JOB: M5#QSBR DESC: ALLOC AND INIT PDS
***
*** 11:42:05 JOB: S3#4SBR DESC: REGISTER PRODUCT TO THE TEMS
***
*** 11:42:09 JOB: S3#ISBR DESC: CREATE RUNTIME MBRS, AGT TEMS
***
*** 11:42:14 JOB: S3#PSBR DESC: CREATE PDS MBRS
***
*** 11:42:16 JOB: CN#3SBR DESC: INSTALL CANDLE SUBSYSTEM

228 IBM Tivoli OMEGAMON XE and Tivoli Management Services on z/OS: Common Planning and Configuration Guide
The time stamps indicate when the job was submitted, not necessarily when it ran. The jobs are submitted with the same job name, so they execute in sequence. If any job scheduling is in place that might cause the jobs not to execute in sequence, use the SUBMIT(NO) option and submit the jobs manually. The CICATB job ends before the runtime environment creation jobs complete. Monitor the SYSLOG/OPERLOG to see the job processing and to identify any unexpected return codes (anything other than 0 or 4). This example shows a portion of the SYSLOG/OPERLOG messages, where `jobname` is the jobname used for the jobs submitted by CICATB.

```
IEF403I CICATB - STARTED - TIME=11.40.42
IEF403I jobname - STARTED - TIME=11.41.13
Jobname Procstep Stepname CPU Time EXCPs RC
jobname --None-- STEP1 00:00:00 17 00
jobname --None-- STEP3 00:00:00 740 00
IEF404I jobname - ENDED - TIME=11.41.32
IEF403I jobname - STARTED - TIME=11.41.32
Jobname Procstep Stepname CPU Time EXCPs RC
PAGE 2
jobname --None-- STEP1 00:00:00 32 00
jobname --None-- STEP2 00:00:00 60 00
jobname --None-- STEP3 00:00:00 32 00
jobname --None-- STEP4 00:00:00 25 00
jobname --None-- STEP5 00:00:00 21 00
jobname --None-- STEP6 00:00:00 36 00
IEF404I jobname - ENDED - TIME=11.41.33
IEF403I jobname - STARTED - TIME=11.41.33
Jobname Procstep Stepname CPU Time EXCPs RC
jobname --None-- COPYUSRJ 00:00:00 32 04
jobname --None-- STEP1 00:00:00 25 00
jobname --None-- STEP2 00:00:00 33 00
jobname --None-- STEP3 00:00:00 24 00
jobname --None-- STEP4 00:00:00 49 00
jobname --None-- STEP5 00:00:00 32 00
jobname --None-- COPYUSR P 00:00:00 30 04
jobname --None-- STEP6 00:00:00 30 00
IEF404I jobname - ENDED - TIME=11.41.34
```

Figure 47. SYSLOG/OPERLOG messages

For this submission, runtime environment creation jobs completed about 2 minutes after the CICATB job ended.

**Step 6. Customize and complete the configuration**

If all jobs have completed with 0 or 4 return codes, the runtime environment has been successfully created. You can now view and modify it from the Configuration Tool. Start the Configuration Tool and select a product to configure. The list of runtime environments now includes the newly created SYSB
runtime environment

![Figure 48. Runtime Environments (RTEs) panel with new SYSB runtime environment]

You can now perform further customization. For example, if OMEGAMON XE on z/OS is installed, you might want to improve performance by moving the sysplex proxy from the hub on SYSA to the remote on SYSB. Additionally, you must perform any steps required to complete the configuration outside the Configuration Tool.

Now you can enable the SYSB runtime environment by performing the post-configuration procedures required for the products defined in the new runtime environment. Use the Configuration Tool to review the Complete the configuration steps for each of the products in the environment, or to generate a file that consolidates the post-configuration instructions for all the products in the runtime environment. See also Chapter 6, “Completing the configuration,” on page 129.

To generate the consolidated instructions, complete the following steps:

1. From the Configuration Tool Main Menu (Figure 25 on page 105), enter 3 (Configure products).
2. On the Product Selection Menu, select any product.
3. On the Runtime Environments (RTEs) panel, enter Z (Utilities) next to the runtime environment in which you want to complete product configuration.
4. On the RTE Utility Menu (Figure 50 on page 233), enter 9 (Create Consolidated Complete the configuration instructions).

This option creates JCL containing a consolidated list of Complete the configuration instructions for all products configured in the runtime environment. The list is stored in the shlev.INSTJOBS data set, in a member named DFII@ssss (where ssss is the unique JCL suffix for the runtime environment).

5. Perform the tasks listed in the Complete the configuration instructions. The list is generated dynamically and tailored specifically to your configuration.

The most common tasks for configuring a monitoring server are covered in Chapter 6, “Completing the configuration,” on page 129. For detailed information about completing the configuration of monitoring agents, see the configuration documentation for each monitoring agent in the Tivoli Monitoring and OMEGAMON XE Information Center at http://publib.boulder.ibm.com/infocenter/tivihelp/v15r1/index.jsp?toc=

Tips:
- You do not have to configure the monitoring server in a runtime environment more than once, even though the monitoring server shows up on the Product Component Selection Menu for every monitoring agent.
- The Complete the configuration list for every monitoring agent on z/OS includes the task of registering the monitoring agent with the monitoring server in its runtime environment. See IBM Tivoli Management Services on z/OS: Configuring the Tivoli Enterprise Monitoring Server on z/OS.
Appendix C. Configuration Tool reference

The topics in this section contain information about the services and utilities provided by the Configuration Tool. It also contains information about the Runtime Environments (RTE) panel and the actions and utilities associated with it.

The following ISPF-related restrictions apply to the Configuration Tool:

- If you are using a 3270 Model 2 (24 x 80) display, you must turn off the predefined function (PF) keys so that the Configuration Tool panels are not truncated. To turn off the predefined function keys, type PFSHOW on any ISPF command-line and press Enter until the function keys are no longer displayed.

- You cannot use the ISPF feature for edit recovery. If you enter the ISPF RECOVERY ON command, edits produce a recovery error message. Enter the RECOVERY OFF command to suppress the error messages.

Configuration services and utilities

The configuration services and utilities are used to perform various services on a runtime environment and to collect diagnostic information. Some of the services modify the Configuration Tool values stored in ISPF tables. Do not modify any values unless you are told to do so in the documentation or by Software Support personnel. If the Configuration Tool values are modified incorrectly, the Configuration Tool might stop functioning or produce unpredictable results.

Take these steps to access the configuration services and utilities:
1. On the Configuration Tool Main Menu panel enter 3 (Configure products).
   The Configure Products panel is displayed (Figure 29 on page 110).
2. Enter 5 (Services and utilities).
   The Configuration Services and Utilities panel is displayed, as shown in Figure 49.

   KCIPCUTL ----------- CONFIGURATION SERVICES AND UTILITIES ------------------------
   OPTION ===>
   Enter the number to select an option: Last selected
   Services:
   1  Unlock runtime high-level qualifiers
   Utilities:
   3  DEBUG options
   4  Display an ISPF table
   5  Execute a CLIST in the TKANCUS library
   6  Prepare user libraries

   F1=Help F3=Back

   Figure 49. Configuration Services and Utilities panel

Select any of the following options:

Unlock high-level qualifier values
   You specify high-level qualifier values when you set up your configuration environment. If you have to modify these values later, you must first unlock them.
Attention: If you unlock and change the high-level qualifiers, the Configuration Tool does not automatically delete and reallocate the existing libraries. The jobs generated by the Configuration Tool fail if they are pointing to incorrect libraries.

DEBUG options
When IBM Software Support personnel request additional information to investigate a problem you have encountered with the Configuration Tool, use this option to specify or modify DEBUG parameter values.

Display an ISPF table
Use this option to view the contents of an ISPF table located in the data library. You can limit the information displayed for an ISPF table by specifying one to three sets of display criteria under Optional selection parameters. You can use these actions on this table:

- **END (F3)**
  Go to previous record
- **ENTER**
  Go to next record
- **CANCEL**
  Go back to prior panel. To cancel an action, you must type the word cancel. The usual End key (F3) does not function as usual here.
- **UP/DOWN**
  Use scroll variable

Execute a CLIST in the TKANCUS library
When IBM Software Support personnel request additional information to investigate a problem you have encountered with the Configuration Tool, they might ask you to execute a specific CLIST/REXX executable that is in the TKANCUS library. Specify the values requested by IBM Software Support personnel.

Prepare user libraries
The Prepare User Libraries utility supports the allocation of any user libraries added since the prior installed release. This utility displays the Runtime Environments (RTES) for Conversion panel. This panel lists all the runtime environments in this installation environment except base environments. On this panel, you can prepare user libraries for the default listing (all runtime environments) or delete the runtime environments you want to exclude from the conversion process.

The Configuration Tool generates a shilev.INSTJOBS(KCIJSP01) job to allocate the applicable new user libraries for each runtime environment selected.

Runtime environment utilities
The runtime environment utilities perform various useful processes for maintaining a runtime environment and creating new runtime environments using batch mode.

Take the following steps to access the utilities:
1. On the Main Menu panel, select option 3 (Configure products). The Configure Products panel is displayed.
2. Select option 1 (Select product to configure). The Product Selection Menu panel is displayed.
3. Select a product that is configured in the runtime environment for which you want to access the utilities. The Runtime Environments (RTES) panel is displayed.
4. Enter Z beside the appropriate runtime environment. The RTE Utility Menu is displayed.
Select any of the following utilities:

Create/Clone batch mode parameters
This utility creates a member that contains the configuration values for all the products in the selected environment. This member can be used to back up the configuration parameters of an existing runtime environment, restore a runtime environment based on a saved configuration, and reconfigure an existing runtime environment when something in the environment has changed.

You can also clone the parameter member for an existing environment and modify it as necessary to create a new runtime environment by using the batch mode job, CICATB. You can make an exact duplicate of the current environment (Clone batch mode parameter deck, unmodified) or eliminate some of the products configured in the environment (Clone batch mode parameter deck, with modifications). You can have the Configuration Tool open the member after it is generated so you can edit the parameter values for the target environment.

See Chapter 8, “Using the Configuration Tool batch mode to replicate a configured environment,” on page 179 for more information about the use of batch mode parameters. See “Step 2. Create and clone a batch mode parameter member” on page 183 for step-by-step instructions for using this utility.

Create the system variable parameters
This utility creates a system variable support parameter member for the selected runtime environment. Supported products resolve these parameters at execution time. z/OS system variable usage must be enabled for the selected runtime environment to execute this utility. See Appendix A, “Enabling system variable support,” on page 199 for instructions for using this utility.

Create system variable VTAM major node rename job
This utility creates a job named CB#VSHRA in the rhilev.rte.RKANSAMU library. The job creates new major nodes with names that are resolved from your system variable specification. The job must be submitted on every remote system where system variable support is being used. After the new nodes are created, they must be copied to SYS1.VTAMLST.

See “Create the VTAM major node rename job” on page 204 for instructions on using this utility.

Create VTAM major node (one node for all products)
This utility creates a single VTAM major node in the rhilev.rte.RKANSAMU library that contains the VTAM applids for all of the IBM products you have configured in the runtime environment. This single major node is then used in place of the individual product major nodes.
Generate sample transport JCL
This utility generates sample batch mode configuration scenarios and transport runtime environment jobs and copies them to the rhilev.rte.RKANSAMU library.

See Chapter 8, “Using the Configuration Tool batch mode to replicate a configured environment,” on page 179 for descriptions of the sample transport jobs and scenarios and step-by-step instructions for generating and using the jobs.

Generate sample system procedure copy JCL
This utility creates sample JCL to copy system procedures from RKANSAMU to a user-specified system library for this runtime environment. You can specify any system library if you do not want to update your PROCLIB directly.

The Configuration Tool generates the sample IEBCOPY job named CB#Pssss, which creates the KCISYSPJB sample in the rhilev.rte.RKANSAMU library. The CB#Pssss job generates a compression step for the respective libraries if this runtime environment is not using the PDSE format. Before you submit the KCISYPJB job, ensure that you have authority to update the respective system libraries.

Generate sample system VTAMLST copy JCL
This utility creates sample JCL to copy VTAM major node members from RKANSAMU to a user-specified system library for this runtime environment. You can specify any system library if you do not want to update your VTAMLST directly. The default system library is SYS1.VTAMLST.

The Configuration Tool generates the sample IEBCOPY job named CB#Nssss, which creates the KCISYNJB sample in the rhilev.rte.RKANSAMU library. The CB#Nssss job generates a compression step for the respective libraries if this runtime environment is not using the PDSE format. Before you submit the KCISYNJB job, ensure that you have authority to update the respective system libraries.

If you have enabled system variable support for this runtime environment, this option does not apply.

Analyze user modified elements
This process facilitates the identification of user modified elements and shows which of those elements are modified by a specific batch job. This process is not applicable to a remote runtime environment.

See “Step 3. Adjust and validate the parameters” on page 186 for information on using analysis of user-modified elements before submitting batch mode jobs.

Create consolidated Complete the Configuration and README reports
This option concatenates the contents of the Complete the Configuration instructions for all products configured in a runtime environment into a single, printable report. The report is stored in the shilev.INSTJOBS data set, in a member named DFI@ssss (where sssss is the unique JCL suffix for the runtime environment).

This option also consolidates all the README files in the thilev.TKANCUS data set and stores them in a single file named $README in the &shilev.INSTJOBS data set.

Process Edit subcommands for INSTJOBS or RKANPARU members
This option applies a list of EDIT subcommands to configuration batch jobs in the shilev.INSTJOBS data set, sample parameter decks in the thilev.TKANCUS data set, or members of the rhilev.rte.RKANPARU data set.

Verify configuration and generate parameter map
This utility has two options:

• By using the Verify configuration option, you can verify the creation and successful execution of all configuration batch jobs, and the creation of all required runtime data sets and members. You can run the configuration verification job by selecting this option at any time; it is a good idea to run it as soon as you finish configuring a runtime environment and before you load the
runtime libraries. Additionally, the configuration verification job is run automatically when you load the runtime libraries. This job can be run in batch mode only; it cannot be run in foreground. If a large number of components and products are configured in the runtime environment, the configuration verification batch job might take a long time to run.

The configuration verification job is specific to the runtime environment. You must generate a unique configuration verification job for each runtime environment you want to verify. If the configuration of the runtime environment changes (for example, if new components are configured or existing components are reconfigured), be sure to regenerate and rerun the configuration verification job.

The configuration verification job generates a report that lists the required data sets and members; indicates the batch job that creates each one; identifies which data sets and members are missing, and which batch jobs are missing or did not complete successfully; and specifies which of the missing or unsuccessful batch jobs and which of the missing data sets and members are required and which are optional. This report is stored in the &thilev.INSTJOBS.IVP$ssss(IP1ssss) member, where ssss is the JCL suffix for the runtime environment. The job information in the report is retrieved from a configuration batch job log stored in the shilev.INSTLOG data set. This log is cumulative for all jobs run to configure all runtime environments in the installation environment.

- The Generate parameter map option generates a parameter map for the runtime environment. If the map is generated before any products are configured in the runtime environment, a basic report is generated for all the components installed into the target libraries that were installed by SMP/E. The report gives a short description and other basic information about each parameter. If the map is generated after products are configured, the report also contains the value for each configured parameter and other post-configuration parameter information such as the name of the configuration job that stored each parameter and data about completion of each job and each step. The job and step completion codes reported in the parameter map are retrieved from the configuration batch job log stored in the shilev.INSTLOG data set.

The parameter map is generated in foreground and is displayed as soon as it is complete. If a large number of components and products are configured in the runtime environment, generating the parameter map might take several minutes. The parameter map for a runtime environment is stored in the shilev.INSTJOBS.IVP$ssss(PMAPssss) member.

A detailed parameter map requires data that is produced by the configuration verification job. Consequently, to produce a detailed parameter map for a runtime environment, you must first run the configuration verification job for the same runtime environment. If you subsequently reconfigure components or configure additional component in the runtime environment, regenerate and rerun the configuration verification job and then regenerate the detailed parameter map for the runtime environment.

Enter README IVP on the command-line for more information about the configuration verification report and the parameter map. You can also find more information about the parameter map in "Set appropriate parameter values" on page 122.

Upgrade attribute and catalog members

This option updates the catalog and attribute files in all runtime environments accessible to the Configuration Tool, by copying the most recently installed files from the thilev.TKANDATV target data set that was installed by SMP/E to the runtime library rhilev.rte.RKANDATV for each runtime environment. For more information on catalog and attribute files, see the IBM Tivoli Monitoring: Installation and Setup Guide.

Generate health check configuration members

Use this option to configure the two health checks implemented by the Configuration Tool. One health check verifies, at a user-defined interval, that all started tasks configured for a runtime environment are running. The default interval is 30 minutes, and the valid range is 10 minutes to 1440 minutes (24 hours). The second health check, which is run at IPL, verifies the APF authorization of all load libraries that must be APF-authorized for a runtime environment. Each health check runs as a system REXX EXEC and enables a system alert if an exception is found.
Batch mode utilities

Three Configuration Tool utilities are available to assist you if you are using batch mode processing.

**KCISETUP**
Sets up the environment required for using the ISPF macros provided with the Configuration Tool.

**KCICFKEY**
Manages the PF keys that are used for the Configuration Tool batch utilities.

**KCICPGHP**
Displays help information for parameters in a batch parameter deck member.

These utilities are designed to run outside the Configuration Tool, but can also be used while in the Configuration Tool.

**KCISETUP: Set up the environment for batch utilities**

The KCISETUP utility sets up the environment that is necessary for using the other Configuration Tool batch utilities. This utility must be run once after starting your TSO ISPF session and can be run only from an ISPF session. Before you can use the KCISETUP utility, you must generate the KCISETUP member in your `shilev.INSTLIB` data set.

KCISETUP can be run only once per session. No confirmation message is issued to indicate successful completion of KCISETUP.

**About this task**

Take the following steps to generate KCISETUP.

**Procedure**

1. Start the Configuration Tool.
2. On the Main Menu panel select option 3 (Configure products). The Configure Products panel is displayed.
3. Select option 5 (Services and utilities). The Configuration Services and Utilities panel opens.
4. Select option 2 (Create batch mode job). Wait until you see the message CICATB JOB CREATED in the upper-right corner of the panel.
   
   **Tip:** The default location for the CICATB job is the INSTJOBS library. To see the complete path of the library the batch mode job was created in, press F1. This displays a text box with the complete path at the bottom of the panel. The text box is shown only if F1 is pressed immediately after the option 2 (Create batch mode job) was selected. Otherwise, the general help for the panel is shown.
5. Press F3 until you return to the main menu.

**Invoke KCISETUP**

**About this task**

KCISETUP must be created once on an image and can be used for all subsequent parameter deck processing on that image. If the ISPF environment on the new image is different the one on the model image, you must recreate KCISETUP.

**Procedure**

Invoke the environment setup utility using either of the following methods:

- From the ISPF command-line enter
TSO EXEC 'shilev.INSTLIB(KCISETUP)'

• From the ISPF Primary Option Menu, select Enter TSO or Workstation commands and enter
  EXEC 'shilev.INSTLIB(KCISETUP)'

**KCICFKEY: Manage ISPF PF Keys**

The KCICFKEY utility is used to manage ISPF session PF keys that are used for batch utilities. This includes turning the PF keys on and off, and toggling which set of keys display. This utility can be run only under an ISPF session.

**Before you begin**

If you plan to use KCICFKEY to manage the ISPF session PF keys for the batch utilities, you must turn on the predefined function keys. To do this, issue the PFSHOW command from either the ISPF command-line or any of the Configuration Tool panel command-lines.

Before using this Configuration Tool batch utility, you must use the KCISETUP utility to set up the environment.

To use the KCICFKEY utility, your ISPF session must support 24 PF keys. Follow these steps to set up your ISPF session to support 24 PF keys:

1. From the ISPF Primary Option Menu under Settings, select Terminal and user parameters > Function keys > Non-Keylist PF Key settings.
2. In the Number of PF Keys field, type 24 and press F3 to return to the ISPF Primary Option Menu.

If the ISPF session is not set up to support 24 PF keys, the KCICFKEY utility issues the following ISPF warning message:

PFKEYS COUNT ERROR. Number of PF Keys must be 24. See ISPF Settings.

The Configuration Tool PF Key Manager owns PF keys 13 through 24. On keyboards that do not support 24 PF keys, PF keys 13-24 are obtained by holding the Shift key and pressing a function key. While the Shift key is pressed, function keys 1-12 become 13-24. After you exit the Configuration Tool PF Key Manager, all of your prior PF key and Show State settings are restored.

**About this task**

Take the following steps to invoke the KCICFKEY utility:

**Procedure**

Invoke the PF Key Manager using one of the following methods:

• From the ISPF Edit command-line, enter
  KCICFKEY state
  To use this method you must be running under an ISPF session, editing a member or data set.

•
  – From the ISPF Edit command-line, enter
    KCICFKEY state
    To use this method you must be running under an ISPF session, editing a member or data set.
  – From the ISPF command-line, enter
    TSO KCICFKEY state
    where state is the desired state of the Configuration Tool PF keys.

The valid states are:
ON       Turns on the PF keys.
OFF      Turns off the PF keys.

SHOW     Toggles the PF keys (if active) among All (1–24), Alternate (13–24), and Primary (1–12).
HELP     Displays the PF Key Manager help information.

If state is not specified, the Configuration Tool PF keys toggle between ON and OFF.
The PF Key Manager sets the following keys for the batch utilities:
F13      Displays the PF Key Manager Help information.
F14      Provides detailed help information for the batch parameter on the line where the cursor is positioned in the batch parameter deck.
F15      Turns off the PF keys.
F21      Toggles the PF keys shown at the bottom of the display among All (1-24), Alternate (13-24), and Primary (1-12).

**KCICPGHP: Display help for configuration parameters**

The KCICPGHP utility is used to display help information for parameters in a batch parameter deck member. The detailed help information is used to help you modify or construct a batch parameter deck. This utility must be run from an ISPF Edit session.

**Before you begin**

Before using this batch utility, you must use the KCISETUP utility to set up the environment.

**Procedure**

Invoke the help utility using either of the following methods:

- From an ISPF Edit command-line when editing the batch parameter deck, enter the command KCICPGHP, position the cursor on the row that contains a batch parameter, and then press Enter.
- Position your cursor on the row that contains a batch parameter and then select the PF key assigned by the Configuration Tool PF Key Manager.

With either method, you must position the cursor on the row that contains the batch parameter. The utility then isolates the parameter, performs a lookup, and displays a popup window with the detailed help information.

The batch parameter help contains the sections shown in Table 13.

<table>
<thead>
<tr>
<th>Parameter section</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Title Information</td>
<td>The parameter name, followed by a short description of the parameter.</td>
</tr>
<tr>
<td>Description Area</td>
<td>The detailed help information for the parameter. This area is scrollable, as indicated by the (+) indicator on the bottom right. F7 and F8 are assigned to scroll this area.</td>
</tr>
<tr>
<td>Attribute Area</td>
<td>The attributes of the parameter. This information can help you determine what type of data is expected for this parameter.</td>
</tr>
<tr>
<td>Key Area</td>
<td>The function key assignments that apply only to the pop-up window. <strong>Note:</strong> F5 (Show All) displays the help information for all parameters that make up this product.</td>
</tr>
</tbody>
</table>
The Runtime Environments (RTE) panel

The Runtime Environment panel in the Configuration Tool, shown in Figure 51, is one of the primary recurring panels in all configuration operations.

**Figure 51. Runtime environments panel**

This panel supports the following options:

**A (Add RTE)**
Defines the runtime environment to the Configuration Tool. This option creates a definition for the Configuration Tool only. For step-by-step instructions on adding a runtime environment, see page 111.

**B (Build libraries)**
Generates a job to allocate the required runtime libraries for the selected product. This job is presented for your review and submission. Action B must be performed for every runtime environment that will contain the selected product. This includes base runtime environments. For step-by-step instructions, see page 120.

**C (Configure)**
Presents panels to collect the parameter values required to configure the selected product for this runtime environment. Default values are used when applicable. Action C must be performed after Action B for every runtime environment that contains the selected product. You do not have to perform action C for a base runtime environment.

**L (Load all product libraries after SMP/E)**
Loads all product libraries after SMP/E maintenance or product configuration. This action generates a job to load the members of the target libraries that were installed by SMP/E into the runtime environment libraries.

**D (Delete)**
Deletes the runtime environment. The runtime environment definition and associated configuration values are removed from the Configuration Tool and a batch job is created to delete the libraries. See “Deleting a runtime environment” on page 241 for more information and instructions.

**U (Update)**
Displays current runtime environment values and indicates those you can change. To apply changes, you must use actions B, C and L in sequence for the affected products. You must also complete any configuration steps required outside the Configuration Tool, such as copying renamed procedures to your PROCLIB.

**V (View Values)**
Displays current runtime environment values.

**Z (Utilities)**
Displays a list of installed utilities used to perform various useful processes for maintaining the runtime environment. (See “Runtime environment utilities” on page 232 for descriptions.)
R (README Table of Contents)
Displays a list of available README files for selection (Figure 52).
Select the README file of interest. If you select AAA, the Configuration Tool displays information about new features that affect the configuration of all components and products installed in the CSI.
To display a README file from any other Configuration Tool panel, enter README ccc (where ccc is the 3-character code for the type of README file) at the command-line.

Table 14 shows the actions that must be performed when new products or maintenance for existing products are installed.

Table 14. Actions required for configuration and maintenance tasks

<table>
<thead>
<tr>
<th>Installation and maintenance tasks</th>
<th>Add RTE (A)</th>
<th>Build libraries (B)</th>
<th>Configure (C)</th>
<th>Load all libraries after SMP/E (L)</th>
</tr>
</thead>
<tbody>
<tr>
<td>You installed new products or new versions of existing products and you want to create a new runtime environment.</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>You installed new products and you want to use an existing runtime environment.</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>You installed maintenance for existing products, and the PSP bucket file indicates that the configuration settings for the product must be changed.</td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>You installed maintenance for existing products, and the PSP bucket file indicates that no changes to the configuration settings are required. Note: For sharing-with-SMP/E runtime environments, you are not required to reload the libraries unless the action is specifically indicated by ++HOLD information on a Program Temporary Fix (PTF).</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
</tbody>
</table>

If you are installing a new version of an existing product in an existing runtime environment, you do not have to perform the Build and Configure steps unless you want to take advantage of new features that are not enabled by default. You only have to load the libraries. If you are upgrading a Tivoli Enterprise Monitoring Server, you must also run the monitoring server migration job, DS#Mssss. For more information, see the IBM Tivoli OMEGAMON XE and IBM Tivoli Management Services on z/OS: Upgrade Guide.
**Loading runtime environments**

The Load All Product Libraries After SMP/E option generates JCL to load the runtime libraries for this runtime environment from the target libraries that were installed by SMP/E.

The runtime environment libraries must be loaded under following conditions:
- after initial product configuration
- after SMP/E maintenance
- after configuring a new product into an existing runtime environment

In a sharing configuration, only the data sets for the master (shared) full or base runtime environment must be loaded after SMP/E maintenance is applied, because these types of runtime environments contain copies of the target libraries that were installed by SMP/E. In a sharing-with-base or sharing-with-full configuration, the data sets for the master runtime environment are loaded automatically when the Load option is selected for the sharing runtime environment.

If the runtime environment is set up to share the target libraries that were installed by SMP/E, no load of the runtime environment libraries is required after maintenance.

**Deleting a runtime environment**

If you no longer require a particular runtime environment, including the runtime libraries and the configuration values set for the runtime environment, you can delete it.

Confirm that this runtime environment is no longer required before proceeding. If you delete a runtime environment, any other runtime environment that shares libraries with it becomes inoperable. The delete job deletes all libraries that match one of the following patterns:
- `rhilev.rte.*`
- `rohilev.rte.*`

Manually delete any libraries that you allocated for this runtime environment with a different high-level prefix. Also confirm that any libraries that match the pattern `rhilev.rte.*` or `rohilev.rte.*` for a base runtime environment (which might not be part of the runtime environment) are renamed if you do not want to delete those libraries.

As a precaution, back up the entire SMP/E and runtime environment, including the `shilev.INST*` libraries. This allows the runtime environment and the installation libraries to be restored as required. Restoring the runtime environment alone is not sufficient because the runtime environment information is deleted from these installation libraries.

Follow these steps to delete a runtime environment.
1. Enter 0 next to the runtime environment you want to delete and press Enter. This displays the Delete the Runtime Environment panel shown in Figure 53 on page 242.
On the Delete Runtime Environment panel, enter Y in response to the following question:
Are you sure you want to delete this RTE?

2. Review the JCL and submit the job. Check the line Delete rhilev.*. Ensure that this applies only to your libraries. This job deletes all libraries that match the pattern.

3. Press F3 to return to the Main Menu.

Important:

After you respond Y on Figure 53 and the JCL for deleting the runtime environment is generated, you are committed to deleting the runtime environment. If you back out of the JCL without submitting it, the runtime environment is removed from the Runtime Environments (RTEs) panel, but the data sets for that runtime environment are not deleted and these data sets are no longer accessible from the Configuration Tool. If you did not submit the JCL when it was presented, locate the delete job in the shilev.INSTJOBS data set, and submit the job manually to clean up the data sets.
Appendix D. PARMGEN Reference

The topics in this section contain information about the services and utilities provided by the Parameter Generator configuration method. It also contains information about the PARMGEN Workflow panel and the actions and components associated with it.

KCIRPLBS online help macro

The KCIRPLBS online help macros provides help for parameters in the configuration profiles. After the macro is set up, help can be invoked by pressing setting the cursor on a parameter and pressing F14.

To set up the help macro, copy KCIRPLBS from %GBL_TARGET_HILEV%.TKANCUS to your SYSPROC concatenation. To find your SYSPROC concatenation, enter TSO ISRDDN on the ISPF command line. Note that after you copy the macros to your SYSPROC concatenation, you can run them directly from the command line without specifying the fully qualified library name.

Tip: An alternative to copying KCIRPLBS set-up macro to your SYSPROC concatenation is to execute the REXX program directly from your %GBL_TARGET_HILEV%.TKANCUS library by executing the following command: 

```
=> TSO EXEC '%GBL_TARGET_HILEV%.TKANCUS(KCIRPLBS)'.
```

To invoke the help:
1. Enter TSO KCIRPLBS at the command line to run the help macro.
2. If necessary, enter PFSHOW ON at the command line to display the PF keys.
3. Place the cursor anywhere on the line containing the parameter for which you want help text to be displayed, and press F14=ParmHelp key.

Other sources of help for customizing the configuration profile include the following:

- Comments in the configuration profile
- IBM Tivoli OMEGAMON XE and Tivoli Management Services on z/OS: Common Parameter Reference
- [PARMGEN Technote](http://www.ibm.com/support/docview.wss?uid=swg21417935)
- Product-specific Parameter Reference Guide

XF edit macro

The XF (“exclude find”) macro is useful in cases where you want to EXCLUDE certain parameters when editing PARMGEN's configuration profiles. For example, you can EXCLUDE all parameters for started tasks, then FIND all occurrences of these parameters so you can quickly issue a CHANGE ALL command.

IBM provides an XF macro with PARMGEN for your use. The macro comprises the following commands:

```
ISREDIT MACRO (FARG)
ISREDIT EXCLUDE ALL
ISREDIT F &FARG ALL
```

Example

1. Do an EXCLUDE FIND of all *_STC parameters in the WCONFIG(%RTE_NAME%) PARMGEN runtime environment profile.
2. Customize all *_STC started task parameters using the TSSI naming convention by changing all CANS references to TSSI as shown in the following example:

```
EDIT TSTEST.&userid.&rtname.WCONFIG(nte)
Command ===>C ' CANS' ' TSSI' ALL+ Scroll ===> CSR
```
Appendix E. Maintaining the persistent data store

If you intend to collect historical data, you must make provisions to manage the collected data. Without additional action, on distributed computers, the history data files grow unchecked, using up valuable disk space. On z/OS, the data sets allocated to store historical data are maintained by the persistent data store, which empties and reuses the data set with the oldest data.

About this task

The topics in this appendix explain how the persistent data store works and documents the maintenance procedure and options.

- “Overview of the persistent data store on z/OS”
- “Automatic maintenance process” on page 246
- “Making archived data available” on page 248
- “Exporting and restoring persistent data” on page 250
- “Format of exported data records” on page 251
- “Extracting persistent data store data to flat files” on page 255
- “Components of the persistent data store” on page 257
- “Operation of the persistent data store” on page 258
- “Command interface” on page 258

Overview of the persistent data store on z/OS

The persistent data store is used for writing and retrieving historical data. The data written to the persistent data store is organized by tables (attribute groups), groups, and data sets. Each table is assigned to a group. Multiple tables can be assigned to each group, and each group can have one or more data sets assigned to it. Typically, three data sets are assigned to each group.

One of the data sets in each collection group is kept empty so that it can be quickly switched to if the active data set fills up. After this switch is made, the persistent data store looks to see if there are any empty data sets in the group. If not, the automatic maintenance process empties the data set with the oldest data.

During configuration using the Configuration Tool you allocate space for the data sets and specify what is to be done with historical data that is sitting in a data set that is being emptied. The default is to delete the data. Alternatively, you can configure maintenance to back up the data to tape or DASD, export the data to an external program for processing, or extract the data and write it in a viewable format.

Note that if the Tivoli Data Warehouse is used, especially with a 1-hour warehousing interval, there is little or no advantage to invoking the BACKUP, EXPORT, or EXTRACT functions for data collected by OMEGAMON XE monitoring agents. These functions are more relevant if warehousing is not enabled, or if you are maintaining historical data for OMEGAMON agents whose data is not displayed by the Tivoli Enterprise Portal.

No historical data collection takes place until collection is configured and started through the History Collection Configuration window of the Tivoli Enterprise Portal. In the window, you specify the attribute groups for which you want data to be collected, the interval for data collection, the location where you want the collected data to be stored (at the monitoring server or at the agent), and whether you want to store the data in the Tivoli Data Warehouse so it will be available for long-term history data reporting.
**Automatic maintenance process**

When a data set becomes full, the persistent data store selects an empty data set to make active. When the data set is active, the persistent data store checks to see if there are any more empty data sets. If there are no more empty data sets, maintenance is started on the oldest data set.

The maintenance process consists of two files that are generated and tailored by the Configuration Tool and invoked by the persistent data store.

- **KPDPROC1**
  KPDPROC1 is a procedure that is started by the automatic maintenance processing if one of the three maintenance functions is configured. Limited information is passed to this started task to drive a CLIST in a TSO environment. The Configuration Tool creates this file and puts it into the RKANSAMU library for each runtime environment that has a persistent data store component. This procedure must be copied to a system level procedure library so the command issued to start it can be found.

  The parameters passed to KPDPROC1 vary based on the version of the Configuration Tool and the persistent data store. This document assumes the latest version is installed. Three parameters are passed to the started task:
  - **HILEV**
    High-level qualifier for the runtime environment that configured this version of the persistent data store. It is obtained by extracting information from the DD statement that points to the persistent data store control files.
  - **LOWLEV**
    Low-level qualifier for the sample library. It currently contains the RKANSAMU field name.
  - **data set**
    Fully qualified name of the data set being maintained. It is possible to have a data set name that does not match the high-level qualifier specified in the first parameter.

- **KPDPROCC**
  KPDPROCC is the CLIST that is executed by the KPDPROC1 procedure. The CLIST has the task of obtaining all of the information required to perform the maintenance, saving the data, and initialize the data set so it can be used again. This procedure performs the following actions:
  - Backs up the data
  - Deletes the data set
  - Allocates a new data set with the same parameters as before
  - Makes the new data set available for reading and writing

Before launching the KPDPROC1 process, the persistent data store checks to see if either the BACKUP, EXPORT, or EXTRACT function has been specified. If neither function has been specified, then the data set is initialized in the persistent data store started task and KPDPROC1 is not executed.

The Configuration Tool allows you to pick the first seven characters of the maintenance procedure names. KPDPROC is the default. If you changed KPDPROC to some other than the default during configuration, the suffixes remain 1 and C.

The persistent data store procedure runs with the user ID of the person who installed the product.

**The part of maintenance you control**

Most of the persistent data store maintenance procedure is automatic and does not require your attention. However, there are a few options you must decide upon.

Using the Configuration Tool, you specify the EXTRACT, BACKUP and EXPORT options by indicating a Y or N for each data set group. See “Command interface” on page 258 for descriptions of additional commands that are used primarily for maintenance.
• BACKUP makes an exact copy of the data set being maintained. You can indicate one of these backup options:
  – Back up the data for each data set group.
  – Back up the data to tape or to DASD for all data set groups.
• EXPORT writes the data to a flat file in an internal format that can be used by external programs to post process the data. This file is also used for recovery purposes when the persistent data store detects potential problems with the data.
• EXTRACT writes the data to a flat file in human-readable form, which is suitable for loading into other DBMS systems. The EXTRACT option is not strictly considered a way to maintain the data. If EXTRACT is specified, the data is pulled out and externalized in EBCDIC form, but the extraction does not empty the data set. If EXTRACT is specified, the data is extracted immediately from a data set that is switched from an active state to a full state. The data set is not initialized until it becomes a candidate for being emptied, so the most current data becomes available in an extracted form without that data being lost.

If none of the options is specified, the data in the data set being maintained is erased.

**Indicating data set backup to tape or to DASD**
For all data set groups that you selected to back up, you must indicate whether you want to back up the data to tape or to DASD. This decision applies to all data sets.

**About this task**

Tip: If you edit the KPDPROCC member of rhilev.rte.RKANSAM and then upgrade or reconfigure the runtime environment, your modifications might be overwritten.

**Procedure**
Complete one of the following steps to indicate where data should be backed up.
• If you are backing up data sets to tape, use KPDPROCC as included in the product.
• If you are backing up data sets to DASD, follow this procedure to modify KPDPROCC:
  1. With any editor, open the procedure in the rhilev.rte.RKANSAM(KPDPROCC) member.
  2. Find the statement
     
     TAPE = N

     and change the N to Y.
  3. Save the procedure.
  4. Copy procedure KPDPROCC to your system procedure library.

**Names of exported data sets**
If you choose to export data, you are requesting that the data be written to a sequential data set. The names of all exported data sets follow a specific format.

**About this task**

The names of all exported data sets follow this format:

```
rhilev.rte.dsnlolev.Annnnnnn
```

where:
• *rhilev* is the high-level qualifier of all data sets in the persistent data store, as you specified in the Configuration Tool
Making archived data available

After data has been backed up to DASD or to tape, it must be made available to those products that use it.

About this task

When the automatic maintenance facility backs up a data set in the persistent data store, it performs the following actions:

- Disconnects the data set from the monitoring server
- Copies the data set to tape or DASD in a format readable by the monitoring server
- Deletes and reallocates the data set
- Reconnects the empty data set to the monitoring server

To view archived data from the product, you must ensure that the data set is stored on an accessible DASD volume and reconnect the data set to the monitoring server. The following topic provides instructions for reconnecting the data set.

Data set naming conventions

The maintenance facility uses the following format to name the data sets it creates when it backs up data:

rhilev.rte.dsnlolev.Bnnnnnnn

where:

- **rhilev** is the high-level qualifier of all data sets in the persistent data store, as you specified during configuration
- **rte** is the mid-level qualifier of all data sets in the persistent data store, as you specified during configuration
- **dsnlolev** is the low-level qualifier of the data set names as set by the Configuration Tool
- **B** is a required character
- **nnnnnnn** is a sequential number

Prerequisites

About this task

Before you begin to restore the connection between the archived data set and the monitoring server, find the following information:

- The name of the archived data set that contains the data you want to view. Your system programmer can help you locate the name of the data set.
- The name of the persistent data store group that corresponds to the data you want to view.

Finding background information

You can use the Configuration Tool to find the name of the persistent data store group to which the archived data set belongs.
About this task

Using the following procedure to determine the name of the group to which the data set belongs.

Procedure
1. Stop the Tivoli Enterprise Monitoring Server if it is running.
2. Log on to a TSO session and invoke ISPF.
3. At the ISPF Primary Option menu, enter 6 in the Option field to access the TSO command mode.
4. At the TSO command prompt, enter this command to start the Configuration Tool:
   \texttt{EX \textasciitilde{shilev},INSTLIB}
5. From the Main Menu, select Configure products and then Select product to configure.
6. From the Product Selection Menu, select the product.
7. On the Runtime Environments (RTEs) panel, specify C to configure the runtime environment.
8. On the Configure product panel, select Configure persistent data store and then Modify and review data store specifications.
9. Locate the low-level qualifier of the data set you want to reconnect and note the corresponding group name.
10. Press F3 until you exit the Configuration Tool.

Connecting the data set to the monitoring server

To view archived historical data from the product, the archived data set must be connected to the monitoring server to which the monitoring agent connects.

About this task

Use the following procedure to connect the archived data set to the monitoring server.

Procedure
1. If the data set is located on tape, use a utility such as IEBGENR to copy the data set to a DASD volume that is accessible by the monitoring server.
2. Copy job KPDCOMMJ from \textit{thilev},TKANSAM to \textit{rhilev},RTE.RKANSAMU.
3. Access job \textit{rhilev},RTE.RKANSAMU(KPDCOMMJ) with any editor.
4. Substitute site-specific values for the variables in the job, as described in the comments at the beginning of the job.
5. Locate the COMM ADDFILE statement near the bottom of the job and remove the comment character (*).
6. Submit KPDCOMMJ to restore the connection between the data set you specified and the monitoring server.
7. To verify that the job ran successfully, you can view a report in RKPDLLOG that lists all the persistent data store data sets that are connected to the monitoring server. RKPDLLOG is the dname of a SYSOUT file allocated to the monitoring server. Locate the last ADDFILE statement in the log and examine the list of data sets that follows the statement. If the job ran successfully, the name of the data set you reconnected is in the list.

Disconnecting the data set

The data set that you connected to the monitoring server is not permanently connected. The connection is removed automatically the next time the monitoring server stops. You can also remove the data set from the persistent data store immediately after you view the data.
About this task

To disconnect the data set after viewing the data, use the following procedure.

Procedure

1. Access job \texttt{rhilev.rte.RKANSAMU(KPDCOMMJ)} with any editor.
2. Retain all site-specific values that you entered when you modified the job to reconnect the data set in the previous procedure.
3. Locate the COMM ADDFILE statement near the bottom of the job and perform the following steps:
   a. Remove the comment character from the statement, if one exists.
   b. Change the word ADDFILE to the word DELFILE.
   c. Remove the Group parameter together with its value.
   d. Remove the RO parameter if it exists.
4. Submit KPDCOMMJ to remove the connection between the data set and the monitoring server.
   To verify that the job ran successfully, you can view a report in RKPDLOG that lists all data sets connected to the monitoring server. Locate the last DELFILE statement in the log and examine the list of data sets that follows the statement. If the job ran successfully, the name of the data set you disconnected is not included in the list.
5. If the data set is located on tape, you might want to conserve space by deleting the data set from DASD.

Exporting and restoring persistent data

In addition to the standard maintenance jobs used by the persistent data store, there are sample jobs distributed with the monitoring server that you can use to export data to a sequential file and then restore the data to the original indexed format.

About this task

These jobs are not tailored by the Configuration Tool at installation time and must be modified to add pertinent information.

Exporting persistent data

About this task

Follow this procedure to export persistent data to a sequential file:

Procedure

1. Stop the monitoring server if it is running.
2. Copy \texttt{rhilev.rte.RKANSAMU(KPDEXPTJ)}.
3. Update the JCL with the following values:
   \begin{itemize}
   \item \texttt{rhilev} high-level qualifier of the runtime environment where the persistent data store resides.
   \item \texttt{pdsn} fully qualified name of the persistent data store data set to be exported
   \item \texttt{expdsn} fully qualified name of the export file you are creating
   \item \texttt{unit2} DASD unit identifier for \texttt{expdsn}
   \item \texttt{ssz} record length of output file (You can use the same record length as defined for \texttt{pdsn}.)
   \item \texttt{sct} count of blocks to allocate (You can use the same size as the blocks allocated for \texttt{pdsn}.)
   \end{itemize}
bsz &ssz value plus eight

With the exception of &pdsn, these values can be found in the PDSLOG SYSOUT of the monitoring server started task.

4. Submit the job.

**Restoring exported data**

**About this task**

Follow this procedure to restore a previously exported persistent data store data set.

**Procedure**

1. Copy rhilev.rte.RKANSAMU(KPDRESTJ).
2. Update the JCL with the following values:
   - rhilev high-level qualifier of the runtime environment where the persistent data store resides
   - pdsn fully qualified name of the persistent data store data set to be restored
   - expdsn fully qualified name of the file you are creating
   - unit2 DASD unit identifier for expdsn
   - group identifier for the group that the data set will belong to
   - siz size of the data set to be allocated, in megabytes

   With the exception of pdsn, these values can be found in the PDSLOG SYSOUT of the monitoring server started task.
3. Submit the job.

**Format of exported data records**

Exported data records share a common format, but the content of the tables and columns is product-specific. The topics in this section describe the format of the dictionary entries.

A single dictionary data set contains a description and mapping information for every table recorded in the original data set. In many cases, the tables have variable-length columns and some rows of data where columns are not available. The information about missing columns and lengths for variable columns is embedded in the data records. Some tables have columns that physically overlay each other. This must be taken into account when trying to obtain data for these overlays.

Data in the exported file is kept in internal format, and many of the fields are binary. The output file is made up of three sections with one or more data rows in each.

- Section 1 describes general information about the data source used to create the exported data.
- Section 2 contains a dictionary required for mapping the data.
- Section 3 contains the actual data rows.

The historical data is maintained in relational tables. Therefore, the dictionary mappings provide table and column information for every table that had data recorded for it in the persistent data store.

**Section 2 records**

Section 2 provides information about the tables and columns that are represented in Section 3. This section has a header record followed by a number of table and column description records.

**Dictionary header record:**
The dictionary header is the first Section 2 record (and therefore the second record in the data set). It provides general information about the format of the dictionary records that follow. It is used to describe how many tables are defined in the dictionary section.

The data layout for the dictionary header record follows.

**Table 15. Section 2 data record format, exported persistent data**

<table>
<thead>
<tr>
<th>Field</th>
<th>Offset</th>
<th>Length</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RecID</td>
<td>0</td>
<td>4</td>
<td>Char</td>
<td>Record ID. Contains DD10 for header record 2.</td>
</tr>
<tr>
<td>Dictionary Len</td>
<td>4</td>
<td>4</td>
<td>Binary</td>
<td>Contains the length of the entire dictionary.</td>
</tr>
<tr>
<td>Header Len</td>
<td>4</td>
<td>4</td>
<td>Binary</td>
<td>Length of the header record.</td>
</tr>
<tr>
<td>Table Count</td>
<td>8</td>
<td>4</td>
<td>Binary</td>
<td>Number of tables in dictionary (1 record per table).</td>
</tr>
<tr>
<td>Column Count</td>
<td>12</td>
<td>4</td>
<td>Binary</td>
<td>Total number of columns described.</td>
</tr>
<tr>
<td>Table Row Len</td>
<td>16</td>
<td>4</td>
<td>Binary</td>
<td>Size of table row.</td>
</tr>
<tr>
<td>Col Row Len</td>
<td>20</td>
<td>4</td>
<td>Binary</td>
<td>Size of column row.</td>
</tr>
<tr>
<td>Expansion</td>
<td>24</td>
<td>28</td>
<td>---</td>
<td>Unused area.</td>
</tr>
</tbody>
</table>

**Table description record:**

Each table in the exported data set has a table record that provides its name and identifier, and additional information about the columns. All table records are provided before the first column record. The column records and all of the data records in section 3 use the identifier number to associate it with the appropriate table.

The map length and variable column count fields can be used to determine exactly where the data for each column starts and to determine if the column exists in a record. The format of the table description record is described in the table that follows.

**Table 16. Section 2 table description record, exported persistent data**

<table>
<thead>
<tr>
<th>Field</th>
<th>Offset</th>
<th>Length</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RecID</td>
<td>0</td>
<td>4</td>
<td>Char</td>
<td>Record ID. Contains DD20 for table record.</td>
</tr>
<tr>
<td>Identifier Num</td>
<td>4</td>
<td>4</td>
<td>Binary</td>
<td>Unique number for this table.</td>
</tr>
<tr>
<td>Application</td>
<td>8</td>
<td>8</td>
<td>Char</td>
<td>Application name table belongs to.</td>
</tr>
<tr>
<td>Table Name</td>
<td>16</td>
<td>10</td>
<td>Char</td>
<td>Table name.</td>
</tr>
<tr>
<td>Table Version</td>
<td>26</td>
<td>8</td>
<td>Char</td>
<td>Table version.</td>
</tr>
<tr>
<td>Map Length</td>
<td>34</td>
<td>2</td>
<td>Binary</td>
<td>Length of the mapping area.</td>
</tr>
<tr>
<td>Column Count</td>
<td>16</td>
<td>4</td>
<td>Binary</td>
<td>Count of columns in the table.</td>
</tr>
<tr>
<td>Variable Cols</td>
<td>36</td>
<td>4</td>
<td>Binary</td>
<td>Count of variable name columns.</td>
</tr>
</tbody>
</table>
Table 16. Section 2 table description record, exported persistent data (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Offset</th>
<th>Length</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Row Count</td>
<td>40</td>
<td>4</td>
<td>Binary</td>
<td>Number of rows in exported file for this table.</td>
</tr>
<tr>
<td>Oldest Row</td>
<td>44</td>
<td>16</td>
<td>Char</td>
<td>Timestamp for oldest row written for this table.</td>
</tr>
<tr>
<td>Newest Row</td>
<td>64</td>
<td>16</td>
<td>Char</td>
<td>Timestamp for newest row written for this table.</td>
</tr>
<tr>
<td>Expansion</td>
<td>80</td>
<td>16</td>
<td>---</td>
<td>Unused area.</td>
</tr>
</tbody>
</table>

Column description record:

One record exists for every column in the associated table record. Each record provides the column name, type, and other characteristics. The order of the column rows is the same order in which the columns are displayed in the output row. However, some columns might be missing on any given row. Use the mapping structure defined under section 3 to determine whether a column is present.

The format of the column records is described in the following table.

Table 17. Section 2 column description record, exported persistent data

<table>
<thead>
<tr>
<th>Field</th>
<th>Offset</th>
<th>Length</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RecID</td>
<td>0</td>
<td>4</td>
<td>Char</td>
<td>Record ID. Contains DD30 for table record.</td>
</tr>
<tr>
<td>Table Ident</td>
<td>4</td>
<td>4</td>
<td>Binary</td>
<td>Identifier for the table this column belongs to.</td>
</tr>
<tr>
<td>Column Name</td>
<td>8</td>
<td>10</td>
<td>Char</td>
<td>Column name.</td>
</tr>
<tr>
<td>SQL Type</td>
<td>18</td>
<td>2</td>
<td>Char</td>
<td>SQL type for column.</td>
</tr>
<tr>
<td>Column Length</td>
<td>20</td>
<td>4</td>
<td>Binary</td>
<td>Maximum length of this column’s data.</td>
</tr>
<tr>
<td>Flag</td>
<td>24</td>
<td>1</td>
<td>Binary</td>
<td>Flag byte.</td>
</tr>
<tr>
<td>Spare</td>
<td>25</td>
<td>1</td>
<td>---</td>
<td>Unused.</td>
</tr>
<tr>
<td>Overlay Col ID</td>
<td>26</td>
<td>2</td>
<td>Char</td>
<td>Column number if this is an overlay.</td>
</tr>
<tr>
<td>Overlay Col Off</td>
<td>28</td>
<td>2</td>
<td>Char</td>
<td>Offset into row for start of overlay column.</td>
</tr>
<tr>
<td>Alignment</td>
<td>30</td>
<td>2</td>
<td>---</td>
<td>Unused.</td>
</tr>
<tr>
<td>Spare 1</td>
<td>32</td>
<td>8</td>
<td>---</td>
<td>Unused.</td>
</tr>
</tbody>
</table>

Section 1 record

The Section 1 record is not required for mapping the data in the exported file. However, it is useful for determining how to reallocate a data set when a persistent data store file must be reconstructed.

Section 1 contains a single data row used to describe information about the source of the data recorded in the export file. The data layout for the record follows.
Table 18. Section 1 data record format, exported persistent data

<table>
<thead>
<tr>
<th>Field</th>
<th>Offset</th>
<th>Length</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RecID</td>
<td>0</td>
<td>4</td>
<td>Char</td>
<td>Record ID. Contains AA10 for header record 1.</td>
</tr>
<tr>
<td>Length</td>
<td>4</td>
<td>4</td>
<td>Binary</td>
<td>Contains the record length of the header record.</td>
</tr>
<tr>
<td>Timestamp</td>
<td>8</td>
<td>16</td>
<td>Char</td>
<td>Timestamp of export. Format: CYYMMDDHHMMSSMMMM</td>
</tr>
<tr>
<td>Group</td>
<td>24</td>
<td>8</td>
<td>Char</td>
<td>Group name to which the data belongs.</td>
</tr>
<tr>
<td>Data Store Ver</td>
<td>32</td>
<td>8</td>
<td>Char</td>
<td>Version of KPDMANE used to record original data.</td>
</tr>
<tr>
<td>Export Version</td>
<td>40</td>
<td>8</td>
<td>Char</td>
<td>Version of KPDARCH used to create exported file.</td>
</tr>
<tr>
<td>Total Slots</td>
<td>48</td>
<td>4</td>
<td>Binary</td>
<td>Number of blocks allocated in original data set.</td>
</tr>
<tr>
<td>Used Slots</td>
<td>52</td>
<td>4</td>
<td>Binary</td>
<td>Number of used blocks at time of export.</td>
</tr>
<tr>
<td>Slot Size</td>
<td>56</td>
<td>4</td>
<td>Binary</td>
<td>Block size of original data set.</td>
</tr>
<tr>
<td>Expansion Area</td>
<td>60</td>
<td>20</td>
<td>---</td>
<td>Unused area.</td>
</tr>
<tr>
<td>Data Store Path</td>
<td>80</td>
<td>256</td>
<td>Char</td>
<td>Name of originating data set.</td>
</tr>
<tr>
<td>Export Path</td>
<td>336</td>
<td>256</td>
<td>Char</td>
<td>Name of exported data set.</td>
</tr>
</tbody>
</table>

Section 3 records

Section 3 has one record for every row of every table that was in the original persistent data store data set being exported. Each row starts with a fixed portion, followed by the actual data associated with the row. The length of the column map can be obtained from the table record (DD20). Each bit in the map represents one column. A 0 for the bit position indicates that the column data is not present, while a 1 indicates that data exists in this row for the column. Immediately following the column map field is an unaligned set of 2-byte-length fields. One of these length fields exists for every variable-length column in the table. This mapping information must be used to find the starting location for any given column in the data structure. The actual data starts immediately after the last length field.

If you are dealing with overlay columns, use the column offset defined in the DD30 records to determine the starting location for this type of column. Do not worry about overlaid columns with extracted data. If you want to look at the actual content of an overlaid column, you can expand the data by reinserting any missing columns and expanding all variable length columns to the maximum length before doing the mapping.

The table that follows maps the fixed portion of the data.

Table 19. Section 3 record format, exported persistent data

<table>
<thead>
<tr>
<th>Field</th>
<th>Offset</th>
<th>Length</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RecID</td>
<td>0</td>
<td>4</td>
<td>Char</td>
<td>Record ID. Contains ROW1 for column record.</td>
</tr>
<tr>
<td>Table Ident</td>
<td>4</td>
<td>4</td>
<td>Binary</td>
<td>Identifier for the table this record belongs to.</td>
</tr>
</tbody>
</table>
Table 19. Section 3 record format, exported persistent data (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Offset</th>
<th>Length</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Row Length</td>
<td>8</td>
<td>4</td>
<td>Binary</td>
<td>Total length of this row.</td>
</tr>
<tr>
<td>Data Offset</td>
<td>12</td>
<td>4</td>
<td>Binary</td>
<td>Offset to start of data.</td>
</tr>
<tr>
<td>Data Length</td>
<td>16</td>
<td>4</td>
<td>Binary</td>
<td>Length of data portion of row.</td>
</tr>
<tr>
<td>Column Map</td>
<td>20</td>
<td>Varies</td>
<td>Binary</td>
<td>Column available map plus variable length fields.</td>
</tr>
</tbody>
</table>

Extracting persistent data store data to flat files

Historical data can be exported to a flat file in EBCDIC format. The data can then be loaded into spreadsheets or databases for reporting and analysis.

About this task

The format of the data is converted to tab-delimited columns. The data is written to separate files for each table; therefore, the data format for all rows in each data set is consistent. The program also generates a separate file. This file contains a single row that provides the column names in the order in which the data is organized. This file is also delimited for ease of use. An option (NOFF) on the KPDXTRA program bypasses creating the separate file and places the column information as the first record of the data file.

This job is not tailored by the Configuration Tool at installation time and must be modified to add pertinent information.

The output from this job is written to files with the following naming standard:

&hilev.&rtetablename.D
&hilev.&rte tablename.H

The *tablename* is the first eight bytes of the actual table with any invalid characters changed to "#". The "D" suffix is for the data file and the "H" suffix is for the header file. Note that the NOFF parameter can be passed to the KPDXTRA program to merge the data from the header file into the data file and therefore not create a unique header file. The high level and mid level qualifiers are passed as a parameter to the KPDXTRA program (PREF=xxxx) and are therefore controlled by the KPDPROCC member in the RKANSAMU data set.

If this job is run more than once on a given day, data is appended to any data previously extracted for that day.

All data sets are kept in read/write state even if they are not active. This makes the data sets unavailable if the monitoring server is running. Jobs cannot be run against the active data sets and the inactive data sets must be taken offline.

You can remove a data set from the monitoring server by issuing the modify command:

```
F stcname,KPDCMD QUIESCE FILE=DSN:dataset
```

If you must run a utility program against an active data store, issue a SWITCH command prior to issuing this QUIESCE command.
Extracting persistent data store data to EBCDIC files

About this task

Use this job to extract persistent data store data to EBCDIC files.

1. Copy `rhilev:RTE.RKANSAMU(KPDXTRAJ)`.

2. Update the JCL with the following values:

   - `rhilev`  
     High-level qualifier of the runtime environment where the persistent data store resides.
   - `pdsn`  
     Fully qualified name of the persistent data store data set to be extracted
   - `pref`  
     High-level qualifier for the extracted data

3. Add the parameters you want to use for this job

   - `PREF=`  
     Identifies the high-level qualifier for the output file. This field is required.
   - `DELIM=`  
     Identifies the separator character to be placed between columns. The default is 05.
   - `NOFF=`  
     If used, causes the format file not to be generated. The column names are placed into the data file as the first record.
   - `QUOTES`  
     Places quotation marks around character type of data

4. Submit the job.

Format of extracted header and data file records

Header record

The following example shows an extracted header file record:

```
TMZDIFF(int,0,4) WRITETIME(char,1,16) ORIGINNODE(char,2,128) QMNAME(char,3,48) APPLID(char,4,12) APPLTYPE(int,5,4) SDATE_TIME(char,6,16) HOST_NAME(char,7,48) CNTTRANPGM(int,8,4) MSGSPUT(int,9,4) MSGSREAD(int,10,4) MSGSOUTSIZE(int,11,4) SIZEAVG(int,12,4) AVGTIME(int,13,4) SOCKETTIME(int,14,4) SOCKETOPTTIME(int,15,4) AVGIOFSIZE(int,16,4) AVGQSIZE(int,17,4) AVGQPTIME(int,18,4) AVGMQTIME(int,19,4) AVGMQGTIME(int,20,4) DEFSTATE(int,21,4) INT_TIME(int,22,4) INT_TIMEC(char,23,8) CNTTASKID(int,24,4) INTERVAL(int,25,4)
```

Each field is separated by a tab character (by default). The data consists of the column name with a type, column number, and column length field within the parentheses for each column. The information within parentheses is used primarily to describe the internal formatting information, and therefore can be ignored.

Data record

The following example shows an extracted data file record for the preceding header file record:

```
0 "1000104003057000" "MQM7:SYSG:MQESA" "MQM7" "XCXS2OPL" 2 "1000104003057434" "SYSG" 1 0 0 0 0 2 90007 0 2 0 1 96056 "016: 01" 1 1 900
```

The header file and the data file match up as follows:

```
Table 20. Extracted data

<table>
<thead>
<tr>
<th>Field name</th>
<th>Data</th>
<th>Data type</th>
</tr>
</thead>
<tbody>
<tr>
<td>TMZDIFF</td>
<td>0</td>
<td>Integer</td>
</tr>
<tr>
<td>WRITETIME</td>
<td>&quot;1000104003057000&quot;</td>
<td>&quot;Character&quot;</td>
</tr>
<tr>
<td>ORIGINNODE</td>
<td>&quot;MQM7:SYSG:MQESA&quot;</td>
<td>&quot;Character&quot;</td>
</tr>
</tbody>
</table>
```
Components of the persistent data store

The persistent data store consists of a set of components used for writing and retrieving historical data.

KPDMANE

KPDMANE is the primary executable program. It is a server for other applications running in the same address space. This program is designed to run inside the TMS:Engine address space as a separate subtask. Although it is capable of running inside the TMS:Engine, the program does not make any use of TMS:Engine services. This is because the KPDMANE program is also used in other utility programs that are intended to run in batch mode. KPDMANE is the program that eventually starts the maintenance task when it does a switch and determines that no empty data sets are available.

KPDUUTIL

This program is used primarily to initialize one or more data sets for persistent data store use. The program attaches a subtask and starts the KPDMANE program in it. The DD statements used when this program is run dictate what control files are executed by the KPDMANE program.

KPDARCH

This client program pulls data from the specified data set and writes it out to a flat file. The program attaches a subtask and starts up the KPDMANE program in it. The output data is still in an internal format, with index information excluded.

KPDREST

This client program reads data created by the KPDARCH program, inserts it into a data set in the correct format for use by the persistent data store, and rebuilds the index information. The program attaches a subtask and starts the KPDMANE program in it.

KPDXTRA

This client program pulls data from a data set and writes it to one or more flat files, with all column data converted to EBCDIC and separated by tabs. This extracted data can be loaded into a database management system or into spreadsheet programs such as Excel. The program attaches a subtask and starts the KPDMANE program in it.

KPDDSCO

This program communicates with the started task that is running the persistent data store and sends it commands. The typical command is RESUME, which tells the persistent data store that it can resume using a data set. The program can use two forms of communication:
- SNA protocol to connect to the monitoring server and submit command requests.
- SVC 34 to issue a modify command to the started task.

This program also logs information in a general log maintained in the persistent data store tables.

### Table 20. Extracted data (continued)

<table>
<thead>
<tr>
<th>Field name</th>
<th>Data</th>
<th>Data type</th>
</tr>
</thead>
<tbody>
<tr>
<td>QMNAME</td>
<td>&quot;MQM7&quot;</td>
<td>&quot;Character&quot;</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>SAMPLES</td>
<td>1</td>
<td>Integer</td>
</tr>
<tr>
<td>INTERVAL</td>
<td>900</td>
<td>Integer</td>
</tr>
</tbody>
</table>
Operation of the persistent data store

Maintenance is invoked automatically by the KPDMANE program.

The KPDMANE program invokes maintenance in two places:

- On startup, while reading and processing data sets, KPDMANE examines internal data to determine whether each data set is in a known and stable state. If not, KPDMANE issues a RECOVER command.
- When KPDMANE is recording information from applications onto an active data set for a group, if it detects that it is running out of room, it executes the SWITCH command internally.

Command interface

The persistent data store uses a command interface to perform many of the tasks required for maintaining the data sets used for historical data. Most of these commands can be invoked externally through a command interface supported in the Engine environment.

These maintenance commands can be executed by using the standard z/OS MODIFY interface with the following format:

```
F stcname,KPDCMD command arguments
```

where

- `stcname` Is the started task name of address space where the persistent data store is running.
- `command` Is one of the supported dynamic commands.
- `arguments` Are valid arguments to the specified command.

Maintenance commands

Persistent data store supports a number of commands used primarily for maintenance.

SWITCH command

This dynamic command causes a data store file switch for a specific file group. At any given time, update-type operations against tables in a particular group are directed to one and only one of the files in the group. That one file is called the active file. A file switch changes the active file for a group. In other words, the switch causes a file other than the currently active one to become the new active file.

If the group specified by this command has only one file, or the group currently has no inactive file that is eligible for output, the switch is not performed.

At the conclusion of a switch, persistent data store starts the maintenance process for a file in the group if no empty files remain in the group.

The [NO]EXTRACT keyword can be used to force or suppress an extract job for the data store file deactivated by the switch.

Syntax

```
SWITCH GROUP=groupid [ EXTRACT | NOEXTRACT ]
```

where

- `groupid` Specifies the ID of the file group that is to be switched. The group must have multiple files assigned to it.
EXTRACT
Specifies that the deactivated data store file is to be extracted, even if the file's GROUP statement did not request extraction.

NOEXTRACT
Specifies that extraction is not to be performed for the deactivated data store file. This option overrides the EXTRACT keyword of the GROUP statement.

Note that if neither EXTRACT nor NOEXTRACT is specified, the presence or absence of the EXTRACT keyword on the file's GROUP statement determines whether extraction is performed as part of the switch.

SWITCH logic
The SWITCH command looks at all data sets assigned to the group and attempts to find an empty one. If no empty data sets are available, future attempts to write data to any data set in the group will fail. Normally, an empty data set is found and is marked as the active data set.

When a data set is deactivated because it is full, it is tested to see whether the EXTRACT option was specified. If so, the EXTRACT command for the data set is executed.

The next test is to check whether there are any empty data sets in the current group. If not, the code finds the data set with the oldest data and marks it for maintenance. With the latest release of the persistent data store, the code checks to see whether any of the maintenance options BACKUP, EXPORT, or EXTRACT were specified for this data set. If not, the INITDS command is executed. Otherwise, the BACKUP command is executed.

BACKUP command
This command causes a maintenance task to be started for the data store file named on the command. The maintenance task typically deletes, allocates and initializes a data store file, optionally backing up or exporting the file before deleting it. (The optional export and backup steps are requested through parameters on the data store file's GROUP command in the RKPDIN file.)

Syntax
BACKUP FILE=DSN:dsname

where

    dsname
    Specifies the physical data set name of the file that is to be maintained.

BACKUP logic
This code quiences and closes the data set. A test is made to see whether either BACKUP or EXPORT is specified for the data set and appropriate options are set for the started task. The options always include a request to initialize the data set. An SVC 34 is issued to start the KPDPROC1 procedure. The code returns to the caller, and the data set is unavailable until the RESUME command is executed.

ADDFILE command
The ADDFILE command is used to dynamically assign a new physical data store file to an existing file group. The command can be issued any time after the persistent data store initialization has completed in the Tivoli Enterprise Monitoring Server. It can be used to increase the number of files assigned to a group or to bring old data back online. It cannot, however, be used to define a new file group ID. It can be used to add files only to groups that already exist as the result of GROUP commands in the RKPDIN input file.
Syntax

ADDFILE GROUP=groupid FILE=DSN:dsname [ RO ] [ BACKUP ] [ ARCHIVE ]

where

groupid
  Specifies the unique group ID of the file group to which a file is to be added.

dsnname
  Specifies the fully qualified name (no quotation marks) of the physical data set that is to be
  added to the group specified by groupid.

RO
  Specifies that the file is to be read-only (that is, that no new data can be recorded to it). By
  default, files are not read-only (that is, they are modifiable). This parameter can also be specified
  as READONLY.

BACKUP
  Specifies that the file is to be copied to disk or tape before being reallocated by the automatic
  maintenance task. (Whether the copy is to disk or tape is a maintenance process customization
  option.) By default, files are not backed up during maintenance.

ARCHIVE
  Specifies that the file is to be exported before being reallocated by the automatic maintenance
  task. By default, files are not exported during maintenance.

DELFILE command

The DELFILE command is used to drop one physical data store file from a file group’s queue of files. It
  can be issued any time after persistent data store initialization has completed in the Tivoli Enterprise
  Monitoring Server.

The file to be dropped must be full, partially full, or empty; it cannot be the “active” (output) file for its
  group (if it is, the DELFILE command is rejected).

The DELFILE command is conceptually the opposite of the ADDFILE command, and is intended to be
  used to manually drop a file that was originally introduced by a GROUP or ADDFILE command. After a
  file has been dropped by DELFILE, it is no longer allocated to the Tivoli Enterprise Monitoring Server
  task and can be allocated by other tasks. Note that DELFILE does not physically delete a file or alter it in
  any way. To physically delete and uncatalog a file, use the REMOVE command.

Syntax

DELFILE FILE=DSN:dsname

where

dsnname
  Specifies the fully qualified (without quotation marks) name of the file that is to be dropped.

EXTRACT command

The EXTRACT command causes an extract job to be started for the data store file named on the
  command. The job converts the table data in the data store file to delimited text format in new files, and
  then signals the originating Tivoli Enterprise Monitoring Server to resume use of the data store file.

For each table extracted from the data store file, two new files are created. One file contains the converted
data and one file contains a record describing the format of each row in the first file.
Syntax

EXTRACT

FILE=DSN:dsname

where

dsname

Specifies the physical data set name of the file to have its data extracted.

EXTRACT logic

The logic for this command is similar to the BACKUP logic, except that the only option specified is an EXTRACT run with no initialization performed on the data set.

INITDS command

The INITDS command forces a data store file to be initialized in the address space where the persistent data store is running.

Syntax

INITDS

FILE DSN:dsname

where

dsname

Identifies the data set name of the data store file to be initialized.

RECOVER command

The RECOVER command causes a recovery task to be started for the data store file named on the command. The recovery task attempts to repair a corrupted data store file by exporting it, reallocating and initializing it, and restoring it. The restore operation rebuilds the index information, the data most likely to be corrupted in a damaged file. The recovery is not guaranteed to be successful, however; some severe forms of data corruption are unrecoverable.

Syntax

RECOVER

FILE=DSN:dsname

where

dsname

Specifies the physical name of the data set to be recovered.

RECOVER logic

This code quiesces the data set and closes the file. Information is set up to request an ARCHIVE, INIT, and RESTORE operation to be performed by the maintenance procedures. An SVC 34 is issued for a START command on KPDPROC1 (or its overridden name). The command exits to the caller, and the data set is unusable until a RESUME command is executed.

RESUME command

The RESUME command is used to notify the persistent data store that it can once again make use of the data set specified in the arguments. The file identified must be one that was taken offline by the backup, recover, or extract commands.

Syntax

RESUME

FILE=DSN:dsname

where
Specifies the physical name of the data set to be brought online.

**RESUME logic**

This code opens the specified data set name and verifies that it is valid. The data set is taken out of the quiesce state and made available for activation during the next SWITCH operation.

**Other useful commands**

The commands in this section can be used to obtain information about the persistent data store and the data sets in the store, and to flush buffered data to disk.

**QUERY CONNECT command**

The QUERY CONNECT command displays a list of applications and tables that are currently defined in the persistent data store. The output of this command shows the application names, table names, total number of rows recorded for each table, the group the table belongs to, and the current data set that the data is being written to.

**Syntax**

```
QUERY CONNECT <ACTIVE>
```

where

**ACTIVE**

Is an optional parameter that displays only those tables that are active. An active table is one that has been defined and assigned to an existing group, and the group has data sets assigned to it.

**QUERY DATASTORE command**

The QUERY DATASTORE command displays a list of data sets known to the persistent data store. For each data set, the total number of allocated blocks, the number of used blocks, the number of tables that have data recorded, the block size, and status are displayed.

**Syntax**

```
QUERY DATASTORE <FILE=DSN:datasetname>
```

where

**FILE**

Optional parameter that allows you to specify that you are only interested in the details for a single data set. When this option is used, the resulting display is changed to show information that is specific to the tables being recorded in the data set.

**COMMIT command**

This dynamic command flushes to disk all pending buffered data. For performance reasons, the persistent data store does not immediately write to disk every update to a persistent table. Updates are buffered in virtual storage. Eventually the buffered updates are flushed (written to disk) at an optimal time. However, this architecture makes it possible for persistent data store files to become corrupted (not valid) if the files are closed prematurely, before pending buffered updates have been flushed. Such premature closings might leave inconsistent information in the files. The COMMIT command is intended to limit the exposure to data store file corruption. Some applications automatically issue this command after inserting data.

The following known circumstances might cause corrupted files:

- Severe abnormal monitoring server terminations that prevent the persistent data store recovery routines from executing
- IPLs performed without first stopping the monitoring server
Syntax

COMMIT
Appendix F. Product codes

The following table lists many of the product codes associated with the Tivoli Management Services components and OMEGAMON XE products. Where a component or product has more than one code, the principal configuration code is shown in **bold print**. Throughout this publication, the variable *pp* represents the product code.

*Table 21. Two-character product codes*

<table>
<thead>
<tr>
<th>Product or component</th>
<th>Codes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration Tool (ICAT);PARMGEN</td>
<td>CI, CN, DZ, IN, KC, KI, PZ, RT, Z2</td>
</tr>
<tr>
<td>End-to-End response time component</td>
<td>ET</td>
</tr>
<tr>
<td>EPILOG</td>
<td>EB, EC, ED, EI, EO, EP</td>
</tr>
<tr>
<td>Monitoring agent on z/OS (common to all monitoring agents)</td>
<td>AG</td>
</tr>
<tr>
<td>NetView for z/OS</td>
<td>NA</td>
</tr>
<tr>
<td>OMNIMON base (includes the IBM Tivoli OMEGAMON enhanced 3270 user interface and OMEGAMON subsystem)</td>
<td>CA, CC, EB, IA, JI, OB</td>
</tr>
<tr>
<td>OMEGAMON XE for CICS on z/OS</td>
<td>BG, C2, C3, C5, CP, IA, IN, OC</td>
</tr>
<tr>
<td>OMEGAMON XE for CICS TG on z/OS</td>
<td>GW</td>
</tr>
<tr>
<td>OMEGAMON XE for IMS on z/OS</td>
<td>AT, DE, I2, I3, I5, IA, ID, IN, IP, ML, OI, OR, OS, RI</td>
</tr>
<tr>
<td>OMEGAMON XE for Mainframe Networks</td>
<td>N3, ON</td>
</tr>
<tr>
<td>OMEGAMON XE for Storage on z/OS</td>
<td>DE, RC, S3</td>
</tr>
<tr>
<td>OMEGAMON XE on z/OS</td>
<td>CG, CS, EA, IA, IN, M2, M3, M5, MH, MR, OE, OM, OS, PM, RA, WD, XD, XO</td>
</tr>
<tr>
<td>OMEGAMON XE on z/VM and Linux</td>
<td>VL</td>
</tr>
<tr>
<td>OMEGAMON z/OS Management Console</td>
<td>HL</td>
</tr>
<tr>
<td>OMEGAVIEW</td>
<td>MV, SD, WO</td>
</tr>
<tr>
<td>Summarization and Pruning agent</td>
<td>SY</td>
</tr>
<tr>
<td>Tivoli Enterprise Monitoring Server</td>
<td>AG, CO, DS, EF, FA, FW, GL, IB, IH, LC, LG, LI, LS, MA, MS, NS, NV, O4, OU, OX, PD, PS, PT, QM, SH, SM, SS, TN, TR, UI, UT, VI, VT</td>
</tr>
<tr>
<td>Tivoli Enterprise Portal desktop client</td>
<td>CJ</td>
</tr>
<tr>
<td>Tivoli Enterprise Portal Server</td>
<td>CQ, FW</td>
</tr>
<tr>
<td>Tivoli Enterprise Portal browser client</td>
<td>CW</td>
</tr>
<tr>
<td>TMS:Engine</td>
<td>BB, DC, DE, DF, DH, LB, LC, LD, LE, LF, LG, LH, LI, LS, LT, LU, LV, LX, SB</td>
</tr>
<tr>
<td>Warehouse Proxy agent</td>
<td>HD</td>
</tr>
</tbody>
</table>
Appendix G. Predefining and managing OMEGAMON started tasks

If you are using System Automation for z/OS, make sure the automation rules are defined to start and manage the OMEGAMON started tasks. System Automation comes with an OMEGAMON policy that predefines these rules.


The following table lists the OMEGAMON started tasks (default names have been used) and associated automation dependencies.

*Table 22. OMEGAMON started tasks and automation dependencies*

<table>
<thead>
<tr>
<th>PARMGEN Configuration Parameter</th>
<th>Default started task name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Dependency on VTAM</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VTAM must be running and the major node must be active.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>KM2_CLASSIC_STC</td>
<td>CANSM2RC</td>
<td>IBM Tivoli OMEGAMON XE on z/OS (3270)</td>
</tr>
<tr>
<td>KM2_CUA_STC</td>
<td>CANSM2</td>
<td>IBM Tivoli OMEGAMON XE on z/OS (3270)</td>
</tr>
<tr>
<td>KM2_CUA_STC</td>
<td>CANSOCx</td>
<td>IBM Tivoli OMEGAMON XE for CICS on z/OS (where n = 1 per 3270 pair)</td>
</tr>
<tr>
<td>KC2_CCnn_CLASSIC_STC</td>
<td>CANSOlx</td>
<td>IBM Tivoli OMEGAMON XE for IMS on z/OS (where n = 1 per 3270 pair)</td>
</tr>
<tr>
<td>K12_CUA_STC</td>
<td>CANSI2</td>
<td>IBM Tivoli OMEGAMON XE for IMS on z/OS (3270)</td>
</tr>
<tr>
<td>GBL_DB2_KD2_CLASSIC_STC</td>
<td>CANSO2</td>
<td>IBM Tivoli OMEGAMON XE for DB2 Performance Expert/Performance Monitor on z/OS (3270)</td>
</tr>
<tr>
<td>KOS_TEMS_STC</td>
<td>CANSDSST</td>
<td>Tivoli Enterprise Monitoring Server</td>
</tr>
<tr>
<td>RTE_CANSETE_STC</td>
<td>CANSETE</td>
<td>OMEGAMON End-to-End</td>
</tr>
<tr>
<td>KM2_EPILOG_COLLECTOR_STC</td>
<td>CANSM2HI</td>
<td>IBM Tivoli OMEGAMON XE on z/OS (3270) Historical Collector (Epilog)</td>
</tr>
<tr>
<td>KM2_EPILOG_ZOOM_STC</td>
<td>CANSM2EZ</td>
<td>IBM Tivoli OMEGAMON XE on z/OS (3270) Epilog Zoom</td>
</tr>
<tr>
<td>KDF_CUA_STC</td>
<td>CANSDF</td>
<td>IBM Tivoli OMEGAMON XE for Storage on z/OS (3270)</td>
</tr>
<tr>
<td>KMV_CUA_STC</td>
<td>CANSMV</td>
<td>IBM Tivoli OMEGAMON DE on z/OS OMEGAVIEW (Agent)</td>
</tr>
<tr>
<td>KON_CUA_STC</td>
<td>CANSON</td>
<td>IBM Tivoli OMEGAMON XE for Mainframe Networks on z/OS (3270)</td>
</tr>
<tr>
<td><strong>Dependency on TCP/IP</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TCP/IP must be started, ready to accept binds, and able to resolve DNS names; this automation usually requires OMPROUTE.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>KOS_TEMS_STC</td>
<td>CANSDSST</td>
<td>Tivoli Enterprise Monitoring Server</td>
</tr>
<tr>
<td>KM2_CUA_STC</td>
<td>CANSM2</td>
<td>IBM Tivoli OMEGAMON XE on z/OS (3270)</td>
</tr>
<tr>
<td>KC5_AGT_STC</td>
<td>CANSC5</td>
<td>IBM Tivoli OMEGAMON XE for CICS on z/OS (Agent)</td>
</tr>
</tbody>
</table>
Table 22. OMEGAMON started tasks and automation dependencies (continued)

<table>
<thead>
<tr>
<th>PARMGEN Configuration Parameter</th>
<th>Default started task name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>KD5_AGT_STC</td>
<td>CANSD5</td>
<td>IBM Tivoli OMEGAMON XE for DB2 Performance Expert/Performance Monitor on z/OS (Agent)</td>
</tr>
<tr>
<td>KI5_AGT_STC</td>
<td>CANSI5</td>
<td>IBM Tivoli OMEGAMON XE for IMS on z/OS (Agent)</td>
</tr>
<tr>
<td>KN3_AGT_STC</td>
<td>CANSN3</td>
<td>IBM Tivoli OMEGAMON XE for Mainframe Networks on z/OS (Agent)</td>
</tr>
<tr>
<td>KMQ_AGT_STC</td>
<td>CANSMQ</td>
<td>IBM Tivoli OMEGAMON XE for Messaging-WebSphere MQ Monitoring (Agent)</td>
</tr>
<tr>
<td>KMC_AGT_STC</td>
<td>CANSMC</td>
<td>IBM Tivoli OMEGAMON XE for Messaging-WebSphere MQ Configuration (Agent)</td>
</tr>
<tr>
<td>KQI_AGT_STC</td>
<td>CANSQI</td>
<td>IBM Tivoli OMEGAMON XE for Messaging-WebSphere Message Broker (Agent)</td>
</tr>
<tr>
<td>KYN_AGT_STC</td>
<td>CANSYN</td>
<td>ITCAM for Application Diagnostics on z/OS (Agent)</td>
</tr>
<tr>
<td>KOB_TOM_STC</td>
<td>CANSTOM</td>
<td>IBM Tivoli OMEGAMON enhanced 3270 user interface</td>
</tr>
<tr>
<td>KGW_AGT_STC</td>
<td>CANSGW</td>
<td>IBM Tivoli OMEGAMON XE for CICS TG on z/OS (Agent)</td>
</tr>
</tbody>
</table>

Dependency on Tivoli Enterprise Monitoring Server (CANSDSST)

These tasks retry if they do not succeed in connecting to CANSDSST. The dependency is optional.

<table>
<thead>
<tr>
<th>PARMGEN Configuration Parameter</th>
<th>Default started task name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>KC5_AGT_STC</td>
<td>CANSC5</td>
<td>IBM Tivoli OMEGAMON XE for CICS on z/OS (Agent)</td>
</tr>
<tr>
<td>KD5_AGT_STC</td>
<td>CANSD5</td>
<td>IBM Tivoli OMEGAMON XE for DB2 Performance Expert/Performance Monitor on z/OS (Agent)</td>
</tr>
<tr>
<td>KI5_AGT_STC</td>
<td>CANSI5</td>
<td>IBM Tivoli OMEGAMON XE for IMS on z/OS (Agent)</td>
</tr>
<tr>
<td>KM2_EPILOG_COLLECTOR_STC</td>
<td>CANSM2HI</td>
<td>IBM Tivoli OMEGAMON XE on z/OS (3270) Historical Collector (Epilog)</td>
</tr>
<tr>
<td>KDF_CUA_STC</td>
<td>CANSDF</td>
<td>IBM Tivoli OMEGAMON XE for Storage on z/OS (3270)</td>
</tr>
<tr>
<td>KMQ_AGT_STC</td>
<td>CANSMQ</td>
<td>IBM Tivoli OMEGAMON XE for Messaging-WebSphere MQ Monitoring (Agent)</td>
</tr>
<tr>
<td>KMC_AGT_STC</td>
<td>CANSMC</td>
<td>IBM Tivoli OMEGAMON XE for Messaging-WebSphere MQ Configuration (Agent)</td>
</tr>
<tr>
<td>KQI_AGT_STC</td>
<td>CANSQI</td>
<td>IBM Tivoli OMEGAMON XE for Messaging-WebSphere Message Broker (Agent)</td>
</tr>
<tr>
<td>KYN_AGT_STC</td>
<td>CANSYN</td>
<td>ITCAM for Application Diagnostics on z/OS (Agent)</td>
</tr>
<tr>
<td>KM2_CUA_STC</td>
<td>CANSM2</td>
<td>IBM Tivoli OMEGAMON XE on z/OS (3270)</td>
</tr>
<tr>
<td>KN3_AGT_STC</td>
<td>CANSN3</td>
<td>IBM Tivoli OMEGAMON XE for Mainframe Networks on z/OS (Agent)</td>
</tr>
<tr>
<td>KOB_TOM_STC</td>
<td>CANSTOM</td>
<td>IBM Tivoli OMEGAMON enhanced 3270 user interface</td>
</tr>
<tr>
<td>KGW_AGT_STC</td>
<td>CANSGW</td>
<td>IBM Tivoli OMEGAMON XE for CICS TG on z/OS (Agent)</td>
</tr>
</tbody>
</table>

No dependencies

<table>
<thead>
<tr>
<th>PARMGEN Configuration Parameter</th>
<th>Default started task name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RTE_CANSCN_STC</td>
<td>CANSCN</td>
<td>OMEGAMON Subsystem</td>
</tr>
<tr>
<td>KM2_CSA_ANALYZER_STC</td>
<td>CANSM2CS</td>
<td>IBM Tivoli OMEGAMON XE on z/OS (3270) Common Storage Area Analyzer</td>
</tr>
</tbody>
</table>

Note: CANSOCx must come up before the CICS regions, or the OMEG INIT fails.
Workload Manager (WLM) settings for the OMEGAMON tasks

The various started tasks created during the configuration process have varying levels of priority. Some started tasks, typically the data collecting monitoring agents, require a high priority to enable the ability to make requests for data.

The following table lists the OMEGAMON started tasks (default names have been used) and indicates the priority level to use to allocate these tasks:

Table 23. OMEGAMON started tasks by priority group

<table>
<thead>
<tr>
<th>PARMGEN Configuration Parameter</th>
<th>Default started task name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Group 1: Collector tasks</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RTE_CANSN_STC</td>
<td>CANSCN</td>
<td>OMEGAMON Subsystem</td>
</tr>
<tr>
<td>GBL_DB2_KD2_CLASSIC_STC</td>
<td>CANSO2</td>
<td>IBM Tivoli OMEGAMON XE for DB2 Performance Expert/Performance Monitor on z/OS (3270)</td>
</tr>
<tr>
<td>KM2_CUA_STC</td>
<td>CANSOCl</td>
<td>IBM Tivoli OMEGAMON XE for CICS on z/OS (where n = 1 per 3270 pair)</td>
</tr>
<tr>
<td>KC2_CCnn_CLASSIC_STC</td>
<td>CANSOlx (one for each IMS subsystem)</td>
<td>IBM Tivoli OMEGAMON XE for IMS on z/OS (where n = 1 per 3270 pair)</td>
</tr>
<tr>
<td>KON_CUA_STC</td>
<td>CANSON</td>
<td>IBM Tivoli OMEGAMON XE for Mainframe Networks on z/OS (3270)</td>
</tr>
<tr>
<td>KM2_EPILOG_COLLECTOR_STC</td>
<td>CANSM2HI</td>
<td>IBM Tivoli OMEGAMON XE on z/OS (3270) Historical Collector (Epilog)</td>
</tr>
<tr>
<td>KM2_CLASSIC_STC</td>
<td>CANSM2RC</td>
<td>IBM Tivoli OMEGAMON XE on z/OS (3270)</td>
</tr>
<tr>
<td>KM2_CUA_STC</td>
<td>CANSM2</td>
<td>IBM Tivoli OMEGAMON XE on z/OS (3270)</td>
</tr>
<tr>
<td>KM2_CSA_ANALYZER_STC</td>
<td>CANSM2CS</td>
<td>IBM Tivoli OMEGAMON XE on z/OS (3270) Common Storage Area Analyzer</td>
</tr>
<tr>
<td>RTE_CANSETE_STC</td>
<td>CANSETE</td>
<td>OMEGAMON End-to-End</td>
</tr>
<tr>
<td>KDF_CUA_STC</td>
<td>CANSDF</td>
<td>IBM Tivoli OMEGAMON XE for Storage on z/OS (3270)</td>
</tr>
<tr>
<td>KC2_CCnn_CLASSIC_STC</td>
<td>CANSOlx (one for each IMS subsystem)</td>
<td>IBM Tivoli OMEGAMON XE for IMS on z/OS (where n = 1 per 3270 pair)</td>
</tr>
<tr>
<td><strong>Group 2: OMEGAMON XE tasks</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>KDS_TEMS_STC</td>
<td>CANSDDSST</td>
<td>Tivoli Enterprise Monitoring Server</td>
</tr>
<tr>
<td>KC5_AG'T_STC</td>
<td>CANSC5</td>
<td>IBM Tivoli OMEGAMON XE for CICS on z/OS (Agent)</td>
</tr>
<tr>
<td>KDS_AG'T_STC</td>
<td>CANSD5</td>
<td>IBM Tivoli OMEGAMON XE for DB2 Performance Expert/Performance Monitor on z/OS (Agent)</td>
</tr>
<tr>
<td>K15_AG'T_STC</td>
<td>CANSI5</td>
<td>IBM Tivoli OMEGAMON XE for IMS on z/OS (Agent)</td>
</tr>
<tr>
<td>KN3_AG'T_STC</td>
<td>CANSN3</td>
<td>IBM Tivoli OMEGAMON XE for Mainframe Networks on z/OS (Agent)</td>
</tr>
<tr>
<td>KMQ_AG'T_STC</td>
<td>CANSMQ</td>
<td>IBM Tivoli OMEGAMON XE for Messaging-WebSphere MQ Monitoring (Agent)</td>
</tr>
<tr>
<td>PARMGEN Configuration Parameter</td>
<td>Default started task name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>---------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>KQI_AGT_STC</td>
<td>CANSQI</td>
<td>IBM Tivoli OMEGAMON XE for Messaging-WebSphere Message Broker (Agent)</td>
</tr>
<tr>
<td>KYN_AGT_STC</td>
<td>CANSYN</td>
<td>ITCAM for Application Diagnostics on z/OS (Agent)</td>
</tr>
<tr>
<td>KMC_AGT_STC</td>
<td>CANSMC</td>
<td>IBM Tivoli OMEGAMON XE for Messaging-WebSphere MQ Configuration (Agent)</td>
</tr>
<tr>
<td>KOB_TOM_STC</td>
<td>CANSTOM</td>
<td>IBM Tivoli OMEGAMON enhanced 3270 user interface</td>
</tr>
<tr>
<td>KGW_AGT_STC</td>
<td>CANSGW</td>
<td>IBM Tivoli OMEGAMON XE for CICS TG on z/OS (Agent)</td>
</tr>
</tbody>
</table>

**Group 3: OMEGAMON II presentation tasks**

This group of tasks need lower priority because they display data but do not collect it.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Default started task name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>KI2_CUA_STC</td>
<td>CANSI2</td>
<td>IBM Tivoli OMEGAMON XE for IMS on z/OS (3270)</td>
</tr>
<tr>
<td>KC2_CCnn_CUA_STC</td>
<td>CANSC2n</td>
<td>IBM Tivoli OMEGAMON XE for CICS on z/OS (where n = 1 per 3270 pair)</td>
</tr>
<tr>
<td>KD2_CUA_STC</td>
<td>CANSD2</td>
<td>IBM Tivoli OMEGAMON XE for DB2 Performance Expert/Performance Monitor on z/OS (Agent)</td>
</tr>
<tr>
<td>KMV_CUA_STC</td>
<td>CANSMV</td>
<td>IBM Tivoli OMEGAMON DE on z/OS OMEGAVIEW (Agent)</td>
</tr>
<tr>
<td>KM2_HIST_DATA_INTERFACE_STC</td>
<td>CANSM2HD</td>
<td>IBM Tivoli OMEGAMON XE on z/OS (3270) Historical data interface</td>
</tr>
<tr>
<td>KM2_EPILOG_ZOOM_STC</td>
<td>CANSM2EZ</td>
<td>IBM Tivoli OMEGAMON XE on z/OS (3270) Epilog Zoom</td>
</tr>
</tbody>
</table>

**Group 4: Maintenance tasks for historical data sets**

This group of tasks can be low-priority because they are started only when needed and they run for a short time.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Default started task name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>KM2_HIST_PROC_PRIMARY_STC</td>
<td>CANSM2HP</td>
<td>IBM Tivoli OMEGAMON XE on z/OS (3270) Historical maintenance (primary)</td>
</tr>
<tr>
<td>KM2_HIST_PROC_SECONDARY_STC</td>
<td>CANSM2HS</td>
<td>IBM Tivoli OMEGAMON XE on z/OS (3270) Historical maintenance (secondary)</td>
</tr>
<tr>
<td>RTE_PDS_KPDPROC_PREFIX</td>
<td>KPDPROCn</td>
<td>Common IBM Tivoli Monitoring Persistent Datastore high-level qualifier prefix historical maintenance procedure (where n = 1)</td>
</tr>
<tr>
<td>KM2_HIST_BATCH_REPORTER_STC</td>
<td>CANSM2BA</td>
<td>IBM Tivoli OMEGAMON XE on z/OS (3270) Historical maintenance (batch reporter)</td>
</tr>
<tr>
<td>KDF_DFDSS_SLAVE_STC</td>
<td>KDFDSPR</td>
<td>IBM Tivoli OMEGAMON XE for Storage on z/OS (3270 DFDSS Slave)</td>
</tr>
</tbody>
</table>
Documentation library

This appendix contains information about the publications related to IBM Tivoli Monitoring and to the commonly shared components of Tivoli Management Services.

These publications are listed in the following categories:

- IBM Tivoli Monitoring library
- Related publications


To find a list of new and changed publications, click What's new on the Welcome page of the IBM Tivoli Monitoring and OMEGAMON XE Information Center. To find publications from the previous version of a product, click Previous versions under the name of the product in the Contents pane.

Shared OMEGAMON XE publications

The publications in this library provide information common to OMEGAMON XE products that are installed on z/OS.

- IBM Tivoli OMEGAMON XE Monitoring Agents on z/OS: Quick Start Guide, GI13-2314
  Summarizes the installation and setup of an OMEGAMON XE monitoring agent on z/OS.

- IBM Tivoli OMEGAMON XE and Tivoli Management Services on z/OS: Common Planning and Configuration Guide, SC22-5484
  Gives instructions for planning and configuration tasks common to the components of Tivoli Management Services on z/OS and the OMEGAMON XE monitoring agents on z/OS.

- IBM Tivoli OMEGAMON XE and Tivoli Management Services on z/OS: Upgrade Guide, SC22-5486
  Gives instructions for complete and staged upgrades to V4.2.0 and V5.1.0 of the OMEGAMON XE products.

- IBM Tivoli OMEGAMON XE and Tivoli Management Services on z/OS: Common Parameter Reference, SC22-5482
  Provides reference information on parameters used for setting up runtime environments and configuring hub and remote Tivoli Enterprise Monitoring Servers on z/OS, including detailed descriptions of the PARMLIB parameters.

- IBM Tivoli OMEGAMON XE and Tivoli Management Services on z/OS: PARMGEN Reference, SC22-5483
  Provides detailed instructions and common configuration scenarios for creating and maintaining runtime environments using the PARMGEN configuration method.

- IBM Tivoli OMEGAMON XE and Tivoli Management Services on z/OS: Tivoli OMEGAMON Enhanced 3270 User Interface Guide, SC22-5487
  Describes the features of the OMEGAMONenhanced 3270 user interface and provides operating instructions and reference material.

- IBM Tivoli OMEGAMON XE and Tivoli Management Services on z/OS: End-to-End Response Time Feature Reference, SC27-2303
  Provides instructions and reference information for the End-to-End Response Time Feature, which supplies response time data to several OMEGAMON XE products.
IBM Tivoli OMEGAMON XE and Tivoli Management Services on z/OS: Reports for Tivoli Common Reporting, SC27-2304

Explains how to use the Tivoli Common Reporting tool to create reports from data displayed in the Tivoli Enterprise Portal and stored in the Tivoli Data Warehouse database.

IBM Tivoli Monitoring library

The publications in this library provide information about the components of Tivoli Management Services (IBM Tivoli Monitoring) that are installed on distributed platforms.

- **Quick Start Guide**
  Introduces the components of IBM Tivoli Monitoring.

- **Installation and Setup Guide, SC22-5445**
  Provides instructions for installing and configuring IBM Tivoli Monitoring components on Windows, Linux, and UNIX systems.

- **High Availability Guide for Distributed Systems, SC22-5455**
  Gives instructions for several methods of ensuring the availability of the IBM Tivoli Monitoring components.

- **Administrator’s Guide, SC22-5446**
  Describes the support tasks and functions required for the Tivoli Enterprise Portal Server and clients, including Tivoli Enterprise Portal user administration.

- **Command Reference, SC22-5448**
  Provides detailed syntax and parameter information, as well as examples, for the commands you can use in IBM Tivoli Monitoring.

- **Messages, SC22-5450**
  Lists and explains messages generated by all IBM Tivoli Monitoring components and by z/OS-based Tivoli Management Services components (such as Tivoli Enterprise Monitoring Server on z/OS, the OMEGAMON enhanced 3270 user interface, and TMS:Engine).

- **Troubleshooting Guide, GC22-5449**
  Provides information to help you troubleshoot problems with the software, including Tivoli Management Services on z/OS components.

- **Tivoli Enterprise Portal online help**
  Provides context-sensitive reference information about all features and customization options of the Tivoli Enterprise Portal. Also gives instructions for using and administering the Tivoli Enterprise Portal.

- **Tivoli Enterprise Portal User’s Guide, SC22-5447**
  Complements the Tivoli Enterprise Portal online help. The guide provides hands-on lessons and detailed instructions for all Tivoli Enterprise Portal features.

- **Agent Builder User’s Guide, SC32-1921**
  Explains how to use the Agent Builder for creating monitoring agents and their installation packages, and for adding functions to existing agents.

- **Tivoli Universal Agent User’s Guide, SC32-9459**
  Introduces you to the IBM Tivoli Universal Agent, an agent of IBM Tivoli Monitoring. The IBM Tivoli Universal Agent enables you to use the monitoring and automation capabilities of IBM Tivoli Monitoring to monitor any type of data you collect.

- **Performance Analyzer User’s Guide, SC27-4004**
  Explains how to use the Performance Analyzer to understand resource consumption trends, identify problems, resolve problems more quickly, and predict and avoid future problems.

- **IBM Tivoli Universal Agent API and Command Programming Reference Guide, SC32-9461**
Explains the procedures for implementing the IBM Tivoli Universal Agent APIs and provides descriptions, syntax, and return status codes for the API calls and command-line interface commands.

Documentation for the base agents
If you purchased IBM Tivoli Monitoring as a product, you received a set of base monitoring agents as part of the product. If you purchased a monitoring agent product (for example, an OMEGAMON XE product) that includes the commonly shared components of Tivoli Management Services, you did not receive the base agents.

The following publications provide information about using the base agents.

- **Operating system agents:**
  - Windows OS Agent User’s Guide, SC22-5451
  - UNIX OS Agent User’s Guide, SC22-5452
  - Linux OS Agent User’s Guide, SC22-5453
  - UNIX Log Agent User’s Guide, SC32-9471
- **Agentless operating system monitors:**
- **Warehouse agents:**
  - Warehouse Summarization and Pruning Agent User’s Guide, SC22-5457
  - Warehouse Proxy Agent User’s Guide, SC22-5456
- **System P agents:**
  - AIX Premium Agent User’s Guide, SA23-2237
  - CEC Base Agent User’s Guide, SC23-5239
  - VIOS Premium Agent User’s Guide, SA23-2238
- **Other base agents:**
  - Tivoli Log File Agent User’s Guide, SC14-7484
  - Systems Director base Agent User’s Guide, SC27-2872
  - Monitoring Agent for IBM Tivoli Monitoring 5.x Endpoint User’s Guide, SC32-9490

Other sources of documentation
You can obtain technical documentation about OMEGAMON XE and related products from a number of additional sources.

- **Service Management Connect (SMC)**
  For introductory information about SMC, see [IBM Service Management Connect](http://www.ibm.com/developerworks/servicemanagement).
  Connect, learn, and share with Service Management professionals. Get access to developers and product support technical experts who provide their perspectives and expertise. Using SMC, you can:
  - Become involved with transparent development, an ongoing, open engagement between external users and developers of Tivoli products where you can access early designs, sprint demos, product roadmaps, and pre-release code.
- Connect one-on-one with the experts to collaborate and network about Tivoli and Integrated Service Management.
- Benefit from the expertise and experience of others using blogs.
- Collaborate with the broader user community using wikis and forums.

The [System z community](https://www.ibm.com/developerworks/servicemanagement/z/index.html) should be of particular interest.

- **IBM Integrated Service Management Library**
  IBM Integrated Service Management Library is an online catalog that contains integration documentation and other downloadable product extensions.

- **Redbooks**
  IBM Redbooks and Redpapers include information about products from platform and solution perspectives.

- **Technotes**
  Technotes provide the latest information about known product limitations and workarounds. You can find Technotes through the IBM Software Support Web site at [http://www.ibm.com/software/support/](http://www.ibm.com/software/support/)
Support information

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

**Online**
Go to the IBM Software Support site at [http://www.ibm.com/support/servicerequest](http://www.ibm.com/support/servicerequest) and follow the instructions.

**IBM Support Assistant**
The IBM Support Assistant (ISA) is a free local software serviceability workbench that helps you resolve questions and problems with IBM software products. The ISA provides quick access to support-related information and serviceability tools for problem determination. To install the ISA software, go to [http://www.ibm.com/software/support/isa](http://www.ibm.com/software/support/isa).

**Troubleshooting Guide**
For more information about resolving problems, see the product's Troubleshooting Guide.
Notices

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 give 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.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, 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 might 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:

IBM Corporation
2Z4A/101
11400 Burnet Road
Austin, TX 78758 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.

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. 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 IBM's application programming interfaces.

If you are viewing this information in softcopy form, the photographs and color illustrations might not be displayed.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol (® or ™), these symbols indicate U.S. registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at www.ibm.com/legal/copytrade.shtml.

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

Other company, product, and service names may be trademarks or service marks of others.
Index

Special characters
$BGL$USR profile 64
%BGL_USER_JCL%(%RTE_NAME%) 64
&bsz variable 251
&expdsn variable 250
&pdsn variable 250
&rhllev variable 250
&sct variable 250
&ssz variable 250
&unit2 variable 250

Numerics
3270 OMEGAMON interface 59

A
Add Runtime Environment panel
base 112
full 112, 115
sharing 112, 115
ADDFILE command 260
address translation
broker partitions 39
enabling 38
external address 39
internal address 39
Alignment field 253
APF-authorization
enable 90
APF-authorizing the load libraries 132
Application field 252
application support 134
applids, VTAM
model 117
prefix 116, 118
wildcard suffixes 117
archived historical data
accessing 248
connecting 249
disconnecting 250
name of persistent data store group 249
audit logging
overview 19
authentication
3270 OMEGAMON interface 59
CUA interface 59
LDAP registry 58
local registry 58
menu interface 59
OMEGAMON II interface 59
passwords 58
Service Console users 60
SOAP server users 60
users 58
authorization
users 58
well-known port 39
autonomous agents 32

B
BACKUP command 258, 259
base libraries 44
base runtime environments 44, 48, 111, 112
batch jobs
JCL 107
log 109
REGION parameter 107
SCAN option 182, 223
SCAND option 182
batch mode
enabling 179
job 180
parameter format 238
parameters 183
scenarios 207, 216
utilities 236
KCICFKEY to manage PF keys 237
KCICPGHP to display batch mode help 238
KCISESET for setting up the environment 236
BATCHLIB library 181, 217
BATCHMEM parameter member 181, 217
best practices 26, 31
broker partitions 39
building runtime libraries 121

C
CANCTDCS logmode 118
CBPDO 21
CISUPG2 member 109
CICATB job
creating 180, 217
data dictionary 182, 188
location 217
reports 183
running to validate changes 223
SCAN option 182, 223
SCAND option 182
SUBMIT options 181, 217
CLIST, executing in TKANCUS library 232
cloning override files 80
cloning runtime environments 179
CMS_NODEID environment variable 29
codes, product 265
Col Row Len field 252
Column Count field 252
column description record 253
Column Length field 253
Column Map field 255
Column Name field 253
commands
ADDFILE 260
BACKUP 259
COMMIT 262
DELFILE 260
EXTRACT 258, 260
INITIDS 261
ISPF RECOVERY OFF 231
persistent data store 258
commands (continued)
  QUERY CONNECT 262
  QUERY DATASTORE 262
  RECOVER 261
  RECOVERY OFF 231
  RESUME 261
  SWITCH 258
  COMMIT command 262
  Common Event Console 7
communication protocols
decisions 36
limitations 36
piped 36
port numbers 40
SNA 36, 117
SOAP server 31
TCP/IP 36, 37, 40, 118
communications, securing 57
completing the configuration 127, 129
components
codes 265
overview 3
configuration
completing 127, 129
Configuration Tool 103
copying values 115
decisions 25
planning 25
runtime environments 46
security, runtime environment 117
setting up Configuration Tool 103
verifying 134
configuration methods
Configuration Tool (ICAT) 34
PARMGEN Workflow user interface 34
configuration profile (PARMGEN) 82
configuration profile parameters
validate using PARMGEN 96
configuration profiles 64
Configuration Tool 3270 Model 2 display requirements 231
batch mode 179
batch utilities 236
commands
PFSHOW 231
Configuration Tool
display requirements 231
DEBUG options 232
display requirements 104, 231
displaying an ISPF table 232
dynamically generated content 105
edit recovery 105
environment 103, 104, 106
executing a CLIST in TKANCUS library 232
function keys 231
help 105
ISPF
display requirements 231
ISPF features 105
keys, PF 231
online help 105
panel IDs 105
PF keys 104, 231
PFSHOW command 231
prerequisites
  Configuration Tool 231
display 231
Configuration Tool (continued)
  recovery 105
  reference information 231
  requirements
  Configuration Tool 231
display 231
  restrictions 231
  rules 231
  Runtime Environments (RTE) panel 109, 239
  services 231
  setup 103
  split screen 105
  starting 104
  system variable support
  creating the VTAM major node rename job 204
  enabling 202
  parameter member 203
  sample usage 202
  unlocking high-level qualifiers 231
  usage notes 104
  utilities 219, 231
  work environment 106
  work libraries 108
Configuration Tool (ICAT) configuration method 34
Configuration Tool environment
  cloned 103
  new 104
Configuration Tool panels
Add Runtime Environment 112, 115, 119
Configure Products 110, 202, 204, 205
Create/Clone Batch Mode Parameters 184, 220
Create/Clone Parameter Deck Menu 219
KCIMBATO 219
KCIMRTU 219
KCIPBATO 220
KCIPIN1 106
KCIPIN2 106
KCIPPLST 110, 202, 204, 205
KCIPPR4 108
KCIPRCM 110, 202, 204, 205
KCIPRDME 240
KCIPRIM 104
KCIPRTA1 115
KCIPRTA2 119
KCIPRTAB 112
KCIPRTE 110
KCIPRTEA 112
Main Menu 104
Product Selection Menu 110, 202, 204, 205
README Table of Contents Display 240
RTE Utility Menu 219
Runtime Environments (RTEs) 110
Set up Work Environment 106
Specify JCL Options 106
Specify Nonstandard Parameters panel 125
configuration verification report 122
Configure Products panel 110, 202, 204, 205
console views 7
COUNT option 41, 42
Create/Clone Batch Mode Parameters panel 184, 220
Create/Clone Parameter Deck Menu panel 219
CSI
  cloned 103
  decisions 25
  existing 103
  new 104
CSI (continued)
  shared 25
CTDDSN node ID 131
CUA interface 59
CUASITE profile 139
Customize PARMGEN configuration profile members panel 86

D
Dashboard Application Services 8
dashboard applications 8
data dictionary, CICATB 182, 188
Data Length field 255
Data Offset field 255
data record format, exported historical data 251
data sets
  INSTDATA 108
  INSTDATW 108
  INSTJOBS 109
  INSTLIB 109
  INSTLIBW 109
  INSTQCK 109
  INSTSTAT 109
  naming convention 45
  SYSTCPD 129
  VTAMLST 131
Data Store Path field 254
Data Store Ver field 254
debugging 232
decisions
  communication protocols 36
  configuration methods 34
  CSI 25
  high-availability hub 27
  historical data collection 53
  monitoring agents 31
  placement of monitoring servers 26
  platform for hub monitoring server 26
  runtime environments 44
  security 56
  user interface 32
default certificate 57
DELFILE command 260
DELFILE statement 250
demo mode 145
deployment
decisions 25
planning 25
developerWorks 273
diagnostic SYSOUT class 115
dictionary header record 252
Dictionary Len field 252
dictionary mappings, historical data 251
Display PARMGEN environment analysis panel 77
DVIPA 27, 36
Dynamic XE to 3270 (Classic) linking feature 119

E
EBDInC files, exporting to 255
edit macros
  IBM-provided 243
  edit recovery 105, 231
  EGG1 encryption scheme 57
End-to-End Response Time collector 8
enhanced 3270 user interface
demo mode 145
enhancements 18
environment variables
  CMS_NODEID 29
  KDE_TRANSPORT 41, 42
  KDE_INTERFACELIST 42
ephemeral pipe support (EPS)
defined 38
  disabling 38
  forcing ephemeral connections 38
errors, parameter 224
event synchronization 7
Expansion Area field 254
Expansion field 253
Export Path field 254
Export Version field 254
exporting historical data 250, 251
EXTRACT command 258, 260
extracting historical data to flat files 255

F
features, new 18
firewalls
  address translation 37
  basic implementation 37
  broker partitions 39
  ephemeral pipe support (EPS) 37
  gateway support 39
  piped protocols 36
  with IPPipe protocol 37
  with SOAP 29
Flag field 253
flat file, extracting historical data to 255
FMIDs 22
folding passwords to uppercase 117
format, product tapes 21
fragmentation, VSAM control area 26
full runtime environments 44, 47, 111, 112, 115

G
gateway, firewall 39
GBL_DSN_TCP_SYSTCPD_TCPDATA 89
global profile 64
global SAF class name 117
Global Security Toolkit (GSKit) 57
global table, SOAP server 30
granular control 17
Group field 254
Group parameter 250

H
Header Len field 252
header record 256
help, Configuration Tool 105
HFS/zFS USS directory
  update 91
hidden parameters 125
Hierarchical File System/zSeries File System (HFS/zFS) 120
high-availability hub 19, 27
high-level qualifiers
  length 45
  non-VSAM libraries 113

Index 281
high-level qualifiers (continued)
  shared CSI 25
  unlocking 231
  VSAM libraries 113

historical data
  accessing 248
  exporting 250
  restoring 248, 251

historical data collection
  allocating space 55
  decisions 53
  defined 53
  enable 146
  interval 54
  limiting amount 54
  location 54
  managing 56
  planning 53

host name
  runtime environment 118
  Telnet listener 119

Hostname field 118, 119

HTTPS protocol 43, 57

hub monitoring server
  decisions 26
  defined 5
  high-availability 27
  location 26
  platform 26

SOAP server list 30

IBM Support Assistant 275
IBM Tivoli OMEGAMON enhanced 3270 user interface 18,
32, 57
Identifier Num field 252
IEBGENER utility 249
iKeyMan 57
IKJTSOnm member 145
imbed files 65
INITIDS command 258, 261
installation
  checking for fix packs 21
  media 21
INSTDATA data set 104, 108
INSTDATW data set 104, 108
INSTJOBS data set 104, 109
INSTJOBS library 217
INSTLIB data set 104, 109
INSTLIBW data set 104, 109
INSTLOG data set 104, 109
INSTLOGW data set 104, 109
INSTLQCK data set 104, 109
INSTSTAT data set 104, 109
Integrated Cryptographic Service Facility (ICSF) 57
Integrated Service Management Library 273
  integration with other products 9
  interfaces
    OMEGAMON (3270)
    OMEGAMON II (CUA) 33
  interfaces, network 42
interim libraries (PARMGEN) 82
Internet Inter-ORB Protocol (IIOP) 57
interoperability 9
Interoperable Object Reference (IOR) protocol 57
IP address 118
IP.PIPE protocol
  address translation 37
  ephemeral pipe support (EPS) 37
  with firewalls 37
ISA 275
ISPF
  displaying table 232
  edit recovery 105
  PANELID command 105
  RECOVERY command 105
  RECOVERY OFF command 231
  split screen 105

J
  Jazz for Service Management integration service 7
JCL
    options 106
    REGION 107
    stored 109
    suffix 46, 114
  job index (PARMGEN) 79
  jobs, naming convention 46

K
  KAG_X_STC_SYSTCPD_INCLUDE_FLAG 89
  KCIĆFKKEY utility 237
  KCIĆPGHP utility 238
  KCIJCALO job 99
  KCIJCOPY job 100
  KCIJCJIP job 100
  KCIJCLNK job 100
  KCIJLCMD job 99
  KCIJSEC job 99
  KCIJLSYS job 100
  KCIJLUPV job 100
  KCIJUSP job 100
  KCIJUSN job 100
  KCIJW2R job 100
  KCIMBATO panel 219
  KCIMRTU panel 219
  KCIP@PG0 panel 69
  KCIP@PG2 panel 72
  KCIP@PG3 panel 73
  KCIP@PG4 panel 77
  KCIP@PG6 panel 220
    customize 86
  KCIPBATO panel 220
  KCIPIN1 panel 106
  KCIPIN2 panel 106
  KCIPPST panel 110, 202, 204, 205
  KCIPPR4 panel 108
  KCIPRCM panel 110, 202, 204, 205
  KCIPRDME panel 240
  KCIPRIM panel 104
  KCIPRTA1 panel 115
  KCIPRTA2 panel 119
  KCIPRTAB panel 112
  KCIPRTAE panel 110, 239
  KCIPRTAE panel 112
  KCISYUP utility 236
  KCISYUPJ job 131
  KCISYPJB job 130
  KCITPIG1 table 217
KDE_TRANSPORT environment variable
  COUNT option 41, 42
  POOL option 41, 42
  SKIP option 41, 42
KDSB_INTERFACELIST environment variable 42
KDS_X_STC_SYSTCPD_INCLUDE_FLAG 89
KDSMTAB1 logmode table 118
KDSTHUBS table 30
KOBCUA profile 139
KPDCARCH program 257
KPDCOMMJ job 249
KPDDISCO 145
KPDDISCO program 257
KPDMANE program 257
KPDPROC* procedures 130
KPDPROCI procedure 245, 246
KPDPROCC 145
KPDPROCC CLIST 245, 246
KPDEST program 257
KPDUUTIL program 257
KPDXTRA program 255, 257
KSHXHUBS member 30

L
languages
  packs 22
  support 22, 134
LDAP registry 58
Length field 254
libraries
  base 44
  Configuration Tool 108
  INSTJOBS 217
  LPAR-specific 44
  OMEGAMON XE shared publications 271
runtime
  building 120
  defined 44
  loading 127
  SMP/E target 44
  target 44
  TKANCUS 232
  types 44
  work 108
library, UKANDATV 13
lists
  hub monitoring servers 30
  user IDs 30
load libraries
  APF-authorizing 132
  RKANMOD 132
  RKANMODL 132
  RKANMODP 132
  RKANMODU 132
  TKANCUS 132
  TKANMODL 132
  TKANMOPD 132
load optimization 114
log, configuration batch jobs 109
logmode table 118
logon profile 139
logon profiles 142
low-level qualifiers 45
LPAR configuration profile 64
LPAR-specific libraries 44
LU6.2 logmodes 118
LUGROUP field 120

M
Main Menu panel 104
maintenance scenarios
  transport scenarios supported 149
Map Length field 252
maps, parameter 122
member, IKJTSOmm 145
members, user-modified 115
menu interface 59
menu, main 104
Mgmtclas field 107, 113
mid-level qualifiers 45, 114
mixed-case passwords 117
model applids, VTAM 117
monitoring agents
  address space 31
  address spaces 31
  autonomous 32
  base 8
  completing the configuration 129
  configure using PARMGEN 93
  defined 8
  OMEGAMON 8
  overview 6
  packaging 22
  stand-alone 31

N
naming
  backup historical data sets 248
  data sets, runtime environment 45
  jobs, runtime environment 46
  members, JCL 46
  TEMS name 29
  Netcool/OMNIbus ObjectServer 7
  NETID parameter 118
NetView
  Take Action command authorization 20
  network ID 118
  network interface list 42
  new configuration features 10
  new features 18
  new in this release 10
  Newest Row field 253
  NOFF option 255
  nonstandard parameters 125

O
Oldest Row field 253
OMEGAMON (3270) interface 33
OMEGAMON enhanced 3270 user interface
  configure using PARMGEN 92
  considerations 57
  enable security 140
  features 5, 18
  requirements 32
OMEGAMON II (CUA) interface 33
OMEGAMON II interface 59
OMEGAMON subsystem 8
OMEGAMON XE
  components 3
OMEGAMON XE (continued)

overview 3
OMEGAMON XE monitoring agents 22
OMNIMON base 8
OMVS segment, default 37
online help, Configuration Tool 105
opaque ports 42
Open Services Lifecycle Collaboration Performance Monitoring (OSLC-PM) service provider 7
optimization, loading runtime libraries 114
Overlay Col ID field 253
Overlay Col Off field 253
override files 80
override values 65

parameters (continued)
RTE_NAME_NODEID 75
RTE_TYPE 75
RTE_VSAM_HILEV 74, 75, 200
RTE_VTAM_APPLID_PREFIX 75
RTE_X_HILEV_SHARING 75
RTE_X_SECURITY_EXIT_LIB 76
setting 122, 124
system variable parameter member 203

PARMGEN
complete set up 98
configuration profile 82
configuring a Tivoli Enterprise Monitoring Server 91
configuring an OMEGAMON enhanced 3270 user interface 92
configuring OMEGAMON monitoring agents 93
conversion job 83
create runtime environment members and jobs 97
customize configuration profiles 84
enable APF-authorization statements (Optional) 90
able enable self-describing agent feature 91
interim libraries 82
job index review 79
maintenance scenarios 149
prerequisites 66
scenarios
PGN06 164
starting the PARMGEN Workflow user interface 67
submit batch jobs 98
SYSTCPD DD statement 89
update global parameters 93
update global VTAM major node 90
update TCP/IP port values 89
update the HFS/zFS USS directory 91
update user prolog (optional) 87
validate profile parameter values 96
verify products 88

PARMGEN configuration method overview 18
PARMGEN configuration profiles
customize 84
PARMGEN Workflow user interface set up system variables 199
starting 67
PARMGEN Workflow user interface configuration method 34
partitions 39
passwords
authentication, configuring 58
mixed-case 117
uppercase 117
PDPROC prefix 245
PDSE field 107, 113
PDSLOG SYSCSOUT 250
performance, effect of fragmentation 26
persistent data store
archived data group name 249
archived data, accessing 248
archived historical data
restoring 248
automatic maintenance 246
backup data sets, naming 248
command interface 258
commands 258
components
KPDARCH 257
KDASICO 257
KPDMAINE 257
persistent data store (continued)
components (continued)
  KPDREST 257
  KPDUUTIL 257
  KPDXTRA 257
connecting archived data sets 249
controlling maintenance 246
data record format 251
data set backup 247
disconnecting archived data sets 250
export data sets, naming 247
exporting data 250
extracting data to EBCDIC files 256
extracting data to flat files 255
maintaining 56
operation 258
overview 245
restoring data 251
user ID for running procedures 246
piped protocols
  firewall support 37
  limitations 36
platforms 26
POOL option 41, 42
Port number field 119, 120
port numbers
  algorithm 41
  allocating 41
  assigning 40, 41
  changing 41, 42
  COUNT option 41
  default 40
  POOL option 41
  range 42
  reserving 40, 42
  SKIP option 41
  Telnet listener 120
ports
  opaque 42
  well-known 39, 40, 41, 119
prefix
  started task procedures 114
  VTAM applid 116, 118
prerequisites
  TCP/IP protocols 37
  prerequisites (PARMGEN) 66
  probes, shared 8
  product codes 265
  product integration 9
  product registration
    automatic 19
Product Selection Menu panel 110, 202, 204, 205
publications
  hardcopy 21
  OMEGAMON XE 271
  shared 271
  Tivoli Management Services on z/OS 271
Q
QUERY CONNECT command 262
QUERY DATASTORE command 262
R
RACF
  MIXEDCASE option 117
  passwords 117
README Table of Contents Display panel 240
RecID field 252, 253, 254
RECOVER command 261
RECOVER logic 258
RECOVERY command 105
RECOVERY OFF command 231
RedBooks 273
REGION parameter 107
Registry Services 7
remote
  defined 4
  remote monitoring server
    configuring in batch mode 216
    decisions 26, 28
    defined 4
    location 26, 28
    reporting to hub on z/OS 216
Remote RTE for Transport field 114
replicating runtime environments 149, 179
reports
  CICATB job 183
  configuration verification 122
  parameter maps 122
requirements
  communication protocols 31, 36
  DVIPA 36
  firewall support 37
  security 56
  SOAP server 29
  TEMS name 29
  Tivoli Management Services 21
  VIPA 36
RERUN option 181, 217
RESUME command 261
RESUME logic 258
RKANSAMU data set 130
RKDSKAS library 13
Row Count field 253
Row Length field 255
RTE directory field 120
RTE Utility Menu panel 219
runtime environment members and jobs
  create using PARMGEN 97
runtime environments
  actions required 240
  Add option 239
  base 44, 48, 111, 112
  Build option 239
  building libraries 121
  cloning 179
  completing the configuration 129
  Configuration Tool 109, 239
  Configuration Tool options 239
  Configure option 239
  configuring 46
  copying configuration values 115
  data sets 45
  decisions 44
  defined 44
  defining using shared DASD 191
  deleting 241
  full 44, 47, 111, 112, 115
  JCL 46

Index 285
runtime environments (continued)
jobs 46
libraries 44
load optimization 114
Load option 239
name, system variable 116
naming convention
data sets 45
jobs 46
optimization 114
options 239
overview 44
PLBSP14 164
replicating 149, 179, 207
security 117
self-contained 44, 47
shared DASD 191
sharing 111, 112, 115
sharing with base 44, 49, 116
sharing with full 45, 51
sharing with SMP/E 45, 52
transporting 114, 190, 193
batch mode parameters 195
from one image to another 192
types 46
Update option 239
update TCP/IP port values 89
Utilities option 239
verify products 88
View option 239
VTAM major node 204
Runtime Environments (RTEs) panel 110, 239
runtime environments, replicating 149
runtime environments, verifying configuration of 126
runtime libraries
building 121
defined 44
loading 127
runtime members, analyzing 115

S
sample procedures 130
SCAN option 181, 182, 217, 223
SCAND option 181, 182, 188, 217
scenarios
batch mode 207, 216
PGN06 164
PLBSP14 164
remote monitoring server with z/OS hub 216
replication of configured environment 207
secure protocols
HTTPS protocol 43, 57
Internet Inter-ORB Protocol (IIOP) 57
Interoperable Object Reference (IOR) protocol 57
Secure Socket Layers (SSL) 43, 57
security 145
3270 OMEGAMON interface 59
command-level 59
communications 57
CUA interface 59
enable 137
firewall implementation 37
menu interface 59
OMEGAMON enhanced 3270 user interface 140
OMEGAMON II interface 59
product-level 59
security (continued)
requirements 56
runtime environment 117
Service Console 60
SOAP server 30, 60
UNIX System Services (USS) 120
security tokens 17
Security: Validate User option 58
securityenhanced 3270 user interface 142
self-describing agent
enable 91
feature overview 19, 28, 31
self-describing agent feature 17
self-describing agents 91
ServerPac 21
Service Console user authentication 60
Service Management Connect 273
Set up Configuration Environment panel 108
Set up PARMGEN work environment for a runtime
environment panel 69, 72
SET UP PARMGEN WORK ENVIRONMENT FOR AN RTE
panel 73
Set up Work Environment panel 106
shared CSI
advantages 25
guidelines 25
shared probes 8
shared publications 271
sharing runtime environments
system variables 116
with base 44, 49
with full 45, 51
with SMP/E 45, 52
Simple Object Access Protocol (SOAP) 29
single sign-on 58
Situation Event Console 7
SKIP option 41, 42
Slot Size field 254
SMC 273
SMP/E
sharing target libraries 45, 52
target libraries 44
SMS considerations 113
SNA protocol
requirements 36
VTAM communication values 117
SOAP server
aliasing 30
communication protocols 31
configuration 30
defined 29
global table 30
hub monitoring server list 30
in firewall environments 29
requirements 29
security 30, 60
terminology 30
user access 30
user access list 30
web services 20
Software Support 275
space requirements, historical data collection 55
Spare 1 field 253
Spare field 253
Specify JCL Options panel 106
Specify Nonstandard Parameters panel 125

286 IBM Tivoli OMEGAMON XE and Tivoli Management Services on z/OS: Common Planning and Configuration Guide
specifying
  global SAF class name 117
SPIPE protocol 57
split screen 105
SQL Type field 253
stack name, TCP/IP 37
stand-alone
  monitoring agents 31
  monitoring servers 31
started tasks
  managing 267
  naming conventions 267
  prefix 114
  sample procedures 130
TCP/IP server 118
  Workload Manager (WLM) settings 269
stash file 57
STC prefix 114
storage
  allocation for work libraries 109
  requirements 25
  specification 108
Storclas field 107, 113
SUBMIT options 181, 217
suffix
  JCL 46, 114
  wildcard 117
Summarization and Pruning agent 6
support assistant 275
SVC 34 258
SWITCH command 258
SWITCH logic 258
sysadmin user account 58
sysadmin user ID 56
SYSCLONE, system variable 116
SYSNAME system variable 116
SYSOUT class
  diagnostic 115
  non-diagnostic 114
SYSTCPD data set 129
system variables 199
  base runtime environment name 116
  changing status 203
  enable 200
  enabling 116, 202
  parameter member 203
  runtime environment name 116
  sample usage 202
  status 203
SYSCLONE 116
SYSNAME 116
  using the PARMGEN Workflow user interface 199
VTAM applid prefix 116
VTAM major node rename job 204
SystemPac 21
tapes, product 21
target libraries, SMP/E 44, 45, 52
TCP/IP
  communication values 118
  host name 118
  interface list 42
  IP address 118
  network interfaces 42
  OMVS segment, default 37
  port number changes 41
  port numbers 40
  requirements 36, 37, 40
  server started task 118
TCP/IP stack name 37
TCPIP.DATA data set 89
Technotes 273
Telnet sessions
  host name override 119
  LU group 120
  overrides 119
  port number override 120
TEMS name
  guidelines 29
  specifying 115
TEPS/e 7
Timestamp field 254
Tivoli Authorization Policy Server 8
Tivoli Common Reporting 8
Tivoli Data Warehouse 6
Tivoli Enterprise Console
  event server 7
  event synchronization 7
  event view 7
  integration 7
Tivoli Enterprise Monitoring Automation Server 7
Tivoli Enterprise Monitoring Server 4
  completing the configuration 129
  configure using PARMGEN 91
  configuring 121
  defined 4
  high-availability hub 27
  hub
    defined 5
    high-availability 27
    location 26
    name 29
    placement of hub 26
    placement of remote 26, 28
    platform decision 26
    remote
      configuring in batch mode 216
      location 26, 28
      reporting to hub on z/OS 216
    TEMS name 29
    verifying the configuration 134
    VIPA name 36
Tivoli Enterprise Portal
  authorizing users 58
  client, defined 5
  enable historical data collection 146
Tivoli Enterprise Portal Server 5
Tivoli Enterprise Portal Server extended services (TEPS/e) 7
Tivoli Management Services
  components 3
  distributed components 3
  overview 3
  packaging 21
Tivoli Management Services (continued)
required software 21
TKANCUS library 232
TMS:Engine, defined 6
TN3270 Telnet sessions 119
Total Slots field 254
transporting runtime environments
  batch jobs 193
  batch mode parameters 195
  between z/OS images 192
  methods 190

U
UKANDATV library 13
Unit field 107, 113
UNIX System Services (USS) 119
unlocking high-level qualifiers 231
uppercase passwords 117
Used Slots field 254
user access list 30
user IDs 58
  persistent data store procedures 246
  SOAP server access 30
  sysadmin 56
  validation 30
user interfaces
  3270 “green screen” 32
  Tivoli Enterprise Portal interface
    OMEGAMON enhanced 3270 user interface 32
  user-modified members 115
users
  authenticating, 58, 59
  authorizing 58
utilities, Configuration Tool 219

V
Variable Cols field 252
Verify configuration option 126
verifying configuration of runtime environments 126
verifying the configuration 134
VIPA
  dynamic 36
  monitoring server name 36
  requirements 36
Volser field 107, 113
VSAM files
  control area splits 26
  fragmentation 26
  managing 26
  performance considerations 26
VSAM libraries
  high-level qualifiers 113
VTAM
  applids
    model 117
    prefix 116, 118
    wildcard suffixes 117
  communication values 117
  definitions 131
  logmode table 118
  major node
    rename job 204
    runtime environment 204
    varying active 132

VTAM (continued)
  network ID 118
  VTAMLST data set 131
  VTAMLST startup member 118

W
Warehouse Proxy agent 6
WCONFIG(%RTE_NAME%) profile 64
WCONFIG(KCIIJUP1) job 83
well-known port
  authorization 39
  changing 41
  default number 40, 119
wildcard suffixes, VTAM applids 117
work environment 106
work libraries 108
work libraries (PARMLIB) 69
Workload Manager (WLM)
  started task settings 269

X
XF macro 243