IMS Enterprise Suite
Version 3 Release 1

SOAP Gateway Administrator's Guide and Reference

SC19-4112-05

IMS Enterprise Suite
Version 3 Release 1

SOAP Gateway Administrator's Guide and Reference

SC19-4112-05

Note

Before using this information and the product that it supports, be sure to read the general information under "Notices" on page 467.

Contents

About this information

These topics provide conceptual, guidance, and reference information for installing, managing, and troubleshooting the IMS™ Enterprise Suite SOAP Gateway server. The topics also describe how to generate the required artifacts and how to deploy web services for the supported usage scenarios: IMS applications as web services, IMS applications calling out to external web services, and IMS application emitting business event data to external event processing servers.

This information is available in IBM® Knowledge Center at www.ibm.com/support/knowledgecenter.

How to read syntax diagrams

The following rules apply to the syntax diagrams that are used in this information:

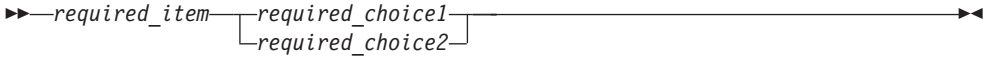
- Read the syntax diagrams from left to right, from top to bottom, following the path of the line. The following conventions are used:
  - The >>--- symbol indicates the beginning of a syntax diagram.
  - The ---> symbol indicates that the syntax diagram is continued on the next line.
  - The >--- symbol indicates that a syntax diagram is continued from the previous line.
  - The --->< symbol indicates the end of a syntax diagram.
- Required items appear on the horizontal line (the main path).

- Optional items appear below the main path.

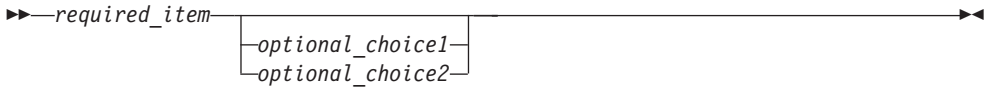
  If an optional item appears above the main path, that item has no effect on the execution of the syntax element and is used only for readability.

- If you can choose from two or more items, they appear vertically, in a stack.

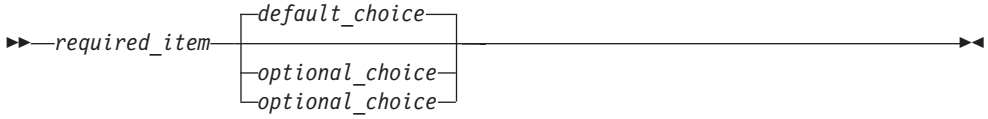
  If you *must* choose one of the items, one item of the stack appears on the main path.



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

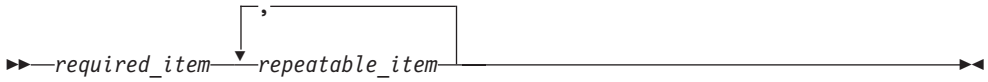


If one of the items is the default, it appears above the main path, and the remaining choices are shown below.



- An arrow returning to the left, above the main line, indicates an item that can be repeated.

If the repeat arrow contains a comma, you must separate repeated items with a comma.



A repeat arrow above a stack indicates that you can repeat the items in the stack.

- Sometimes a diagram must be split into fragments. The syntax fragment is shown separately from the main syntax diagram, but the contents of the fragment should be read as if they are on the main path of the diagram.

fragment-name:

- In IMS, a b symbol indicates one blank position.
- Keywords, and their minimum abbreviations if applicable, appear in uppercase. They must be spelled exactly as shown. Variables appear in all lowercase italic letters (for example, *column-name*). They represent user-supplied names or values.
- Separate keywords and parameters by at least one space if no intervening punctuation is shown in the diagram.
- Enter punctuation marks, parentheses, arithmetic operators, and other symbols, exactly as shown in the diagram.
- Footnotes are shown by a number in parentheses, for example (1).

How to send your comments

Your feedback is important in helping us provide the most accurate and highest quality information. If you have any comments about this or any other IMS information, you can take one of the following actions:

- Click the Feedback link at the bottom of any IBM Knowledge Center topic.
- Send an email to imspubs@us.ibm.com. Be sure to include the book title and the publication number.

Chapter 1. Release overview

An overview of the new features and new and changed properties, commands, messages, and behaviors is provided to help you plan for installation and upgrades.

Use the information in the "Information impact by feature" section to learn more about each new feature and general release changes.

Release notes for IMS Enterprise Suite V3.1 SOAP Gateway

Read this document to find important installation information. You can also learn about product updates, compatibility issues, limitations, and known problems.

Contents

Description

For new features in this release, see "New features in IMS Enterprise Suite Version 3.1 SOAP Gateway" on page 6.

System requirements

For information about hardware and software compatibility, see "System requirements" on page 41 and "Software requirements" on page 43.

Installing SOAP Gateway

For installation instructions, see Chapter 3, "Installing and configuring SOAP Gateway," on page 41.

For information about how to apply maintenance services, see:

- "Applying maintenance services on z/OS" on page 115
- "Applying maintenance services on distributed platforms" on page 116

Updates and fixes

Updates to SOAP Gateway are available via APAR PTFs for the z/OS® platform and via the IMS Enterprise Suite download website for other platforms. For z/OS, see the APAR ++HOLD card for instructions on how to apply the service. For other platforms, the instructions are provided on the download website.

Java™ prerequisites

Before installing or updating SOAP Gateway, download the latest version of Java that is supported by IMS Enterprise Suite.

- For z/OS environments, see the PSP bucket for the latest supported Java version and download instructions. The instructions also specify how to order Java for z/OS through Shopz at no charge.
- For Windows environments, visit the IMS Enterprise Suite downloads site and navigate to the download page for SOAP Gateway. Follow the instructions on the page to download the latest supported Java version.

The following table lists the current service levels and an overview of the fixes and enhancements for each level.

Table 1. Updates and fixes

Service level	APAR (z/OS platform)	Updates
3.1.0.4	PI41092	This service updates the underlying application server. All three components, IMSSERVER, IMSBASE, and IMSSOAP are updated.
3.1.0.3	PI23543 and PI24654 The following APARs are required: • PI23939 • PI23955, PI23956, and PI23958 for 64-bit Java. • PI23952, PI23953, and PI23954 for 31-bit Java.	This service adds the support for callout transaction logging, and updates the underlying application server and IBM Java SDK. This service also addresses the following known issue: • In some resource-contention situations, SOAP Gateway might send the acknowledgement to IMS about reception of the callout request after the callout response is already sent back to IMS. Although the request was processed successfully, an OTMA protocol violation error could surface in the z/OS System Display and Search Facility (SDSF) log: HWSP1495E OTMA PROTOCOL VIOLATION; R=20, C=IOGXTETR, DS=DJR1 , M=SDRC This issue is addressed. For new installations, you can install 3.1.0.3 directly without having to install the base code because all three SOAP Gateway components are updated in this service. This service includes all fixes that were previously addressed.
3.1.0.2	PI21042 The following APARs are required: • PI18382, PI18384, PI18385, and PI18386 for 64-bit Java. • PI18472, PI18473, and PI18474 for 31-bit Java.	This service updates the underlying application server. There are no other changes. For new installations, you can install 3.1.0.2 directly without having to install the base code because all three SOAP Gateway components are updated in this service. This service includes all fixes that were previously addressed.

Table 1. Updates and fixes (continued)

Service level	APAR (z/OS platform)	Updates
3.1.0.1	PI10746 The following APARs are prerequisites for an updated Java: PI11146, PI11147, PI11337, PI11148, PI11150, PI11336, and PI11233.	This service addresses the following issues: • For the send-only with acknowledgement feature for synchronous callout, when the response is successfully delivered to the initiating ICAL call, OTMA sends the ACK back to SOAP Gateway with the response message included. This behavior might be a potential performance issue when the response data is large. This issue is addressed. The response data is no longer sent with the ACK. IMS V13 APARs PM90943 and PI10653 are required. • If the SOAPAction attribute in a WSDL file was set to a URL, the correlator file could not be created, because the SOAPAction value was used as the correlator file name. This issue is now fixed, and only the last string in the URL is used as the file name. • For installation on the z/OS platform, the AEWPOSIN job converts some files to a specific code page that might not work on all systems. This issue is addressed. The AEWPOSIN job now requires one extra argument to specify the code page.
3.1.0.0		The following previous known issues are addressed in V3.1: • The migration utility does not prevent users from setting a shutdown port to port 0. Port 0 is reserved and should not be allowed. Valid port values should be 1 - 65535. This issue is fixed. The SOAP Gateway management utility now does not allow users to set the shutdown port to 0. • If you undeploy a service while the server is down, you get an incorrect, partial error message: `Unable to connect to the runtime server (EDC8128I Connection refused.` The correct message should be the IOGD0751W message: `The undeploy command successfully removed the service associated with` *`correlator_file_name`* `from the SOAP Gateway master configuration. However, it encountered the following error(s): Unable to connect to the runtime server EDC8128I. Connection refused.` This issue is fixed, and the correct error message is provided. • When a WS-Security-enabled callout service is deployed while the server is up, the iogmgmt -view -cf <correlator_file_name> command does not reflect the callout WS-Security type. The output from the command shows the value for the callout WS-Security type as blank. This issue is fixed.

Known issues and workarounds

At the time of publication, the following issues were known.

Table 2. Known issues and workarounds

Issue description	Workaround
When transaction logging is enabled, but the transaction log file is deleted, no transaction information is logged and no error is reported.	To resume the logging, turn off the transaction logger (iogmgmt -tranLog -off) and turn it on again (iogmgmt -tranLog -on) to recreate the log file and start the logging.
When a value other than `-true` or `-false` is specified with the send-only-with-ack flag (the `-k` option) with the iogmgmt -corr command, SOAP Gateway management utility does not indicate that the value is not valid. The Send Only with Ack property remains unchanged (the default is `false`).	Use the iogmgmt -view -cf *correlator_file* command to view the Send Only with Ack property value. Reissue the iogmgmt -corr command to set the `-k` flag to either `true` or `false`.
If you are on V2.1 SOAP Gateway, after installing V2.2 and running the SOAP Gateway migration utility to migrate server properties and services, a warning might occur when you start up the server: `WARNING: [SetAllPropertiesRule] {Server/Service/Connector} Setting property 'maxSpareThreads' to '75' did not find a matching property.`	This property is no longer supported in the new version of the underlying Apache Tomcat server in V2.2 SOAP Gateway. If you encounter this warning, remove the maxSpareThreads property from the `server.xml` file and restart the server.
The JVM security option in the AEWIOGCF configuration member for TLS v1.2 support for the callout scenario that is required by NIST SP800-131A is missing a closing quotation mark at the end: `# IJO="$IJO -Dcom.ibm.ims.soap.httpsProtocolType= TLSv1.2`	To enable the support for NIST SP800-131a for the callout scenario, after uncomment out the line, add a closing quotation at the end before you start the SOAP Gateway server: `IJO="$IJO -Dcom.ibm.ims.soap.httpsProtocolType= TLSv1.2"`

For installation on the z/OS platform, set the code page is CP1047. For systems with other code pages, some characters in the installation sample jobs, such as \ ^ ~ ! [ ] { } # | ` $ and @ in AEWIOGBP might be interpreted differently.	Change the following statements in AEWIOGBP into three single calls:
	`CMD[0]= "iogmgmt -prop -u -java -h $IOGJH";+ CMD[1]="iogmgmt -view -java -h";+ CMD[2]="iogmgmt -view -sgp";+ for _CMD in "${CMD[@]}"; + do $MGM/$_CMD >>$OUT 2>>ERR;+ done`
	
	For example:
	`$MGM/iogmgmt -prop -u -java -h $IOGJH $MGM/iogmgmt -view -java -h $MGM/iogmgmt -view -sgp`

Table 2. Known issues and workarounds (continued)

Issue description	Workaround
On Windows, after the SOAP Gateway code is successfully installed, further installation as a Windows service or starting the server as a Windows service results in an IOG000024E (the service failed to install) or IOG00026E (Windows service failed to start) error.	The user installing or starting the SOAP Gateway server as a Windows service is not an administrator. This issue is most likely to occur on the Windows 7 platform because of its enhanced security to keep non-system related files running in a restricted mode. Add the user responsible for installing and starting the SOAP Gateway server as a Windows service to the administrators' group. Alternatively, a non-administrator user can start the server as an administrator by selecting **Start** > **All Programs** > **IBM IMS Enterprise Suite V3.1** > **SOAP Gateway**, right-clicking **Management Utility**, and selecting **Run as administrator**. In the Management Utility command prompt window, issue the following command: `iogmgmt -service -start`
When the Application Transparent Transport Layer Security (AT-TLS) feature in IBM z/OS Communications Server is used to manage secured connections to the SOAP Gateway server, you receive "Connection Interrupted" instead of a list of web services when you view the deployed web services in the SOAP Gateway administrative console.	Modify the `httpbase.jsp` file in `<soap_install_dir>/imsserver/server/webapps/imssoap/axis2-web/include/` to explicitly set the scheme to HTTPS. See the technote at http://www.ibm.com/support/docview.wss?uid=swg27024607 for details.
The SOAP Gateway management utility iogmgmt -batch command does not catch all failed commands and report them in the `batchFail.`*`timestamp`*`.txt` log file.	After you issue the iogmgmt -batch command, check the console for error messages and failed commands that might not be caught and reported in the batch command failure log.
Stopping callout threads takes a long time and the thread pool does not stop in the predefined 5 minutes.	If, for any reason, the thread pool does not stop gracefully, you can restart the SOAP Gateway server to restart the callout threads. However, after the restart, the first synchronous callout request on each tpipe to SOAP Gateway is returned to OTMA with a NAK response. Subsequent synchronous callout requests are processed normally. For IMS V13, with APAR PM90777 (socket listening enhancement) applied, after the restart, the first synchronous callout request on each tpipe to SOAP Gateway would no longer be returned to OTMA with a NAK response..

Table 2. Known issues and workarounds (continued)

Issue description	Workaround

SOAP Gateway does not support passing of time stamp information in the response messages for web services security if the IncludeTimestamp element is included in the server policy file.	You must modify your client applications or client policy files to not send the time stamp element in the request. Do not add the IncludeTimestamp element back to the server policy file.
	
	If you have server policy files from IMS Enterprise Suite V2.1 or V2.2 SOAP Gateway, ensure that the IncludeTimestamp element is removed or commented out:
	`<wsp:Policy>`
	`  <!-- commented out IncludeTimestamp`
	`  for V3.1 -->`
	`<!--<sp:IncludeTimestamp/>-->`
	`</wsp:Policy>`

If a custom fault message is configured and an error occurs, the error message that is provided by SOAP Gateway incorrectly includes the name of the XML converter for processing the requests rather than the name of the converter for the fault message, which causes confusion about the root cause of the problem. When a custom fault message is configured and an error occurs during request processing, the XML converter for request processing would call the fault converter. When such an error occurs, SOAP Gateway reports the incorrect converter name.	If you are getting an error message reporting issues with a converter and you have custom SOAP fault messages, check the IMS Connect console message for HWSA0380E to identify the converter that caused the failure.

For SOAP Gateway restrictions, see "SOAP Gateway restrictions" on page 9.

As problems are discovered and resolved, IBM Software Support updates the IBM Software Support knowledge base. By searching the knowledge base from the IMS Enterprise Suite Support Portal, you can find workarounds or solutions to problems.

New features in IMS Enterprise Suite Version 3.1 SOAP Gateway

IMS Enterprise Suite Version 3.1 SOAP Gateway adds the support for callout transaction logging, 64-bit z/OS operating environments, send-only with acknowledge for synchronous callout, and SOAP Gateway management utility batch mode.

Callout transaction logging support

Starting in V3.1.0.3 (APARs PI23543 and PI24654), SOAP Gateway generates a unique ID for every IMS callout request message to help track transactions. This vertical ID is used in the SOAP Gateway transaction log file if it is enabled. The vertical ID is also propagated to a remote IBM Tivoli® Composite Application Manager for Transactions (ITCAM) if one is configured.

SOAP Gateway generates a record for different event types that are triggered during callout request and response message processing. There are two ways to use the event information:

- The SOAP Gateway server generates a JSON log file that contains a record for each event. The log file includes the ID that is required to correlate the different events that are associated with a callout message.
- The SOAP Gateway server can send the message event data to a remote ITCAM data collector with the included implementation of the ITCAM Transaction Tracking API (TTAPI).

When transaction logging is enabled, events that are associated with both inbound web service requests (the provider scenario) and the outbound web service requests (the callout scenario) are logged to the same transaction log. To differentiate between the two scenarios, a`Scenario: "CONSUMER"` property is included in the event log for each callout message processing event.

64-bit support for z/OS

SOAP Gateway now runs on the z/OS platform in 64-bit mode, allowing organizations to take advantage of their 64-bit operating environment for extended memory usage.

Send-only with ACK support for synchronous callout

Send-only with acknowledgement protocol support for synchronous callout allows SOAP Gateway to receive a final confirmation that the response message was delivered to the original IMS application that issued the callout request. This confirmation provides SOAP Gateway users additional information about whether a callout response message was sent to IMS and whether IMS received the message.

SOAP Gateway management utility batch mode support

Administrators can now use the batch mode of the management utility to facilitate web service deployment and server management for better performance and manageability. Instead of issuing one command at a time, each with its own JVM instance, you can pass a file with a list of commands to the SOAP Gateway management utility iogmgmt -batch command for execution as a batch in one JVM instance.

Enhanced security cipher suite support

SOAP Gateway is enhanced to use the FIPS 140-2 approved cryptographic provider(s); IBMJCEFIPS (certificate 376) and/or IBMJSSEFIPS (certificate 409) for cryptography. The certificates are listed on the NIST web site at http://csrc.nist.gov/cryptval/140-1/1401val2004.htm. SOAP Gateway also adds the support for Transport Layer Security (TLS) V1.2 and for cipher suites with key length of 2048 and key strength of 112 bit, as required by NIST SP800-131a.

Related concepts:

Chapter 9, “Tracking and monitoring SOAP Gateway transactions,” on page 339
Transaction requests routed through a SOAP Gateway server can include a unique ID that correlates the transactions. Depending on the usage scenario, the transactions can be correlated between the client application (provider scenario), external web service (callout scenario), SOAP Gateway, or IMS (provider scenario).

"Send-only with acknowledgement protocol for web service consumer applications" on page 173
The send-only with acknowledgement protocol allows your application to receive a final confirmation that the response message was delivered to the original IMS application that issued the callout request.

"FIPS 140-2 and NIST SP800-131a" on page 39
Federal Information Processing Standards (FIPS) are standards and guidelines issued by the United States National Institute of Standards and Technology (NIST) for federal government computer systems. FIPS can be enabled for SOAP Gateway.

Related reference:

"-batch: Run management utility commands in batch mode" on page 430
The -batch command runs multiple SOAP Gateway management utility commands as a batch in one JVM instance.

New properties, commands, and messages

New properties, commands, and messages are added to support the new features.

New properties

A new correlator property, calloutSendOnlyWithAck, is added for the send-only with ACK support for synchronous callout.

New commands

A new SOAP Gateway management utility iogmgmt -batch command is added for the batch mode support.

New messages

The following new messages are added for the callout transaction logging support:

- "IOGS4014I" on page 415
- "IOGS7002W" on page 416
- "IOGS7003W" on page 416
- "IOGS7004W" on page 416

The following new messages are added for the send-only with ACK support for synchronous callout:

- "IOGS0081I" on page 408
- "IOGS0082E" on page 408
- "IOGS0083E" on page 409

The following new messages are added for the SOAP Gateway management utility batch mode support:

- "IOGD0758E" on page 400
- "IOGD0759E" on page 400
- "IOGD0760E" on page 400

The following new message is added for migration support:

- "IOGD0129E" on page 390

Related reference:

"-batch: Run management utility commands in batch mode" on page 430
The -batch command runs multiple SOAP Gateway management utility commands as a batch in one JVM instance.

SOAP Gateway restrictions

Check this list of restrictions before you start using SOAP Gateway.

SOAP Gateway has the following restrictions:

- Only non-conversational transactions are supported.
- Only commit mode 1, sync level NONE processing is supported.
- Message Format Service (MFS)-based transactions are not supported.
- Two-phase commit is not supported.
- SOAP Gateway supports XML messages that are encoded in UTF-8 only. The input XML transaction data inside the SOAP message must be encoded in UTF-8 and the output XML transaction data from the IMS application must also be encoded as UTF-8.
- SOAP Gateway supports Document-Literal style WSDL files only.
- For the asynchronous callout scenario and PL/I applications using the bottom-up approach in the provider scenario, the maximum size of the data structure is 32K (single segment).
- For the web service business event scenarios, only COBOL applications are supported (no PL/I applications).
- Multi-operation message support does not apply to the business event scenario. For the callout scenario, multi-operation is supported for only the top-down development approach that generates the COBOL synchronous callout application.
- Multi-segment IMS applications are supported only for the following scenarios:
  - When a PL/I application that is generated from a web service WSDL file using the top-down approach is enabled as a web service.
  - When a COBOL application is enabled as a web service using the bottom-up approach.
  - Synchronous callout applications (because the IMS Message Queue is bypassed).

  For all other scenarios, the maximum size is 32 KB for a single segment.

Restrictions for security support

- If you are using Java keystore (JKS) for SOAP Gateway security:
  - If multiple connection bundle entries point to the same IMS Connect endpoint (identified by the hostname and the port number), you must use the same JKS setting, with the same keystore and truststore.
  - For synchronous callout WS-Security support, the restriction of one keystore and one truststore per SOAP Gateway instance applies. That is, security certificates for external web service servers that IMS applications call out to must be stored in the same SOAP Gateway truststore.

  Recommendations:
  - Configure and run different SOAP Gateway instances, each with its own secure port, to service web service requests and IMS callout requests separately. Using separate Java keystores for the provider and the consumer scenarios improves the overall server performance and eases troubleshooting.

  - Use one connection bundle entry per IMS Connect endpoint.
- Use of the IBM z/OS Communications Server Application Transparent Transport Layer Security (AT-TLS) feature for server and client authentication is for the web service provider scenario only. For the consumer or business event scenarios, you must use System SSL.
- If NIST SP800-131A is required, you must use System SSL between SOAP Gateway and IMS Connect. You must apply the following fix, depending on the IMS version:
  - IMS V13 APAR PM96825
  - IMS V12 APAR PM98017
  - IMS V11 APAR PM98018

Restrictions for tracking IDs and logging

- The SOAP Gateway transaction tracking IDs, logging, and monitoring features support only the provider scenario.

Restrictions for Windows service support

- You cannot run a single installation as both a Windows service and a console application.
- The Windows service and the console application both use the same configurations for the run time as well as deployed services (artifacts).
- Only one instance of the SOAP Gateway Windows service can be installed per system.

Related concepts:

"Support for multi-segment messages" on page 122
You can enable multi-segment IMS COBOL or PL/I applications as web service providers by identifying the layouts of the input and output messages by using IBM Rational® Developer for System z®.

Related information:

Support for multi-segment message processing programs as web services for SOAP Gateway projects in IBM Rational Developer for System z
For more information on Rational Developer for System z support for multi-segment message processing programs as web services, see the Rational Developer for System z information center.

Information impact by feature

Impact to SOAP Gateway information is listed by feature.

New and changed information for callout transaction logging

These are the new and changed topics for callout transaction logging support.

Table 3. New and changed topics

Areas	Topics
Release information	**Release overview** • "Release notes for IMS Enterprise Suite V3.1 SOAP Gateway" on page 1 (changed) • "New features in IMS Enterprise Suite Version 3.1 SOAP Gateway" on page 6 (changed) • "New properties, commands, and messages" on page 8 (changed)

Table 3. New and changed topics (continued)

Areas	Topics
Concepts and tasks	**Chapter 9, "Tracking and monitoring SOAP Gateway transactions," on page 339** (changed) • "SOAP Gateway message processing events" on page 341 (changed) • "IDs for transaction correlation" on page 343 (changed) • "Remote monitoring options for SOAP Gateway" on page 344 (changed) • "Configuring the IBM Tivoli Composite Application Manager for Transactions (ITCAM) Transaction Tracking API (TTAPI)" on page 346 (changed) **Administering the SOAP Gateway server** • "SOAP Gateway logs" on page 303 – "Transaction log format" on page 307 - "Provider request transaction log format by event type" on page 307 (changed) - "Callout request transaction log format by event type" on page 314 (new section with a list of event types) – "Configuring the transaction log" on page 325 (changed)
Management utility	**SOAP Gateway management utility reference** • "-tracking: Configure SOAP Gateway-to-IMS transaction tracking IDs" on page 457 (changed) • "-tranLog: Configure the SOAP Gateway transaction logger" on page 460 (changed) • "-tranAgent: Configure the IBM Tivoli Composite Application Manager for Transactions (ITCAM) Transaction Tracking API (TTAPI)" on page 459 (changed)
Troubleshooting	**Troubleshooting** • **Diagnosing issues with callout and business event requests** – "Error status code returned to IMS during synchronous callout requests processing" on page 358 (changed) • **Messages for SOAP Gateway** – "IOGS4014I" on page 415 (new) – "IOGS7002W" on page 416 (new) – "IOGS7003W" on page 416 (new) – "IOGS7004W" on page 416 (new)

New and changed information for send-only with acknowledgement protocol support

These are the new and changed topics for send-only with acknowledgement protocol support.

Table 4. New and changed topics

Areas	Topics
General information updates	New topics: • "New features in IMS Enterprise Suite Version 3.1 SOAP Gateway" on page 6 • "Send-only with acknowledgement protocol for web service consumer applications" on page 173 Changed topics: • "Consumer-related correlation properties" on page 23 • "-corr: Create or update a correlator entry" on page 439 • "-prop: Set SOAP Gateway properties" on page 450 • "Sample correlator file" on page 24
Messages	New topics: • "IOGS0081I" on page 408 • "IOGS0082E" on page 408 • "IOGS0083E" on page 409

New and changed information for SOAP Gateway management utility batch mode

These are the new and changed topics for the SOAP Gateway management utility batch mode.

Table 5. New and changed topics

Areas	Topics
General concept, task, and background information	New and changed topics: • "New features in IMS Enterprise Suite Version 3.1 SOAP Gateway" on page 6 • "SOAP Gateway management utility" on page 26 • Chapter 8, "Administering the SOAP Gateway server," on page 291
Reference information	New topic: • "-batch: Run management utility commands in batch mode" on page 430
Messages	New messages: • "IOGD0758E" on page 400 • "IOGD0759E" on page 400 • "IOGD0760E" on page 400

New and changed information for security enhancements

These topics are new or changed for security enhancements.

Table 6. New and changed topics

Areas	Topics
General background and concept information	New and changed topics: • "Security support in SOAP Gateway" on page 30 • "FIPS 140-2 and NIST SP800-131a" on page 39 • "SOAP Gateway restrictions" on page 9 • "Security for the web service provider scenario" on page 123 • "Security features supported for the web service provider scenario" on page 125
Related tasks	New and changed topics: • "Configuring compliance for FIPS 140-2 and NIST SP800-131a" on page 97 • "Configuring SOAP Gateway on z/OS" on page 80 • "Configuring IMS Connect for SOAP Gateway" on page 101 • "Configuring SSL and HTTPS support with Java keystore (JKS)" on page 148 • "Creating the server keystore for SOAP Gateway and exporting the public key as a certificate" on page 150 • "Creating the server truststore for SOAP Gateway" on page 152 • "Exporting the certificate from IMS Connect" on page 152 • "Example: Configuring the client authentication and basic authentication security scheme" on page 192
Related reference	New and changed topic: • "-conn: Create, update, or delete a connection bundle" on page 435

New and changed information for installing V3.1 SOAP Gateway

These topics are new or changed for the advanced installation support feature.

Table 7. New and changed topics

Areas	Topics
General concept and background information	New and changed topics: • "New features in IMS Enterprise Suite Version 3.1 SOAP Gateway" on page 6 • "Planning for installation" on page 45 – "Installation process and general concepts" on page 46 – "IBM Installation Manager overview" on page 48 – "Configuration and setup planning" on page 55 – "Upgrading to Version 3.1 SOAP Gateway" on page 57

Table 7. New and changed topics (continued)

Areas	Topics
Installation and configuration tasks	New and changed topics for installation on z/OS: • "Installing SOAP Gateway on z/OS" on page 60 • "Scenario 1. IBM Installation Manager for z/OS is not installed on the target system" on page 62 1. "Installing IBM Installation Manager for z/OS on the target system" on page 63 2. "Installing SOAP Gateway on z/OS by using IBM Installation Manager" on page 67 3. "Worksheet for installation scenario 1" on page 69 • "Scenario 2. IBM Installation Manager for z/OS is installed on the target system" on page 73 1. "Transferring the sample installation jobs to the target system" on page 75 2. "Installing SOAP Gateway on z/OS by using IBM Installation Manager" on page 67 3. "Worksheet for installation scenario 2" on page 78 • "Sample jobs for installation and configuration" on page 84 • "Installing SOAP Gateway on distributed platforms" on page 87 (changed) • "Installing SOAP Gateway using IBM Installation Manager" on page 90 (changed) • "Installing SOAP Gateway in silent mode" on page 91 Changed topics for configuration tasks: • "Configuring SOAP Gateway on z/OS" on page 80 • "Installing multiple SOAP Gateway server instances that share one JVM" on page 110
Uninstallation	• "Removing SOAP Gateway" on page 117

New and changed information for migration support

These topics are new or changed for migration support.

Table 8. New and changed topics

Areas	Topics
Related tasks	New topics: • "Migrating from IMS Enterprise Suite Version 2.1 SOAP Gateway" on page 104 • "Migrating from IMS Enterprise Suite Version 2.2 SOAP Gateway" on page 106
Reference	Changed topic: • "-migrate: Migrate and upgrade SOAP Gateway" on page 448
Messages	Changed message: • "IOGD0129E" on page 390

General release-related information changes

These topics are new or changed for general release updates and other release-related changes.

Table 9. New and changed topics

Areas	Topics
General release updates	New topics for release overview: • "Release notes for IMS Enterprise Suite V3.1 SOAP Gateway" on page 1 • "New features in IMS Enterprise Suite Version 3.1 SOAP Gateway" on page 6 • "New properties, commands, and messages" on page 8 New and changed topics for system requirements, software requirements, and restrictions: • "System requirements" on page 41 – "Disk space for installation on z/OS" on page 42 – "Disk space for installation on distributed platforms" on page 43 • "Software requirements" on page 43 • "SOAP Gateway restrictions" on page 9 • "Cloning the SOAP Gateway server" on page 108 • "Installing multiple SOAP Gateway server instances that share one JVM" on page 110 • "Specifying the Java SDK location" on page 95

Chapter 2. IMS Enterprise Suite SOAP Gateway overview

IBM IMS Enterprise Suite SOAP Gateway is a web services solution that enables IMS applications to interoperate outside of the IMS environment through the SOAP protocol to provide and request services that are independent of platform, environment, application language, or programming model.

SOAP Gateway can assist an organization in the following areas:

- Enterprise modernization
- Application development
- Business-to-business (B2B) integration
- Service-oriented architecture (SOA) implementation

SOAP Gateway enables your IMS application as a web service provider. Different types of client applications, such as Microsoft .NET, Java, and third-party applications, can submit SOAP requests to IMS to drive the business logic of the IMS applications. With the IMS Connect XML adapter, you can enable your IMS application to become a web service without the need of changing the backend IMS application.

Secured communication is supported between SOAP Gateway and the client that issues the request to access an IMS application. Secured communication is also supported between SOAP Gateway and IMS Connect. SOAP Gateway supports web services security (WS-Security), where user identity or security token is authenticated on a per-message basis. Authentication information can also be passed on a per-web service basis, where the user information is defined in the connection bundle for the web service.

With SOAP Gateway, IMS application can also participate in business events processing and monitoring. Business analysts and developers can monitor IMS business activities and enhance IMS business logic with the IBM business events processing and monitoring engines such as WebSphere® Business Events and WebSphere Business Monitor. The only scenario in which SOAP Gateway supports the REST protocol is when the business event processing engine is IBM WebSphere Business Monitor.

SOAP Gateway also enables your IMS application as a web service consumer. Your IMS applications can make a callout request to access external web service providers and get responses back.

SOAP Gateway is compliant with the industry standards for web services, including SOAP/HTTP 1.1 and Web Services Description Language (WSDL) 1.1. This compliance enables your IMS assets to interoperate openly with various types of applications.

SOAP Gateway components

SOAP Gateway includes a server component that processes web service requests between IMS applications and external applications or web services, a utility for deploying web services and managing server properties, and an administrative console for verifying installation and deployed web services.

SOAP Gateway server

The SOAP Gateway server acts as the gateway between external web services and IMS applications. It serves as a web service server where IMS applications are enabled as web services. It serves a web service consumer when it forwards IMS application callout requests or business event data to external web services or event processing services.

In all scenarios, the SOAP Gateway server acts as a client to IMS Connect.

SOAP Gateway provides these functions by offering direct SOAP access to existing IMS transactions. SOAP Gateway also enables IMS transactions to request for external web services or emit business event data through the SOAP protocol. SOAP Gateway communicates with IMS through IMS Connect, the TCP/IP gateway for IMS. Messages between IMS Connect and SOAP Gateway can be transmitted in XML format through Secure Sockets Layer (SSL). IMS Connect converts the XML data into bytes and passes the request to the IMS application through IMS Open Transaction Manager Access (OTMA).

IMS OTMA is a transaction-based, connectionless protocol that addresses the problem of connecting a client to a server so that the client can support a large network, or many sessions, while maintaining high performance. IMS Connect is a client to OTMA, and the SOAP Gateway server is a client to IMS Connect.

SOAP Gateway communicates with its client through the HTTP protocol. HTTPS is used for secure communications.

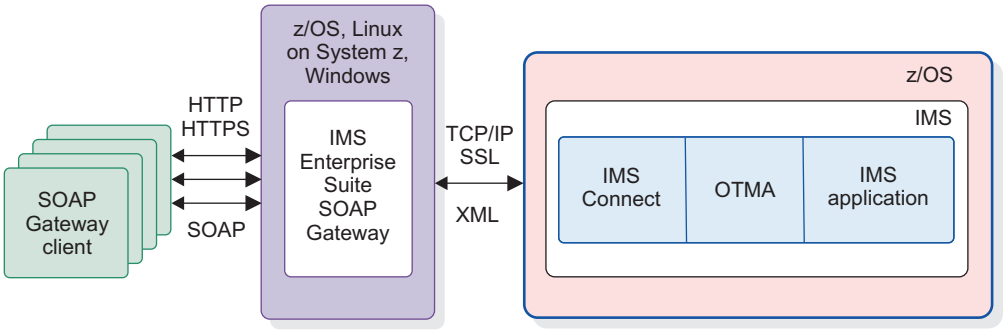


Figure 1. SOAP Gateway system layout and setup

SOAP Gateway can be installed on z/OS in the same or different LPAR as IMS. This setup simplifies the infrastructure and improves the efficiency.

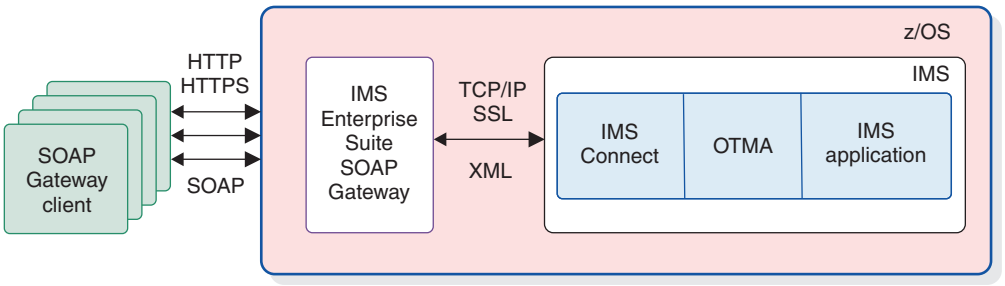


Figure 2. SOAP Gateway system layout and setup on z/OS

You can have a sysplex distributor environment where traffic to SOAP Gateway can be distributed to different installations to help balance the load.

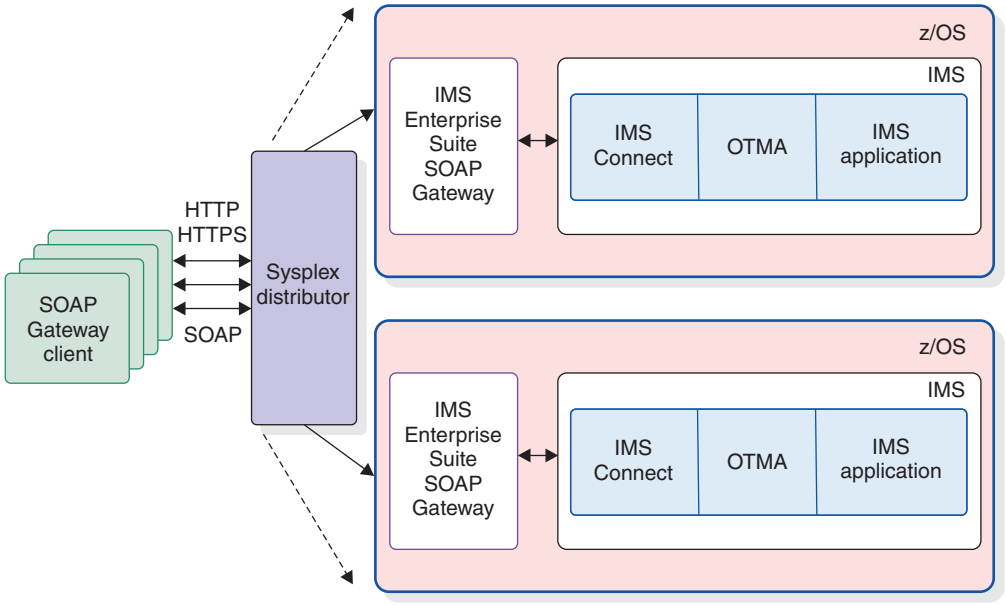


Figure 3. SOAP Gateway system layout and setup on z/OS with a sysplex distributor

When the SOAP Gateway server handles callout requests from IMS applications to external web services, it becomes a client to the external web application server. Regardless, SOAP Gateway communicates with the external web server using the same protocol, and SOAP Gateway remains a client to IMS Connect.

Related concepts:

Chapter 8, “Administering the SOAP Gateway server,” on page 291
Administer the SOAP Gateway server with the SOAP Gateway management utility.

XML-formatted IMS messages

SOAP Gateway sends and receives messages in XML.

IMS requires that all transaction messages be prefixed by a 2-byte LL field, a 2-byte ZZ field, and the transaction code. For messages to flow between your IMS application and an external SOAP Gateway client (such as a web service or a Java application), when you use IBM Rational Developer for System z to generate the XML converter from your application source file, the LL field, the ZZ field, and the transaction code are handled for you.

When you generate a PL/I application from a web service WSDL file by using the top-down development approach in Rational Developer for System z, Rational Developer for System z also generates an XML converter for each operation defined in that WSDL file (that is, it is a one-to-one mapping between each operation and the converter). The required LL and ZZ fields are also handled for you.

At run time, SOAP Gateway adds the simple EBCDIC byte values of the transaction code and LL and ZZ fields to an XML-formatted IMS data input message.

To ensure that the messages are sent and received in the cases you expect, use the EDIT keyword of the IMS TRANSACT macro statement. The EDIT keyword specifies whether the transaction is uppercase (UC) or a mix of upper- and lowercase (ULC).

Related reference:

"Modifying the IMS application for XML messages" on page 224
If you are *not* using the IMS Connect XML adapter function for data transformation, you must modify the IMS application by using the provided guidelines and samples.

Web services description language (WSDL) file

A WSDL file is an XML document that describes a web service.

WSDL files are used by others (for example, the client that invokes the service) to discover the service and to understand how to invoke the service. The WSDL file specifies the location of the service and the operations that the service exposes.

To make your IMS application accessible as a web service (the provider scenario), you can import your COBOL or PL/I application into Rational Developer for System z to generate a WSDL file. The generated WSDL file describes the functions in your application and how the input and output messages are structured in order to invoke the function.

To make a callout request from your IMS application to a web service (the consumer scenario), you must import the WSDL file of the web service to SOAP Gateway.

The WSDL file serves as the web service interface for the IMS application.

Connection bundle properties

The connection bundle specifies the connection and security properties for SOAP Gateway when it communicates with IMS Connect.

The connection bundle properties refer to the location and other parameters that are required for creating a TCP/IP socket connection to IMS. In this case, SOAP Gateway is the client, and IMS Connect is the TCP/IP server.

These connection properties are in a connection bundle XML file. The connection bundle XML file contains all of the connection bundle entries for the server. You can create as many connection bundle entries as needed. However, each web service can be associated with only one connection bundle entry at a time, and a connection bundle entry name must be specified in the correlator file for that web service.

Important: Use the SOAP Gateway management utility to create, inquire, modify, or delete the connection bundle entries. Manual creation or modification can result in unexpected errors.

Each connection bundle entry consists of the following connection and security properties:

- **Connection bundle name:** A name to identify this connection bundle entry, or set of connection properties.
- **IMS Connect hostname:** The name or IP address of the host system where IMS Connect is running.
- **IMS Connect port number:** The port number of IMS Connect.

- **IMS Connect data store ID:** The name of the target IMS data store. The ID must match the ID parameter of the data store statement that is specified in the IMS Connect configuration member. It also serves as the XCF member name for IMS during internal XCF communications between IMS Connect and IMS OTMA.
- **IMS RACF user ID:** The security authorization facility (SAF) user name to use for the connection to IMS. When WS-Security is disabled, SOAP Gateway passes this user ID to IMS Connect for authentication.
- **IMS RACF password:** The security authorization facility (SAF) password to use for the connection to IMS. When WS-Security is disabled, SOAP Gateway passes this user password to IMS Connect for authentication.
- **IMS RACF group name:** The security authorization facility (SAF) group name to use for the connection to IMS.

The following optional properties can be specified for SSL connections based on Java keystore (JKS) to support the web service provider scenario, where SOAP Gateway is the client to IMS Connect.

Important: Leave these properties blank if you are using IBM z/OS Communications Server AT-TLS feature to secure the connection between SOAP Gateway and IMS Connect. Specifying JKS SSL properties in AT-TLS environment will cause SSL handshake to fail.

- **SSL keystore name:** Specifies the fully qualified path name of the keystore in which trusted certificates and private keys are stored.
- **SSL keystore password:** Specifies the password for the keystore. The password length must be 6 - 20 alphanumeric characters.
- **SSL truststore name:** Specifies the fully qualified path name of the truststore in which trusted certificates are stored.
- **SSL truststore password:** Specifies the password of the truststore in which trusted certificates are stored. The password length must be 6 - 20 alphanumeric characters.
- **SSL encryption level:** Specifies the encryption type. A value of **Strong** indicates that a strong cipher suite needs to be used. A value of **Weak** indicates that a weak cipher suite needs to be used. A value of **None** indicates that no encryption is used. The **None** encryption level is used only for authentication.

The following properties support the IMS applications as web service consumers (callout) scenario. They also apply to the scenario where IMS applications emit business event data to external business event processing engines.

- **SOAP Gateway callout basic authentication user ID:** Specifies the user ID to send to the server that hosts the web service for basic authentication.
- **SOAP Gateway callout basic authentication password:** Specifies the password, for the user ID to send to the server that hosts the web service for basic authentication.
- **SOAP Gateway callout truststore name:** Specifies the fully qualified path name of the truststore on SOAP Gateway that stores the certificates of trusted external web service servers. Required for client or server authentication.
- **SOAP Gateway callout truststore password:** Specifies the password of the truststore on SOAP Gateway in which trusted external web service server certificates are stored.
- **SOAP Gateway callout keystore name:** Specifies the fully qualified path name of the keystore on SOAP Gateway that stores the trusted client certificates for a callout application to authenticate with a target web service. Required for client authentication.

- **SOAP Gateway callout keystore password:** Specifies the password of the keystore on SOAP Gateway. Required for client authentication.
- **Callout TPIPE names:** An element for storing tpipe names that are associated with the connection bundle. This property is used by SOAP Gateway to open a connection to the comma-separated list of tpipes in order to pull messages for the callout function.

Related concepts:

"Connection bundle management" on page 296
Manage connection bundle entry names and usage by web services to ensure stable and predictable SOAP Gateway server behavior.

Related tasks:

"Creating a connection bundle entry for callout applications" on page 266
Create a connection bundle entry that describes the connection properties for accessing IMS by using the SOAP Gateway management utility. The connection bundle entries are stored in the `connbundle.xml` file.

Related reference:

"-conn: Create, update, or delete a connection bundle" on page 435
Use the -conn command to create, update, or delete a connection bundle.

Correlator file

The correlator file specifies transaction and runtime properties. This file also specifies the information that SOAP Gateway needs to match incoming requests to the appropriate backend IMS application and outgoing requests from an IMS application to a web service.

These properties are stored in an XML file. The file name is determined by the combination of the target namespace, operation name, and service name. This file is also created by IBM Rational Developer for System z.

Important:

- If you use the IMS Connect XML adapter function, use Rational Developer for System z to generate the correlator file and the XML converter driver(s).
- If you are not using the IMS Connect XML adapter, use the SOAP Gateway management utility to create the correlator file.

Use the SOAP Gateway management utility to view, modify, or delete the correlator file. Manual creation or modification can result in unexpected errors.

Related tasks:

"Creating a correlator file for a callout application" on page 239
You can manually create a correlator file with the SOAP Gateway management utility if you do not have IBM Rational Developer for System z.

General correlation properties:

The general correlation properties specify transaction and runtime properties, and the information that SOAP Gateway needs to match the messages between the IMS application and the web service.

A correlator file has three elements:

wsdlFile
: Defines the web service name and target namespace.

serviceTraceLevel
: This element is reserved for future use.

correlatorEntry

Each correlator entry is defined by operation name and service name. A correlator file might contain multiple correlator entries, with one entry for each operation.

Target namespace, service name, and operation name together are the unique identifier for a web service.

A correlator entry contains the following properties:

- **XML adapter type (adapterType):** The type of XML adapter that is used by IMS Connect. The IMS Connect XML adapter is the only adapter supported.
- **XML converter name (converterName):** The XML converter driver program name generated by Rational Developer for System z. This driver program is used by the IMS Connect XML adapter to transform XML data into bytes for your IMS application. This parameter value is required if you are using the IMS Connect XML adapter function.
- **Connection bundle name (connectionBundleName):** The name of the connection bundle that contains the connection and security properties for connection with the IMS.
- **Socket timeout (socketTimeout):** The timeout value for SOAP Gateway to receive a response from IMS Connect.
- **Execution timeout (executionTimeout):** The timeout value for IMS Connect to send a message to IMS and receive the response.
- **Lterm name (ltermName):** The name of the logical terminal.
- **IMS transaction code (trancode):** The transaction code of the IMS application that is invoked by the web service. This property is required when IMS applications are enabled as web service providers. It is also used to handle response messages when IMS applications are web service consumers in an asynchronous request-response interaction.
- **WS-Security (WSSecurity):** The security token type to use for web services security for the provider scenario.
- **Inbound tpipe name (inboundTPIPEName):** The name of the tpipe for inbound messages from IMS. This property is not used by SOAP Gateway.

Two other properties, extendedProperty1 and extendedProperty2 are reserved properties.

- extendedProperty2 is unused at this time.

For a detailed description of each property, including valid values and the default value, see the SOAP Gateway management utility iogmgmt -corr command syntax reference.

Related reference:

"-corr: Create or update a correlator entry" on page 439
Use the -corr command to create or update the transaction and runtime properties of a correlator entry.

Consumer-related correlation properties:

SOAP Gateway provides a set of correlation properties to support the consumer scenario through the IMS callout function.

- **Callout connection bundle names (calloutConnBundleNames):** The names of the callout connection bundles that contain the connection and security properties that are used to connect with the IMS.

- **Callout web service WSDL file name (wsdlFile):** The name of the WSDL file of the web service to which an IMS application makes a callout request.
- **Callout web service invocation timeout (calloutWSTimeout):** The amount of time in milliseconds that SOAP Gateway must wait to receive a response from the web service.
- **Callout WS-Security (calloutWSSecurity):** The security token type to use for WS-Security for the consumer scenario.
- **Send-only with acknowledgement protocol (calloutSendOnlyWithAck):** A Boolean value that determines if SOAP Gateway sends the callout response message to IMS using the send-only with acknowledgement protocol. If this property is set to true, SOAP Gateway uses the send-only protocol with acknowledgement. The final ACK or NACK message from IMS is not forwarded to the external client application. The ACK or NACK responses are logged by SOAP Gateway if the server log level is set to 2 (for NACK responses) or 4 (for ACK and NACK responses).

A callout request is typically to a specific service that is invoking a specific operation on that service. The service name and operation name attributes of the correlator entry are used to identify the callout web service. The target name space, operation name, and service name together form the unique identifier for the correlator entry.

For a detailed description of each property, including valid values and the default value, see the SOAP Gateway management utility iogmgmt -corr command syntax reference.

Related reference:

"-corr: Create or update a correlator entry" on page 439
Use the -corr command to create or update the transaction and runtime properties of a correlator entry.

Business events-related correlation properties:

For business event data, only a subset of the callout properties is used.

- **Callout location URI (calloutURI):** The URL address for WebSphere Business Monitor. This property applies only when the business event is to be processed by WebSphere Business Monitor. The value of this property is specified in Rational Developer for System z when you generate the request XSD and correlator file from the IMS application copybook file.
- **Callout connection bundle names (calloutConnBundleNames):** The names of the callout connection bundles that contain the connection and security properties that are used to connect with the IMS.
- **Callout web service invocation timeout (calloutWSTimeout):** The amount of time in milliseconds that SOAP Gateway must wait to receive a response from WebSphere Business Monitor. This property is not used when the business event processing server is WebSphere Business Events.

Related reference:

"-corr: Create or update a correlator entry" on page 439
Use the -corr command to create or update the transaction and runtime properties of a correlator entry.

Sample correlator file:

This sample correlator file for a web service provider scenario demonstrates the elements in the correlator XML schema.

```
<?xml version="1.0" encoding="UTF-8"?>
<COR:correlator mode="call_in" INSTALLATION="" UUID=""
  version="3.0" xmlns:COR="http://www.ibm.com/IMS/Correlator"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.ibm.com/IMS/Correlator correlator.xsd ">
  <wsdlFile name="IMSSOAPIVP.wsdl" targetNamespace="http://ivp.soap.ims.ibm.com/"/>
  <serviceTraceLevel>Error</serviceTraceLevel>
  <correlatorEntry operationName="runIMSSOAPIVP" portName="IMSSOAPIVPPort"
   serviceName="IMSSOAPIVPService">
    <adapterType>No Adapter</adapterType>
    <converterName></converterName>
    <connectionBundleName>imssoapivp</connectionBundleName>
    <socketTimeout>0</socketTimeout>
    <executionTimeout>0</executionTimeout>
    <ltermName></ltermName>
    <inboundTPIPEName></inboundTPIPEName>
    <inboundCCSID>1208</inboundCCSID>
    <hostCCSID>1140</hostCCSID>
    <outboundCCSID>1208</outboundCCSID>
    <trancode></trancode>
    <calloutConnBundleNames></calloutConnBundleNames>
      <calloutWSTimeout>7500</calloutWSTimeout>
      <calloutSendOnlyWithAck>false</calloutSendOnlyWithAck>
      <WSSecurity></WSSecurity>
    <calloutURL></calloutURL>
    <extendedProperty1></extendedProperty1>
    <extendedProperty2></extendedProperty2>
  </correlatorEntry>
</COR:correlator>
```

Important: You must use Rational Developer for System z Version 9.0.1 or later to generate the correlator files in correlator schema version 3.0 that is required by IMS Enterprise Suite Version 3.1. Correlator files that are generated by older versions of Rational Developer for System z are in schema version 2.0 and must be migrated by using the SOAP Gateway management utility iogmgmt -migrate command.

SOAP Gateway master configuration and runtime configuration

The master configuration is the authoritative configuration of the SOAP Gateway server, and is stored in the file system. The active server configuration in memory is the runtime configuration.

The *master configuration* is based on the file system and stores all the correlator files, connection bundles, WSDL files, and server properties. When you deploy a web service or callout application, the generated web service Axis Archive files (AAR files) are also stored in the master configuration. The master configuration is directly updated by the SOAP Gateway management utility.

The *runtime configuration* is the active server configuration in memory.

When the SOAP Gateway server starts, it loads the server information and validates the web service information in the master configuration. SOAP Gateway scans through the correlator files and web service archive files, and validates the connection bundle entries. If all required information for a web service exists and is valid, the runtime configuration is loaded into the cache, and the web service is deployed. Storing the runtime configuration information in the cache reduces file system I/O and shortens server response time. Storing only valid configuration information in the cache further ensures the efficiency.

For callout applications or business event applications, SOAP Gateway scans through the correlator files and validates the related WSDL or XSD information. If all required information for a callout or business application is valid, the

configuration is loaded into the runtime configuration and the cache, and the callout or business event application is deployed.

When you deploy a web service or callout application, all the associated service WSDL or XSD file, correlator file, and connection bundle must be valid. Any web service, callout application, or callout properties-related changes, including stopping and starting callout threads and thread pool, made with the SOAP Gateway management utility take effect in the runtime cache immediately. However, the following changes do not take effect until the server is restarted:

- Updating SOAP Gateway server properties
- Updating the connection bundle name in a correlator file
- Updating the details for a connection bundle that is referenced in a correlator file (such as changing the hostname, port number, or datastore name)

Important: When you update any callout properties or applications, you must restart the callout thread pool and callout threads.

Related concepts:

Chapter 8, "Administering the SOAP Gateway server," on page 291
Administer the SOAP Gateway server with the SOAP Gateway management utility.

Related tasks:

"Deploying a web service to SOAP Gateway" on page 327
A web service must be deployed to the SOAP Gateway server before it is available to client applications.

SOAP Gateway management utility

The SOAP Gateway management utility provides a command line or batch interface for configuring server properties, managing the server run time, and working with web service artifacts.

The SOAP Gateway management utility lets you manage the SOAP Gateway server and web services through command-line interface that facilitates task automation and management flexibility.

With the SOAP Gateway management utility, you can do the following:

- Start and stop the SOAP Gateway server on Windows systems.

  Exception: For z/OS, use the START and STOP console commands to run the AEWIOGPR sample job after this job is configured as a started task.
- Configure SOAP Gateway server properties.
- Enable your IMS application as a web service provider or a web service consumer (callout), or to emit business event data.
- Configure web services, callout applications, and business event emitters.
- Remove web services, callout applications, and business event emitters.
- Configure callout properties, and manage the callout threads and thread pool.

The SOAP Gateway management utility also supports running multiple commands in batch mode. This feature facilitates service deployment and server administration tasks by executing multiple commands in one JVM instance.

Important: Use the SOAP Gateway management utility to create, view, modify, or delete the connection bundle entries, correlator files, and the SOAP Gateway server properties. Manual creation or modification can result in unexpected errors.

Restriction: z/OS For SOAP Gateway servers running on z/OS, the SOAP Gateway management utility must run on the same LPAR as the target server.

SOAP Gateway administrative console

The SOAP Gateway administrative console lists the deployed web services when the server is started. Each item in the list is a link to the web services description language (WSDL) file for the web service.

Only IMS applications that are enabled as web services are listed. External web services that IMS applications issue callout requests to are not included.

Related tasks:

"Deploying the web service" on page 230
Deploying the web service to SOAP Gateway enables the application and allows it to begin processing client requests.

"Viewing deployed web services" on page 295
Start the SOAP Gateway Administrative Console to view your deployed web services, or use the SOAP Gateway management utility.

SOAP Gateway architecture

The SOAP Gateway installation is composed of three distinctive parts for flexible installation and for ease of system maintenance. This architecture provides flexibility in server installation and maintenance, allowing the system administrator to install SOAP Gateway on different mount points or directories.

The three-part architecture separates the binary files that run the SOAP Gateway server and the management utility from server configuration files, and user files such as web services-related artifact files. The following diagram shows this architecture in the file system:

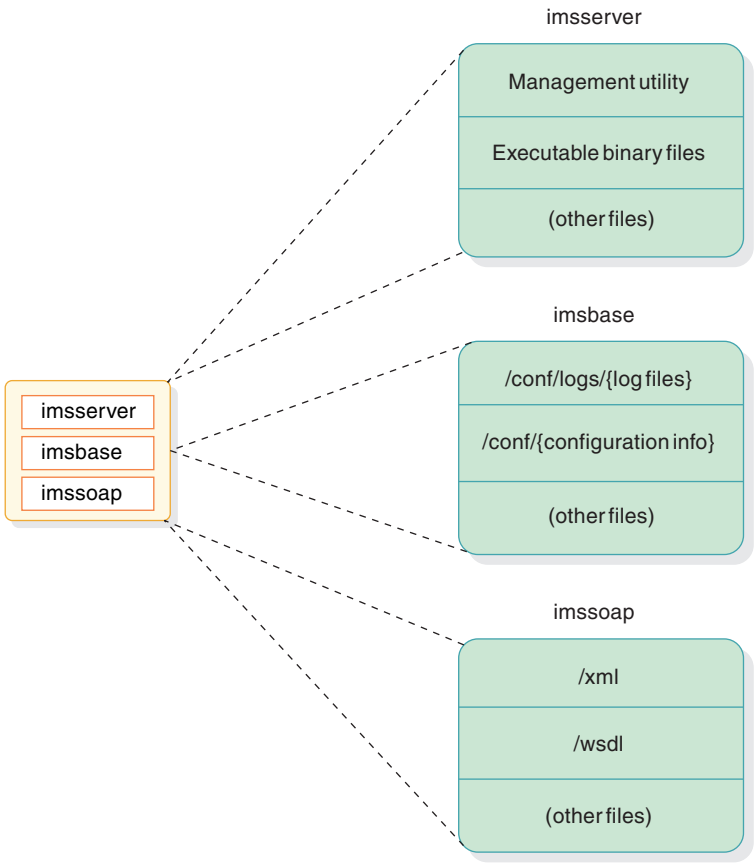


Figure 4. SOAP Gateway system layout for installation

imsserver
The *imsserver* component contains the SOAP Gateway management utility and other executable files that the SOAP Gateway server runs on.

imsbase
The *imsbase* component contains the server configuration files and various log files.

imssoap
The *imssoap* component contains user-deployed web service-related files, such as the correlators, WSDL files, and other Java class files or libraries for custom authentication modules.

This separation is specified during installation by specifying the location for each of the three parts by using the IBM Installation Manager. If no custom location is specified, by default all parts are installed under the same directory or mount point. The three parts, after installation, must be able to communicate with each other, such as over a shared drive, or in the same LPAR or Sysplex. The **imsserver** component must be able to access the files in the other two components as if they were local.

The support for installing the three parts of SOAP Gateway under different directory or mount point provides several advantages:

- User files or web service-related artifacts can be stored separately from the server executable files and configuration information. Subsequent application of services or installation of maintenance releases do not affect user files.

- Installation of the **imsserver** component can be done in READ ONLY mode. Because no log files or user data are written to this component, system security is enhanced.
- When server log files or web service artifacts increase in size, additional storage can be allocated without shutting down the server.

Supported scenarios and features

SOAP Gateway supports three usage scenarios: web service provider, web service consumer, and business event scenarios.

Web service provider scenario

An IMS application can be enabled as a web service that provides services or data to service requesters.

When an IMS application is deployed as a web service provider, the SOAP Gateway server receives the SOAP message from the client application, converts it to an IMS input message, and sends it to IMS. The server then receives the output message from IMS and converts it to a SOAP message to return to the client.

With the tooling support in IBM Rational Developer for System z, the required web services description language (WSDL) file and XML converters can be generated from an COBOL or PL/I application. This approach is known as the *bottom-up* development scenario in Rational Developer for System z. You can also create a WSDL file that describes the web service in Rational Developer for System z, and then use the *top-down* development scenario in Rational Developer for System z to generate a PL/I application. The PL/I application can then be deployed to IMS, and enabled as a web service on SOAP Gateway by using the same WSDL file that was used to generate the application.

Web service consumer scenario

An IMS application can issue a callout request to a web service through SOAP Gateway. When an IMS application is enabled as a web service consumer, it can invoke either a one-way web service operation or a request-response operation. The IMS application uses different calls to initiate a web service request and to indicate, if a response is expected, whether the response is to be processed in the same transaction (also known as *synchronous callout*) or a different transaction (also known as *asynchronous callout*).

When an IMS application is deployed as a web service consumer, the SOAP Gateway server checks with IMS Connect to determine if any callout request messages are in the hold queue. When the server receives an IMS callout request from IMS Connect, SOAP Gateway parses the message and converts it to a SOAP message to invoke the web service. If a response is returned from the web service, SOAP Gateway converts the SOAP message and sends the response to IMS by invoking a new IMS transaction using the send-only protocol. Alternatively, SOAP Gateway can use the send-only protocol with acknowledgement to request a final acknowledgement from IMS that the message was received. The final acknowledgement message is not passed to the client application.

With the tooling support in Rational Developer for System z, you can generate the required XML converter and the correlator file by mapping the input and output data between the web service WSDL file and the IMS application. This approach is

known as the *meet-in-middle* development scenario in Rational Developer for System z.

Business event scenario

When an IMS application emits business event data to event processing services through the SOAP Gateway server, the SOAP Gateway server checks with IMS Connect to determine if any business event messages are in the hold queue. When an IMS business event message from IMS Connect is received, SOAP Gateway parses the message and converts it to an XML message to send to the business event processing engine in either the SOAP or Representational State Transfer (REST) protocol.

The only scenario in which SOAP Gateway supports the REST protocol is when the business event processing engine is IBMWebSphere Business Monitor.

With the tooling support in Rational Developer for System z, you can use the *meet-in-middle* approach to generate the required XML converter and the data correlation file by mapping the input and output data between the web service WSDL or XML schema (XSD) file and the IMS application.

Related concepts:

Chapter 4, "Design and implementation by usage scenario," on page 119
SOAP Gateway supports three usage scenarios: web service provider, web service consumer, and business event.

Security support in SOAP Gateway

SOAP Gateway supports HTTPS communication with its clients, and SSL communications with its host, IMS Connect.

You can configure SOAP Gateway with standards that are specified by the US Department of Commerce National Institute of Standards and Technology (NIST) to define security requirements for encryption.

- Federal Information Processing Standards (FIPS) 140-2 requires that the Transport Layer Security (TLS) protocol and the cryptographic modules are certified.
- SP800-131a requires stronger cryptographic algorithms and key lengths that are used in FIPS 140-2 cryptographic modules.

Server authentication, client authentication, and basic authentication

When an IMS application is enabled as a web service provider or a consumer, SOAP Gateway supports both server authentication and client authentication.

- *Server authentication* is the provision of server authentication information (digital certificate), from the server to the client, that binds the server identify to subsequent communications.
- *Client authentication* is the provision of authentication information from the client to the server. Client authentication is also referred to as *mutual authentication*, because server authentication is required in order to support client authentication.

Certificates are exchanged at the transport level to establish trust before the connection is established or a web service is invoked.

For the web service consumer scenario, SOAP Gateway supports *basic authentication* where the server that hosts the web service requires SOAP Gateway, the client, to have appropriate basic authentication credentials in order to call a service.

The business event scenario is a special case of the consumer scenario, where an IMS application emits business event data to a business event processing engine by using the asynchronous callout function. The security support for the business event scenario is the same as the support for asynchronous callout in the consumer scenario.

WS-Security and custom authentication modules for the web service provider scenario

For the provider scenario, SOAP Gateway supports authentication of users on a per-web service or per-message basis. When the user ID and password information is provided by the connection bundle, the authentication is performed on a per-web service basis. All requests use the same user ID to access IMS. Security certificates can be sent at the transport level for server authentication and client authentication. When users are authenticated on a per-message basis, user ID and password information is enclosed as tokens in the WS-Security header in each message. Requests might come from different user IDs. This feature is known as web service security or *WS-Security*.

SOAP Gateway supports the following security tokens for WS-Security:

- UsernameToken Profile 1.0 user name tokens
- Security Assertion Markup Language (SAML) 1.1 and 2.0 unsigned sender-vouches tokens
- SAML 1.1 and 2.0 signed sender-vouches tokens with two trust types:

  Trust any
  Any valid security certificate in the SOAP header is allowed.

  Trust one
  The security certificate in the SOAP header must be configured with the server truststore path and password.

Only one security token type is allowed per web service. When WS-Security is enabled, you can also provide your own custom authentication module to perform additional checking by using a Java Authentication and Authorization Service (JAAS) module.

WS-Security and custom authentication modules for the synchronous callout scenario

For the synchronous callout scenario, SOAP Gateway supports callout requests to web services that require authentication of users on a per-web service or per-message basis. When users are authenticated on a per-message basis, user ID is enclosed as a token in the WS-Security header in each message. SOAP Gateway supports the following for WS-Security security tokens for the synchronous callout scenario:

- SAML 1.1 unsigned sender-vouches tokens
- SAML 2.0 unsigned sender-vouches tokens

Only one security token type is allowed per callout request. When WS-Security is enabled, you can also provide your own custom authentication module to perform additional checking by using a JAAS module.

Security support by web service scenario

The following table describes the supported SOAP Gateway security features by scenario:

Table 10. Security support in SOAP Gateway by web service scenario

Web service scenario	Key type	Authentication type	WS-Security	Custom authentication module for WS-Security
Provider	• Java keystore (JKS) • System Authorization Facility (SAF) **Important:** SAF is for the z/OS platform only. Use of SAF requires the AT-TLS feature in IBM z/OS Communications Server or later.	• Server authentication • Client authentication	UsernameToken Profile 1.0 tokens **Important:** Without server or client authentication, user name and password are transmitted in clear text. Use of server or client authentication is recommended.	Server authentication or client authentication is required.
			SAML 1.1 unsigned tokens SAML 1.1 signed tokens SAML 2.0 unsigned tokens SAML 2.0 signed tokens	Client authentication is required.
Consumer	JKS	• Basic authentication • Server authentication • Client authentication	For the synchronous callout scenario only, the following security token types are supported: • SAML 1.1 unsigned tokens • SAML 2.0 unsigned tokens Client authentication is required for the SAML token support.	For synchronous callout only, if client authentication is configured.
Business event	JKS	• Basic authentication • Server authentication • Client authentication	Not applicable.	Not applicable.

Related concepts:

"Security for the web service provider scenario" on page 123
SOAP Gateway provides support for both server authentication and client authentication and web-services security (WS-Security) for the web service provider scenario regardless of the platform that SOAP Gateway runs on.

"Security for the consumer (callout) scenario" on page 182
Security support for the callout scenario is provided for messages from IMS to SOAP Gateway through SSL, and from SOAP Gateway to the web service through HTTPS.

"Security for business event requests" on page 201
Security support is provided for business event messages from IMS to SOAP Gateway through Secure Sockets Layer (SSL), and from SOAP Gateway to the business event servers through HTTPS.

Secure sockets layer (SSL) and Transport Layer Security (TLS)

SSL provides security for your interactions by securing the TCP/IP connection between SOAP Gateway and IMS Connect.

TLS is the successor to the SSL protocol. TLS V1.0 was the first version, succeeding SSL V3.0. New TLS versions continue to be defined by the Internet Engineering Task Force (IETF), and the TLS protocol maintains compatibility modes for the earlier SSL protocol.

The TLS protocol defined in RFC 2246 provides communications privacy over the Internet. The protocol enables client and server applications to communicate in a way that is designed to prevent eavesdropping, tampering, and message forgery. The client contacts the server by sending a communication known as a handshake, which enables the client and server to authenticate to each other and specify the type of encryption that is used during the session. All data exchanged between the client and server during the session is encrypted and cannot be read by a third party. In addition, the protocol includes a message integrity check to ensure the integrity and reliability of transmitted data.

The term *SSL* is often used to refer to this entire family of protocols. Unless otherwise specified, this convention is used in this set of information.

With the evolution of the web and on-demand information, data security has become critical for Internet users. The Secure Sockets Layer (SSL) protocol ensures that the transfer of sensitive information over the Internet is secure. SSL protects information from:

- Internet eavesdropping
- Data theft
- Traffic analysis
- Data modification
- Trojan horse browser or server

SOAP Gateway communicates with IMS Connect through TCP/IP sockets. SSL can be used to secure the TCP/IP communication between the two entities. The SSL support provided by SOAP Gateway and IMS Connect uses a combination of public and private keys with symmetric key encryption schemes to achieve client and server authentication, data confidentiality, and integrity. SSL rests on top of TCP/IP communication protocol and allows an SSL-enabled server to authenticate itself to an SSL-enabled client and vice versa.

For the SSL connection between SOAP Gateway and IMS Connect, SOAP Gateway is considered the client and IMS Connect is considered the server. After authentication is complete, the server and client can establish an encrypted connection that also preserves the privacy and integrity of the data.

SOAP Gateway supports the use of the Application Transparent TLS (AT-TLS) feature in IBM z/OS Communications Server for SSL protocol handling. Therefore, any SSL or TLS version that the AT-TLS feature supports, including TLS V1.0, TLS V1.1 and SSL V3.0, are supported.

SSL concepts

Key SSL concepts and terms include *certificates*, *certificate authority*, *keystores*, *truststores*, and *keyrings*.

Certificates

A digital certificate is a digital document that validates the identity of the owner of the certificate. A digital certificate contains information about its owner, such as its name, company, and public key. The certificate is signed with a digital signature by the Certificate Authority (CA), which is a trustworthy authority.

Certificate authority

A Certificate Authority (CA) is a trusted party that creates and issues digital certificates to users and systems. The CA, as a valid credential, establishes the foundation of trust in the certificates. The major task of a trusted CA is to map an identity, such as a host name, to a specific public/private key pair in order to build trust. The CA itself has its own self-signed public/private key pair. As with any public/private key pair the private key is kept secret. Certificates issued by the CA are signed with the private key of the CA, and the authenticity of a certificate can be verified by using the public key of the CA, which is available in the CA's certificate.

Certificate management

Certificates and private keys are stored in files called *keystores*. A keystore is a database of key material. Keystore information can be grouped into two categories: key entries and trusted certificate entries. The two entries can be stored in the same keystore or separately in a keystore and *truststore* for security purposes. Keystores and truststores are used by both the SSL client, SOAP Gateway, and the SSL server, IMS Connect.

Keystore
A keystore holds key entries, such as the private key of the user. For example, it holds the private key of SOAP Gateway.

Truststore
A truststore is a keystore that holds only certificates that the user trusts. An entry should be added to a truststore only if the user makes a decision to trust that entity. An example of a SOAP Gateway truststore entry is the certificate of the target SSL server, IMS Connect.

Keyrings

A keyring is a named collection of certificates and Certificate Authorities that is associated with a specific user. A certificate is identified by its label and the keyring to which it is connected. A keyring is only valid in conjunction with a System Authorization Facility (SAF) user ID. A SAF keyring is a keystore and a truststore for an SSL application on z/OS. You can create multiple keyrings with the same name, but assigned to different user IDs.

For example, the keyring named Z1Keyring that is associated with the started task control (STC) user ID soapadmin is different from the Z1Keyring keyring that is associated with the user ID serveradmin.

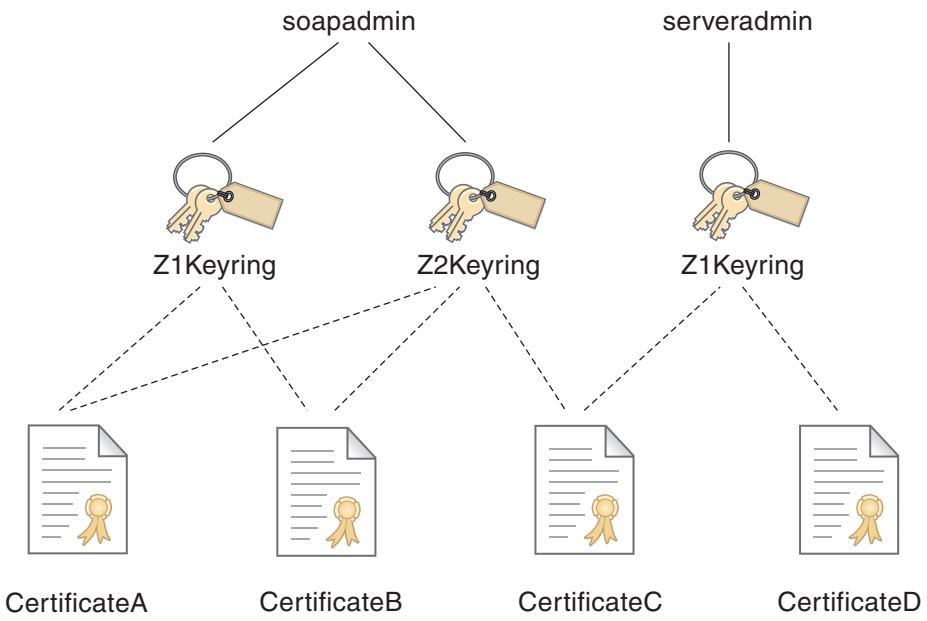


Figure 5. Two keyrings of the same name, but assigned to different users

Related concepts:

IMS Connect SSL connections (IMS Version 13)
For more information about IMS Connect SSL connections, see IMS Version 13 Communications and connections information.

"Security support through z/OS Communications Server Application Transparent Transport Layer Security (AT-TLS)" on page 37
SOAP Gateway on the z/OS platform uses the IBM z/OS Communications Server AT-TLS feature for the web service provider scenario to achieve SSL protection of its communication sessions.

"System SSL" on page 37
System SSL, a feature of the Cryptographic Services base element of z/OS, provides a complete SSL/TLS implementation and a full set of APIs that allow z/OS client and server applications to enable SSL/TLS protection for their TCP network traffic.

"SAF and RACF" on page 39
System Authorization Facility (SAF) is an open standard to access security services on z/OS.

SSL session and handshake process

The SSL protocol consists of server authentication, and client authentication (optional but strongly recommended) followed by an encrypted conversation. The following scenario steps through the establishment of an SSL session.

SSL server authentication allows a client to confirm the identity of a server. SSL-enabled client software uses standard techniques of public-key cryptography to ensure that the certificate and public ID of a server are valid and that the certificate and ID were issued from one of the trusted certificate authorities (CA) of the client.

SSL client authentication allows a server to confirm the identity of a client. Using the same techniques that are used for server authentication, SSL-enabled server

software verifies that the certificate and public ID of a client are valid and that the certificate and ID were issued by one of the trusted certificate authorities (CA) of the server.

The use of HTTPS on the client causes an SSL session to be established between the client and server. When client authentication is configured, the SSL handshake provides mutual authentication (the server authenticates the client and the client authenticates the server) based on each endpoint's digital certificate.

SSL handshake

Both the client, SOAP Gateway, and the server, IMS Connect, store their certificates and private keys in keyrings. The SSL session between SOAP Gateway and IMS Connect is established by following a handshake sequence between the client and the server. The sequence varies depending on whether the server is configured to just provide only a server certificate, or to provide a server certificate and request a client certificate, and which cipher suites are available to be used. A *cipher suite* specifies a combination of cryptographic algorithms for peer authentication, data authentication, and data encryption. The SSL protocol determines how the client and server negotiate the cipher suite to be used, authenticate one another, transmit certificates, establish session keys, and transmit messages. Some of the algorithms used in cipher suites include:

- Peer authentication
  - DSA - Digital Signature Algorithm
  - RSA - A public key algorithm for both encryption and authentication
  - DH - Diffie-Hellman
- Message authentication and integrity
  - MD5 - Message Digest algorithm
  - SHA-1 - Secure Hash Algorithm 1
  - SHA-2 - Secure Hash Algorithm 2
- Message privacy (encryption)
  - AES - symmetric encryption and decryption
  - DES - Data Encryption Standard
  - Triple-DES - DES applied three times.
  - RC2 and RC4 - Rivest encryption ciphers
  - SKIPJACK - A classified symmetric-key algorithm implemented in FORTEZZA-compliant hardware

After the handshake completes successfully, all of the data that flows over the new SSL session is protected by the cryptographic algorithms and session keys that were agreed to during the handshake. In most cases, this protection means that all data is encrypted to ensure privacy and authenticated to ensure it came from the expected sender, and integrity is checked to ensure the data was not modified during transmission.

SSL Version 2, SSL Version 3 and the TLS protocols support overlapping sets of cipher suites. Administrators can enable or disable any of the supported cipher suites for both clients and servers. When a particular client and server exchange information during the SSL handshake, the client and server identify the strongest enabled cipher suites that they have in common and use one of them for the SSL session.

System SSL

System SSL, a feature of the Cryptographic Services base element of z/OS, provides a complete SSL/TLS implementation and a full set of APIs that allow z/OS client and server applications to enable SSL/TLS protection for their TCP network traffic.

In addition to providing the API interfaces to exploit the SSL and TLS protocols, System SSL also provides a suite of Certificate Management APIs and a certificate management utility.

The SSL support in IMS Connect is implemented based on System SSL.

IBM z/OS Communications Server also provides a feature called Application Transparent TLS (AT-TLS) which allows z/OS applications to take advantage of the capabilities of System SSL transparently, without requiring any application code changes by pushing the System SSL calls down into the TCP layer of the network stack.

For SSL setup and initialization of IMS Connect, see IMS Connect SSL connections in *IMS Version 13 Communications and Connections*.

Related concepts:

"Secure sockets layer (SSL) and Transport Layer Security (TLS)" on page 33
SSL provides security for your interactions by securing the TCP/IP connection between SOAP Gateway and IMS Connect.

"SSL concepts" on page 33
Key SSL concepts and terms include *certificates*, *certificate authority*, *keystores*, *truststores*, and *keyrings*.

"SSL session and handshake process" on page 35
The SSL protocol consists of server authentication, and client authentication (optional but strongly recommended) followed by an encrypted conversation. The following scenario steps through the establishment of an SSL session.

"Configuring AT-TLS for SOAP Gateway" on page 128
To configure the IBM z/OS Communications Server AT-TLS feature for SOAP Gateway, use the IBM Configuration Assistant for z/OS Communications Server V1R13.

"Security process flow with AT-TLS for the web service provider scenario" on page 127
For the web service provider scenario, the IBM z/OS Communications Server AT-TLS feature can be set up to provide security for web services hosted by the SOAP Gateway server.

"SAF and RACF" on page 39
System Authorization Facility (SAF) is an open standard to access security services on z/OS.

Security support through z/OS Communications Server Application Transparent Transport Layer Security (AT-TLS)

SOAP Gateway on the z/OS platform uses the IBM z/OS Communications Server AT-TLS feature for the web service provider scenario to achieve SSL protection of its communication sessions.

AT-TLS Overview

The Transport Layer Security (TLS) protocol defined in RFC 2246 provides communications privacy over the Internet. The protocol enables client and server applications to communicate in a way that is designed to prevent eavesdropping, tampering, and message forgery. AT-TLS consolidates TLS implementation in one location, reducing or eliminating application development and maintenance resources and workload.

AT-TLS offers an application-transparent approach to secure the connection socket and data exchange. The application of the SSL and TLS protection is pushed down the network stack into the TCP layer, where AT-TLS invokes System SSL on behalf of the applications.

AT-TLS supports all the System SSL functions, including client authentication (often referred to as mutual authentication), certificates with SAF keyrings, and certificate revocation list (CRL) validation with a Lightweight Directory Access Protocol (LDAP) server. When AT-TLS is enabled, the TCP/IP stack uses policies configured through the Policy Agent to apply SSL protection to TCP traffic based on the characteristics of that traffic (such as local address and port, and remote address and port).

Advantages of AT-TLS

Use of AT-TLS greatly simplifies the security implementation and provides greater flexibility. For example, connections that are secured by SSL would be handled by AT-TLS and are transparent to SOAP Gateway. Another major benefit is the ease to manage the SSL setup.

SOAP Gateway and IMS Connect that run on the z/OS platform in the same LPAR can take advantage of this security infrastructure simplification. The z/OS TCP/IP or network security administrator would configure the connectivity rules in AT-TLS for the end points it runs on, such as SOAP Gateway address and port, and/or IMS Connect address and port

AT-TLS makes full use of hardware cryptography features of System z. In addition, z/OS Communications Server V1R11 Policy Agent (PAGENT) manages the rules and policies that define how SSL is used to connect to SOAP Gateway, providing additional functions, including:

- AT-TLS uses RACF® keyrings and certificates.
- Certificate revocation list (CRL)
- Connection settings such as connection refreshes
- Creation of separate connectivity and security rules for connections between SOAP Gateway and IMS Connect

Related concepts:

"Secure sockets layer (SSL) and Transport Layer Security (TLS)" on page 33
SSL provides security for your interactions by securing the TCP/IP connection between SOAP Gateway and IMS Connect.

"SSL concepts" on page 33
Key SSL concepts and terms include *certificates*, *certificate authority*, *keystores*, *truststores*, and *keyrings*.

"SSL session and handshake process" on page 35
The SSL protocol consists of server authentication, and client authentication (optional but strongly recommended) followed by an encrypted conversation. The

following scenario steps through the establishment of an SSL session.

"Configuring AT-TLS for SOAP Gateway" on page 128
To configure the IBM z/OS Communications Server AT-TLS feature for SOAP Gateway, use the IBM Configuration Assistant for z/OS Communications Server V1R13.

"Security process flow with AT-TLS for the web service provider scenario" on page 127
For the web service provider scenario, the IBM z/OS Communications Server AT-TLS feature can be set up to provide security for web services hosted by the SOAP Gateway server.

"SAF and RACF"
System Authorization Facility (SAF) is an open standard to access security services on z/OS.

SAF and RACF

System Authorization Facility (SAF) is an open standard to access security services on z/OS.

SAF provides an installation with centralized control over system security processing through a system service called the MVS™ router. A common practice among z/OS administrators is to use Resource Access Control Facility (RACF), which is based on SAF, as the certificate authority to generate and sign personal certificates for their internal systems or applications. RACF, as the z/OS security manager, is responsible for making all access control decisions in z/OS.

When IMS Connect is configured to call RACF, IMS Connect can validate the user IDs and passwords on incoming messages with RACF directly. When you use the IBM z/OS Communications Server AT-TLS feature for SOAP Gateway security on a z/OS system, AT-TLS also uses RACF keyrings and certificates.

FIPS 140-2 and NIST SP800-131a

Federal Information Processing Standards (FIPS) are standards and guidelines issued by the United States National Institute of Standards and Technology (NIST) for federal government computer systems. FIPS can be enabled for SOAP Gateway.

FIPS 140-2 is a cryptographic function validation program defining security standards for cryptographic modules used in IT software. For more information on these standards, see the National Institute of Standards and Technology web site.

SOAP Gateway includes cryptographic modules Java Secure Socket Extension (JSSE) and Java Cryptography Extension (JCE), which have undergone FIPS 140-2 certification. These modules can be enabled when FIPS 140-2 compliance is required.

On top of FIPS, NIST SP800-131a requires longer key lengths and stronger cryptography. For NIST SP800-131a requirements, SOAP Gateway adds the following support:

- TLS V1.2
- Cipher suites with key length of 2048 and key strength of 112 bit

For NIST SP800-131a:

- For both the provider and consumer scenarios, you must use System SSL for secure communications with IMS Connect.

- You must use SSL between SOAP Gateway and its client applications (the provider scenario) or any external web services (the callout scenario).

Related tasks:

"Configuring compliance for FIPS 140-2 and NIST SP800-131a" on page 97
You can configure SOAP Gateway to communicate with its clients and IMS Connect over secure sockets by using Java Secure Socket Extension files that are required by FIPS 140-2. In addition, NIST SP800-131a requires the use of TLS V1.2.

Related information:

National Institute of Standards and Technology web site
See the National Institute of Standards and Technology web site for more information about FIPS 140-2 and NIST SP800-131a.

Chapter 3. Installing and configuring SOAP Gateway

For both z/OS and distributed systems, use the IBM Installation Manager to install IMS Enterprise Suite SOAP Gateway. After installation, some configuration options might be necessary, depending on your environment and requirements.

IMS Enterprise Suite Version 3.1 SOAP Gateway supports migration of web services and server properties from Version 2.1 and Version 2.2 SOAP Gateway. If you have an earlier version of SOAP Gateway installed, you must manually migrate your web services and server properties to Version 3.1.

- For z/OS, IMS Enterprise Suite can be ordered through Shopz as a CBPDO.
- For Windows and Linux on System z, SOAP Gateway can be downloaded from the IMS Enterprise Suite download site

The following table shows where you can obtain SOAP Gateway based on your platform.

Table 11. Sources for obtaining SOAP Gateway and subsequent fixes or enhancements

Code to install	Platform	Order through Shopz or service stream as SMP/E installable	Download from the IMS Enterprise Suite website
IBM Installation Manager	z/OS	Yes, included when you order IMS Enterprise Suite.	No
	Distributed	No	Yes, available on the download page.
SOAP Gateway	z/OS	Yes	No
	Distributed	No	Yes
Subsequent APARs or their equivalent update for the distributed platforms	z/OS	Yes	No
	Distributed	No	Yes

System requirements

IBM IMS Enterprise Suite V3.1 SOAP Gateway supports z/OS, Linux on System z, and Microsoft Windows.

Supported platforms

- For the z/OS platform, z/OS V1.13 or later, as long as the version is supported by IBM, for IMS Version 13, Version 12, and Version 11
- Linux on System z
- Microsoft Windows

Disk space for server operation and installation

The SOAP Gateway installation consists of three components: **imsserver**, **imssoap**, and **imsbase**. The installation location for each component can be different. On z/OS, each component can be installed on a different mount point.

For the **imsserver** component contains the SOAP Gateway management utility and other executable files. No files are written to this component directory, and therefore the disk space requirement is constant.

For the **imssoap** component where the web service artifacts are stored, you must allocate additional disk space for web service artifacts based on your business needs. The disk space required per web service varies. In most cases, each web service is about 50 to 100 KB, but complex web services could require more space.

For the **imsbase** component where the log files are stored, the trace level setting affects the size of the log file. If the transaction log is enabled, each transaction record uses approximately 300 bytes of disk space.

Disk space requirements for installation are different depending on the platforms.

z/OS For the z/OS platform, you can allocate a primary allocation and a secondary allocation for the **imssoap** and **imsbase** components to avoid the need to reallocate when you run out of disk space.

Related tasks:

The AGGRGROW PARM on the MOUNT command on zFS
See the z/OS V1R13 Distributed File Service zSeries File System Administration for more information about creating and managing zFS file systems using compatibility mode aggregates.

Disk space for installation on z/OS

z/OS

For z/OS, refer to the *Program Directory for the IMS Enterprise Suite* and the *Program Directory for IBM Installation Manager* for the storage requirements of the SMP/E process. Additional disk space is required after the SMP/E processing.

After the SMP/E process completes, you need the following disk space to install both the Installation Manager and SOAP Gateway.

Table 12. SOAP Gateway disk space requirements for installation on z/OS

Component	Required disk space for post-SMP/E installation
IBM Installation Manager	• 2500 cylinders (2 GB) to install. • If the Installation Manager is not yet installed, and the target system is different from the SMP/E driving system: – A temporary file system of 400 cylinders (300 MB) is needed on both the driving and the target systems to store the installation kit PAX file in order to transfer the installation kit. – 600 cylinders (450 MB) are needed on the target system to store the expanded installation kit.

Table 12. SOAP Gateway disk space requirements for installation on z/OS (continued)

Component	Required disk space for post-SMP/E installation
SOAP Gateway	The following disk space is required for the post-SMP/E installation. • 140 cylinders (~105 MB) to install SOAP Gateway: – The **imsserver** component: 100 cylinders (75 MB) – The **imsbase** component: 20 cylinders (15 MB). **Important:** You must allocate more disk space because log files size could increase significantly depending on log level setting. Transaction logs, if enabled, also requires disk space. – The **imssoap** component: 20 cylinders (15 MB). You must allocate more disk space to allow for additional web services deployment. • 900 cylinders (650 MB) to install IBM Java SDK Version 7.

When the file is mounted on zSeries File System (zFS), you can use the AGGRGROW PARM on the MOUNT command for an aggregate to be dynamically grown if the file becomes full:

```
mount filesystem('xxxx') mountpoint('/yyy') type(zfs) mode(rdwr) parm('aggrgrow')
```

See z/OS V1R13 *Distributed File Service zSeries File System Administration* (SC24-5989) for more information.

Related information:

z/OS V1R13.0 Distributed File Service zSeries File System Administration
Topics in this document provide complete and detailed guidance and reference information for system administrators that work with the zSeries File System (zFS) component of the IBM z/OS Distributed File Service base element.

Disk space for installation on distributed platforms

Linux Windows

You need to plan for enough disk space to download and install the IBM Installation Manager and the IBM IMS Enterprise Suite V3.1 SOAP Gateway.

Depending on your installation scenario and anticipated use of the server, the disk space requirement varies.

Software requirements

IMS Enterprise Suite Version 3.1 SOAP Gateway requires IMS Version 13, Version 12, or Version 11.

Supported IMS versions

IMS Enterprise Suite Version 3.1 supports IMS Version 13, Version 12, and Version 11.

Java requirement

For the z/OS platform, SOAP Gateway runs on both IBM 64-bit and IBM 31-bit Java software development kit (SDK), Java Technology Edition, Version 7.

For distributed platforms, SOAP Gateway runs on IBM 31-bit Java software development kit (SDK), Java Technology Edition, Version 7.

- For z/OS environments, the Java SDK that is provided with the IMS Enterprise Suite Base Services component has been deprecated. See the PSP bucket for the latest supported Java version and download instructions. The instructions also specify how to order Java for z/OS through Shopz at no charge.
- For distributed environments, visit the IMS Enterprise Suite downloads page and navigate to the page for SOAP Gateway. Follow the instructions on the page to download the latest supported Java version.

Requirements for send-only with acknowledge support for synchronous callout

You must apply the fix for IMS V13 APAR PM91344 or V12 APAR PM91443 for this feature.

Requirements for transaction tracking IDs and logging

- IMS Version 12 or later

  Requirement: IMS Version 12 with service for APAR PM69983 and APAR PM76333 applied to the target IMS Connect host is required to use horizontal IDs.
- IBM Tivoli Composite Application Manager for Transactions Application Program Interface V7.3 or later

Required versions of Rational Developer for System z

If you are using IBM Rational Developer for System z to generate web service artifacts, Rational Developer for System z Version 9.0 or later is required.

Rational Developer for System z has a Eclipse-based development environment (or client) for SOAP Gateway web service artifact generation, and a server or host portion that must be configured. The host portion consists of permanently active tasks and ad-hoc started tasks to allow the client to work with the various components of your z/OS host, such as MVS data sets, TSO commands, z/OS UNIX files and commands, job submit, and job output.

To use the IMS Connect XML converter function for SOAP Gateway, the Rational Developer for System z SFEKLMOD module must be concatenated to the STEPLIB of the IMS Connect startup JCL to use the XML converter function. The SFEKSAMP library and the SFEKLMOD module are required at the XML converter compilation time, and the SFEKLMOD module is also required at run time.

Important:

- The correlator schema has changed in IMS Enterprise Suite Version 3.1 SOAP Gateway to schema version 3.0.
  - When you upgrade to IMS Enterprise Suite Version 3.1, migration of the correlator files is handled as part of web service migration process when you use the SOAP Gateway management utility iogmgmt -migrate *path_to_source_installation* command.
  - To generate correlator files in schema version 3.0, use Rational Developer for System z Version 9.0.1 or later. Correlator files that are generated by older versions of Rational Developer for System z are in correlator schema version 2.0 and must be migrated. You must run the iogmgmt -migrate correlator

command to migrate the correlator files in the SOAP Gateway server XML directory to the new schema before deploying the web services.
  - Rational Developer for System z Version 9.0.1 or later generates correlator files in version 3.0 schema only. If you still need to generate correlator files in version 2.0 schema for existing production environment while testing the Version 3.1 of SOAP Gateway, stay with Rational Developer for System z Version 9.0. When version 3.0 schema is needed, use the Version 3.1 SOAP Gateway migration utility iogmgmt -migrate correlator command to migrate the correlator files to version 3.0 schema.
- If you plan to use the top-down provider scenario that generates a PL/I application from a web service WSDL file, the generated application is based on a set of segmentation APIs. The IRZPWSIO segmentation APIs are included in Rational Developer for System z V9.0 and older versions. In Rational Developer for System z V9.0.1 or later, the APIs are removed, and are now included in IMS V12 (APAR PM97469) and IMS V13 (APAR PI17898). The IRZPWSIO segmentation APIs are renamed to DFSPWSIO in IMS. The IRZPWSH include file is renamed to DFSPWSH in IMS.

Requirements for business events

The SOAP Gateway support for business events requires the following software:

- For WebSphere Business Events:
  - WebSphere Business Events Development Toolkit Version 6.2 or later installed in a compatible version of IBM Rational Application Developer for WebSphere Software
  - WebSphere Business Events Server Version 6.2 or later
- For WebSphere Business Monitor:
  - WebSphere Business Monitor Development Toolkit Version 6.2 or later installed in a compatible version of Rational Application Developer
  - WebSphere Business Monitor Server Version 6.2 or later

Planning for installation

Before you install SOAP Gateway, you need to understand the general installation process, and analyze your potential usage to determine how you can take advantage of its three-part architecture. You also need to plan for the use of IBM Installation Manager for installation on all supported platforms.

Examine the following topics to understand:

- The general installation process, and the purposes of using the IBM Installation Manager to install SOAP Gateway on all supported platforms, including the z/OS.
- The concepts required to use the IBM Installation Manager.
- The differences between SMP/E driving system and the target system.
- Things to consider to determine your system setup.
- Required User ID access permission for the SMP/E installation process.
- How to upgrade to this release if you have an older version.
- Skills that are required to install and configure SOAP Gateway.

Installation process and general concepts

Before you install SOAP Gateway, you need to analyze your potential usage to determine how you can take advantage of its three-part architecture. You also need to plan for the use of IBM Installation Manager for installation on all supported platforms.

Three-part architecture for installation and maintenance flexibility

An installation of IMS Enterprise Suite Version 3.1 SOAP Gateway consists of three parts that can be installed in different directories (or mount points on z/OS). This three-part architecture separates the binary files that run the SOAP Gateway server and the management utility from server configuration files and user files such as web services-related artifact files. This separation makes it easier to apply maintenance and allocate additional disk space when more web services are added.

Before you install SOAP Gateway, analyze potential usage, workload, and the number of SOAP Gateway instances you need. Check the system requirements and software requirements, and examine the SOAP Gateway architecture to determine how you want to install the three parts in anticipation of future growth.

Use of the IBM Installation Manager for installation

IBM Installation Manager must be installed on the system that SOAP Gateway is to be installed on, regardless of the platform. Use of IBM Installation Manager for z/OS for SOAP Gateway installation is new starting from IMS Enterprise Suite Version 2.2.

z/OS For the z/OS platform, IBM Installation Manager for z/OS is included when you order the IMS Enterprise Suite from the Shopz website.

Linux Windows For Linux on System z and Windows, IBM Installation Manager is available for download on the same site where you download SOAP Gateway.

Understanding of the IBM Installation Manager terminology, installation modes, and user modes is essential to a successful installation of SOAP Gateway.

SMP/E installation and IBM Installation Manager installation

SMP/E is a tool for managing the installation of software products on your z/OS system and tracking the modifications that you make to those products. The instructions to SMP/E consist of a series of control statements, called modification control statements (or MCSs). SMP/E can also receive SYSMODs packaged in a relative file (RELFILES) format from DASD or tape.

- You must use SMP/E to process the IMS Enterprise Suite Base Services and SOAP Gateway function modification identifiers (FMIDs). After SMP/E processing, the following components are obtained on the file system:
  - Sample jobs for installing SOAP Gateway and for transferring files, including the IBM Installation Manager installation kit, if the system where SOAP Gateway is to be installed is different from the SMP/E driving system.
  - IBM Java SDK Version 7

Important: The IBM Java SDK in the IMS Enterprise Suite Base Services component, which is referred to in the program directory, has been deprecated (APAR PI33917). See the PSP bucket for the latest supported Java version and download instructions. The instructions also specify how to order Java for z/OS through Shopz at no charge.

  - SOAP Gateway repository file. This repository file is a compressed file that IBM Installation Manager uses to install the product. It contains the product code and metadata about how to install and lay out the selected packages on the system.

- You must also use SMP/E to process IBM Installation Manager if you do not have it already. After SMP/E processing, the installation kit that contains a set of files in UNIX System Services file structures and a set of sample installation jobs for installing (or "creating") the Installation Manager are obtained on the file system.

The SMP/E process puts the product installation code onto the file system for further installation. The products are not yet ready to run.

The following figure describes how the components are installed through the SMP/E process.

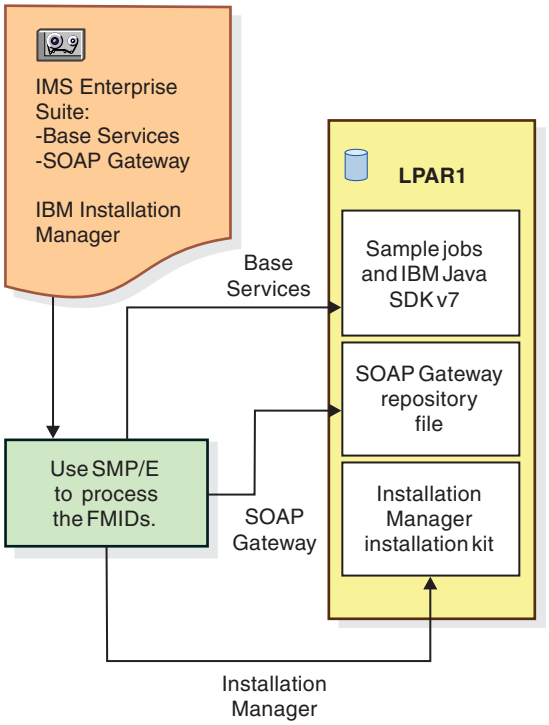


Figure 6. SOAP Gateway SMP/E installation process

In IBM Installation Manager documentation, the post SMP/E step of the installation is called "Creating an Installation Manager." For the Installation Manager and SOAP Gateway, which run on z/OS UNIX System Services instead of MVS services, the SMP/E process puts the installation code onto the file system. The Installation Manager is not yet ready for use until it is created or installed. After the Installation Manager is installed, it can be used to install SOAP Gateway and other products.

You install the Installation Manager by editing and submitting the provided sample jobs (these jobs start with the letters GIN). You use these jobs to:

- Allocate the required file system, create the directory structure, and mount the file system
- Issue an installation command to install (or create) the Installation Manager onto the created file system and directory

After IBM Installation Manager is installed, use the set of AEWTSxxx sample jobs provided in the IMS Enterprise Suite Base Services to install SOAP Gateway. The IMS Enterprise Suite Base Services FMID includes a set of sample JCL jobs for:

- Transferring the IBM Installation Manager installation kit to the system where SOAP Gateway is to be installed if the target system is different from the SMP/E driving system
- Installing SOAP Gateway by using the IBM Installation Manager command mode

Related concepts:

"SOAP Gateway architecture" on page 27
The SOAP Gateway installation is composed of three distinctive parts for flexible installation and for ease of system maintenance. This architecture provides flexibility in server installation and maintenance, allowing the system administrator to install SOAP Gateway on different mount points or directories.

IBM Installation Manager overview

IBM Installation Manager is a general-purpose software installation and update tool that runs on a range of operating systems.

You normally need only one Installation Manager on a system because one Installation Manager can keep track of multiple product installations. If different products have different required versions of Installation Manager, use the most current version because later versions are compatible with earlier versions.

The Installation Manager graphical user interface is not available on z/OS. All interaction with Installation Manager on z/OS is done through the OMVS command line or JCL

For more information about using Installation Manager, see the IBM Installation Manager Version 1.5 Information Center.

How IBM Installation Manager simplifies installation and maintenance

IBM Installation Manager simplifies the overall installation and maintenance process, and provides better coordination among base products, maintenance releases, and products that might share resources.

IBM Installation Manager supports the installation and maintenance process by installing from an *installation repository*. A repository is a location that stores data for installing, modifying, rolling back, updating, or uninstalling packages. A repository contains a `repository.config` file that stores the metadata on how to install and lay out the selected packages on the system.

For SOAP Gateway, the installation repository is provided as a compressed file, or a repository file, that contains the product code and all the metadata that Installation Manager uses for installation.

For the z/OS platform, IBM Installation Manager runs as a UNIX System Services application, and can be started from the UNIX System Services shell, shell scripts, or MVS batch jobs. Each LPAR that SOAP Gateway is installed on must has an instance of the Installation Manager installed unless these LPARs are on the same sysplex. The benefit of using the Installation Manager for SOAP Gateway installation is that it is easier to install and apply updates to SOAP Gateway on multiple server instances. When multiple SOAP Gateway instances are installed on multiple LPARs, Instead of using SMP/E to process and install the FMIDs or APARs on each LPAR, you complete the SMP/E process one time. Then you can run the Installation Manager on each target system (or LPAR) to install from this repository.

Installation packages

Each software product that can be installed with Installation Manager is typically referred to as a *package*. For IMS Enterprise Suite Version 3.1 SOAP Gateway, to support flexible installation on multiple mount points, each of the three SOAP Gateway components (**imsserver**, **imsbase**, and **imssoap**) and the required Java SDK is a package.

When you install SOAP Gateway by using the Installation Manager, you must install all three SOAP Gateway components together. For each component, you can specify an installation directory. For the Java SDK:

- For z/OS environments, the Java SDK that is provided with the IMS Enterprise Suite Base Services component has been deprecated. See the PSP bucket for the latest supported Java version and download instructions. The instructions also specify how to order Java for z/OS through Shopz at no charge.
- For distributed environments, visit the IMS Enterprise Suite downloads page and navigate to the page for SOAP Gateway. Follow the instructions on the page to download the latest supported Java version.

User modes for installing IBM Installation Manager for z/OS

The Installation Manager can be installed, or created, in three user modes:

group mode
: This is the recommended mode for installing Installation Manager and SOAP Gateway. There is no limit to the number of group-mode Installation Managers that you can have on a system. In group mode, the Installation Manager can be invoked by any user ID that is connected to the owning group for the Installation Manager (the default group of the user ID that creates it). Using the group mode to install the Installation Manager means that you can install SOAP Gateway by using any user ID in the group.

admin mode
: In admin mode, the Installation Manager is installed from a superuser ID (uid=0) and can be invoked from any superuser ID. Using the admin mode to install the Installation Manager means that you must install SOAP Gateway by using a superuser ID. There can be only one admin-mode Installation Manager on a system.

user mode
: In user mode (also called nonAdmin mode), the Installation Manager can be invoked only by the user that installed it. There can be only one user-mode Installation Manager for a user.

Installation modes

The Installation Manager installation modes support the installation, update, rolling back to a previous level, or modifying the installation by adding or removing optional features. For SOAP Gateway, you can use the following installation modes:

- z/OS For z/OS, you can use the following mode:

  command mode
  You use the command line (imcl) command to manage installations. You can run installation scripts that include commands and options for specifying the details of your installation.

- Linux Windows For distributed platforms, you can use the following modes:

  wizard mode
  You run Installation Manager from a graphical user interface.

  silent mode
  You can deploy software to one or multiple systems through a response file (an XML file) that contains the installation setup information and launched it from the command line or a batch file.

Shared resources directory

The *shared resources directory* is the directory where installation related resources are located that can be used by one or more package groups. This directory is used to cache repository files, and hold them in case a future rollback is needed and the initial repositories are not available elsewhere.

Tip: The shared resources directory is set the first time you use an Installation Manager instance to install a package. You can omit this value in subsequent product installation. If the value is specified again in a subsequent installation, it is ignored. You cannot change the directory location until you uninstall all packages and the directory is empty.

IBM Installation Manager for z/OS

IBM Installation Manager for z/OS must be installed on each of the systems or LPARs that you plan to install SOAP Gateway on, unless these LPARs are in the same sysplex. If you plan to have multiple instances of SOAP Gateway on multiple LPARs not on the same sysplex, you need to transfer the Installation Manager installation kit to each LPAR to install the Installation Manager. The IMS Enterprise Suite Base Services FMID provides sample JCL jobs to help you transfer these files.

Driving systems versus target systems for z/OS

The terms *driving system* and *target system* refer to the system where SMP/E processing is conducted, and the system where SOAP Gateway is to be installed.

- *Driving system* is the system where SMP/E executes.
- *Target system* is the system on which the program is configured and run.

After the SMP/E RECEIVE, APPLY, and ACCEPT processing, the product code is put onto the driving system. However, the driving system might not be the system that the product is intended to run on.

When the target system is the same as the driving system

When the target system is the same as the driving system, all installation sample jobs and the installation repositories for both the Installation Manager and SOAP Gateway are available after the SMP/E processing. No additional steps are required to transfer the SMP/E installed product code to the target system.

The following figure shows the steps to install the IBM Installation Manager for z/OS in order to install SOAP Gateway on z/OS on one LPAR when the target system is the same as the SMP/E driving system.

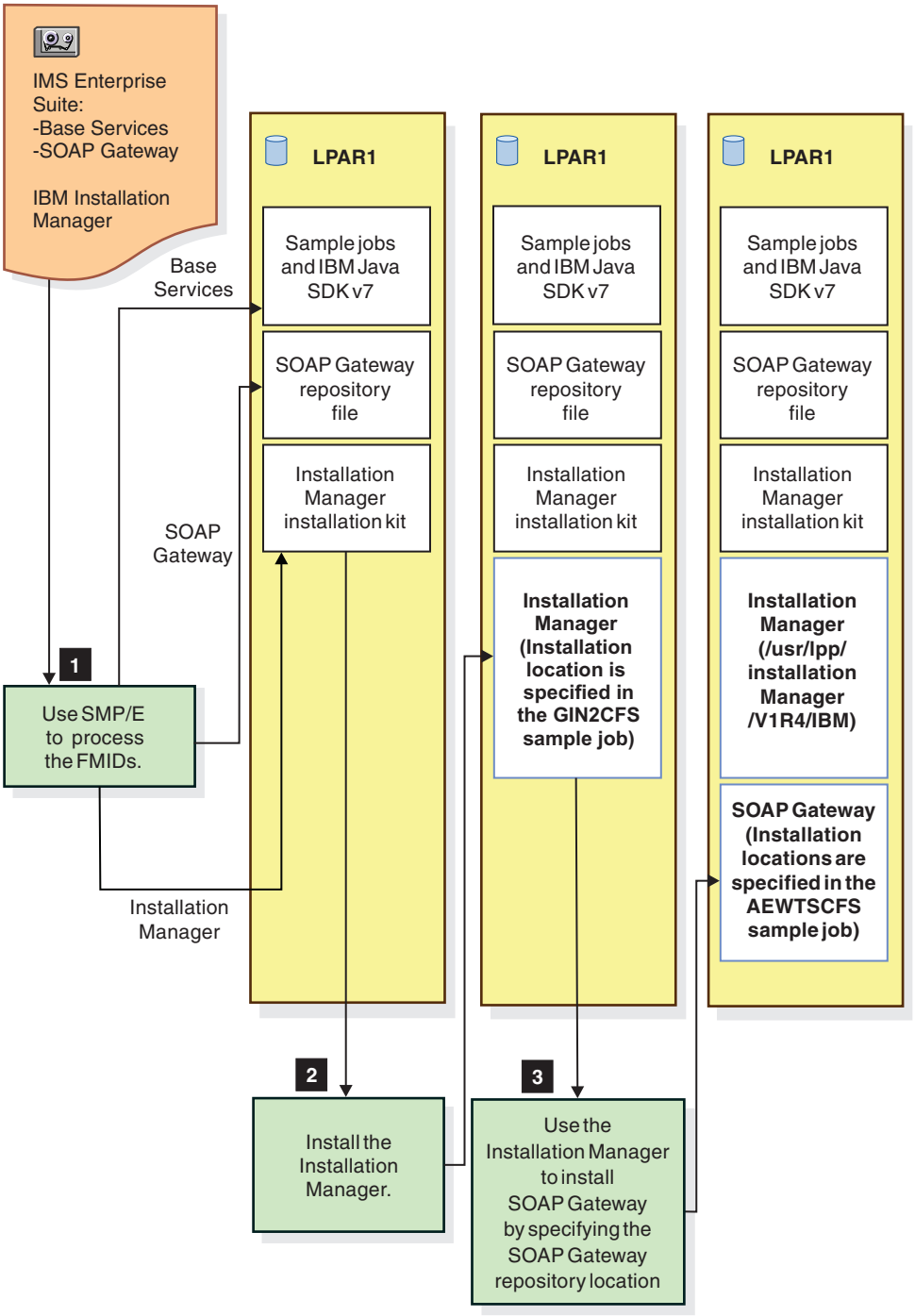


Figure 7. SOAP Gateway installation process

The diagram shows three general steps:

1. First use SMP/E to process the IMS Enterprise Suite Base Services and SOAP Gateway, and IBM Installation Manager for z/OS.
   - SMP/E processing of the IMS Enterprise Suite Base Services FMID puts the sample installation JCL jobs on the file system. Sample installation JCL jobs are stored in SAEWBASE after SMP/E APPLY, and AAEWBASE after SMP/E ACCEPT.

Important: The IBM Java SDK in the IMS Enterprise Suite Base Services component, which is referred to in the program directory, has been deprecated (APAR PI33917). See the PSP bucket for the latest supported Java version and download instructions. The instructions also specify how to order Java for z/OS through Shopz at no charge.

- SMP/E processing of the SOAP Gateway FMID puts the SOAP Gateway repository that IBM Installation Manager uses to install SOAP Gateway on the file system.
- SMP/E processing of the IBM Installation Manager for z/OS FMID and then RECEIVE, APPLY, ACCEPT PTF UK79476 to upgrade to Installation Manager V 1.5.3. This process puts the installation kit onto the file system.

2. Install (or "create" as the process is described in the IBM Installation Manager documentation) the Installation Manager from the installation kit. The installation kit includes a directory structure that consists of executable files, application data, a shared resources cache, and a set of GINxxxxx sample installation jobs.
3. After the Installation Manager is installed, you can use it to install SOAP Gateway by specifying the location of the SOAP Gateway repository. Sample jobs for installing SOAP Gateway are provided in the IMS Enterprise Suite Base Services.

When the target system is different from the driving system

If the target system where SOAP Gateway is to be installed is different from the driving system, the IBM Installation Manager must be installed on the target system first.

The following figure describes how the components are installed through the SMP/E process on to the driving system, LPAR1.

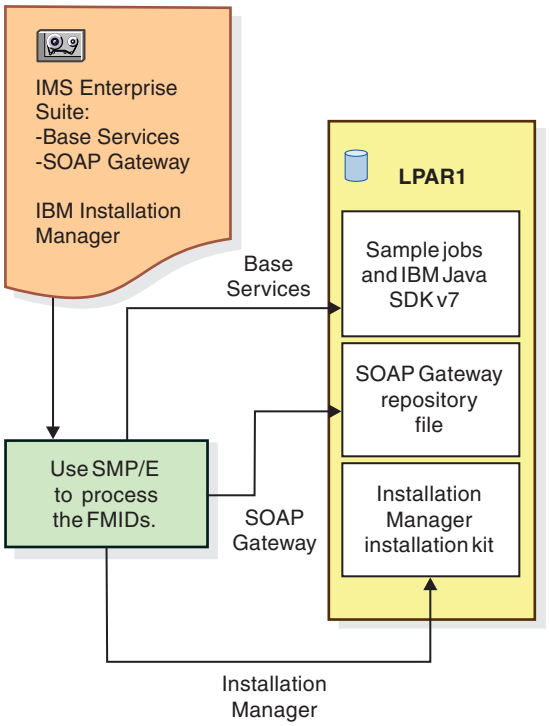


Figure 8. SOAP Gateway SMP/E installation process

If SOAP Gateway is to be installed on a different system, you must transfer the IBM Installation Manager installation kit to the target system in order to install the Installation Manager on the target system.

The following figure shows the high-level steps to install SOAP Gateway on z/OS when the target system is different from the SMP/E driving system.

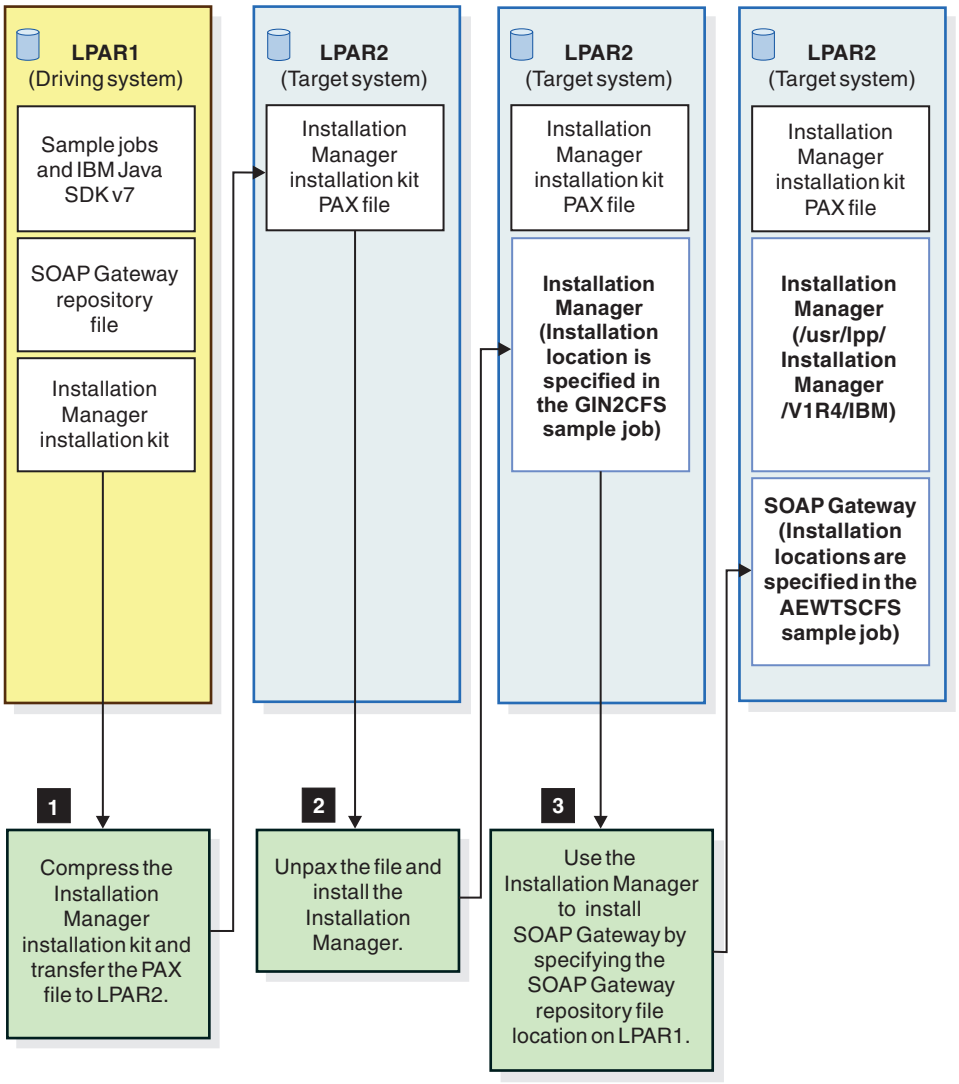


Figure 9. SOAP Gateway installation process when the target system is different from the SMP/E driving system

As shown in the figure, after the SMP/E processing, you must take the following steps:

1. Transfer the Installation Manager installation kit to the target system. Because the kit contains a directory structure with various files, you compress the files into a PAX file. Then transfer the PAX file and the provided JCL sample jobs (GIN2xxxx) to the target system.
2. Extract the PAX file on the target system. Then install the Installation Manager by editing and submitting the provided sample GIN2xxxx jobs.
3. Install SOAP Gateway by using the Installation Manager.

Sample jobs for installing SOAP Gateway are provided in the IMS Enterprise Suite Bases Services FMID to help you with these tasks.

Configuration and setup planning

Before you install SOAP Gateway, you must determine whether to install SOAP Gateway under one directory (or mount point for z/OS), or multiple directories. Sufficient disk space must also be available.

The directory decision depends on several factors. One major factor is the security requirements of your environment, and whether write access to the SOAP Gateway installation directory is a concern. When the **imsserver** component is installed on a separate location or mount point, the **imsserver** directory can be set in read-only mode. By making the **imsserver** directory read only, accidental removal or alteration of code is prevented to preserve the code integrity.

Another major factor is size. While the **imsserver** component does not change in size regardless of how long the SOAP Gateway server is up and running, or how many web services it handles, the **imsbase** and **imssoap** components do. The size of the **imsbase** component grows with the log files. Sufficient disk space must be allocated for log files, and the size depends heavily on your trace level setting requirements and whether you plan to enable transaction logging. The size of the **imssoap** component grows when more web services are added. In general each web service takes up 50 to 100 KB of disk space. Having each of the three SOAP Gateway components installed on a different directory or mount point provides the flexibility to add more disk space without the need to reinstall SOAP Gateway.

If later on you run out of disk space and more must be added, you must create a new directory or mount point in a different path.

z/OS On z/OS, one option is to mount the new directory under the old mount point, and then use a command such as the COPYTREE UNIX command on z/OS to manually copy everything from the old directory. Additional manual configuration is required. Therefore, careful planning of disk space based on anticipated usage before installation is crucial.

If you have multiple SOAP Gateway instances, you might consider sharing one JVM instance.

Related tasks:

"Installing multiple SOAP Gateway server instances that share one JVM" on page 110
Multiple SOAP Gateway server instances can share a single instance of the Java Virtual Machine (JVM). This option reduces the amount of storage required for each additional server instance and improves the serviceability because a SDK service update needs to be installed only once to apply to all SOAP Gateway servers sharing that SDK.

Related reference:

"System requirements" on page 41
IBM IMS Enterprise Suite V3.1 SOAP Gateway supports z/OS, Linux on System z, and Microsoft Windows.

Setting up UID(0) access for SMP/E installation

Before the SMP/E installation (APPLY) of the IMS Enterprise Suite SOAP Gateway, the User ID that is used for the SMP/E process must be set up to have the proper access to several classes in the RACF facility.

The User ID that is used for the SMP/E process must have UID(0) and READ access or higher to the following RACF facility classes:

- BPX.SUPERUSER: Allows users to switch to the superuser authority.
- BPX.FILEATTR.APF: Controls which users are allowed to set the APF-authorized attribute in a z/OS UNIX file. SOAP Gateway installation and configuration has a dependency on Java being APF-authorized.
- BPX.FILEATTR.PROGCTL: Controls which users are allowed to set the program control attribute.
- BPX.FILEATTR.SHARELIB: Indicates that extra privileges are required when setting the shared library extended attribute through the `chattr()` callable service.

1. Use the id command to print your user identity. This command displays your User ID (UID), Group ID (GID) and the groups. The following example shows that UID(0) is needed for installation.

```
#id
# uid=0(userid) gid=0(SYS1) groups=1(IMWEB)
```

2. From ISPF option 6 (command) you can permit and check a user's access to the required facility classes. In the following steps:
   - Use the PERMIT command to set the class permission.
   - The SETROPTS RACLIST(FACILITY) REFRESH command must follow the PERMIT command for the READ access permission to take effect.
   - The RL FACILITY some.facility.class AU command displays the permissions for a specific facility class.

   a. Define BPX.SUPERUSER READ access:

```
- PERMIT BPX.SUPERUSER CLASS(FACILITY) ID(userid) ACCESS(READ)
- SETROPTS RACLIST(FACILITY) REFRESH
- RL FACILITY BPX.SUPERUSER  AU
```

   b. Define HFS programs as program controlled (BPX.FILEATTR.PROGCTL) with READ access (the **p** extended attribute in z/OS UNIX System Services).

```
- PERMIT BPX.FILEATTR.PROGCTL CLASS(FACILITY) ID(userid)
     ACCESS(READ)
- SETROPTS RACLIST(FACILITY) REFRESH
- RL FACILITY BPX.FILEATTR.PROGCTL AU
```

   c. Define HFS programs as APF-authorized programs (BPX.FILEATTR.APF) with READ access (the **a** extended attribute in z/OS UNIX System Services).

```
- PERMIT BPX.FILEATTR.APF CLASS(FACILITY) ID(userid)
     ACCESS(READ)
- SETROPTS RACLIST(FACILITY) REFRESH
- RL FACILITY BPX.FILEATTR.APF AU
```

   d. Define HFS programs as shared library programs (BPX.FILEATTR.SHARELIB) with READ access (the **l** extended attribute in

z/OS UNIX System Services).

```
- PERMIT BPX.FILEATTR.SHARELIB CLASS(FACILITY) ID(userid)
     ACCESS(READ)
- SETROPTS RACLIST(FACILITY) REFRESH
- RL FACILITY BPX.FILEATTR.SHARELIB AU
```

Related information:

The extattr command for setting extended attributes for files
See z/OS UNIX System Services Command Reference in the z/OS V1R13.0 information center.

Upgrading to Version 3.1 SOAP Gateway

Migration of web services, web service related schemas, and server properties set through the SOAP Gateway management utility from Version 2.2 to Version 3.1 are supported.

Table 13. SOAP Gateway upgrade paths to Version 3.1

What you currently have installed	Installation path and migration steps
None	Install IMS Enterprise Suite Version 3.1 SOAP Gateway directly.
IMS Enterprise Suite Version 2.2 SOAP Gateway	Install IMS Enterprise Suite Version 3.1 SOAP Gateway, and use the SOAP Gateway management utility to migrate the web services and server properties.
IMS Enterprise Suite Version 2.1 SOAP Gateway	Install IMS Enterprise Suite Version 3.1 SOAP Gateway, and use the SOAP Gateway management utility to migrate the web services and server properties.
IMS Enterprise Suite Version 1.1 SOAP Gateway	1. Install IMS Enterprise Suite Version 2.1 SOAP Gateway and apply the latest updates. 2. Use the SOAP Gateway management utility in V2.1 to migrate the web services and server properties from V1.1 to V2.1. 3. Install IMS Enterprise Suite Version 3.1 SOAP Gateway. 4. Use the SOAP Gateway management utility in V3.1 to migrate the web services and server properties from V2.2 to V3.1.

Skill requirements by role and responsibility

Installation of SOAP Gateway requires server and system administration knowledge and experiences. To install SOAP Gateway on the z/OS platform, knowledge of z/OS UNIX System Services, the hierarchical file system (HFS) or zSeries File System (zFS), and the security product you use are also required.

The following table describes the different computing roles that members of your organization might have when working with the SOAP Gateway server.

Table 14. Skill requirements for installing SOAP Gateway by role and responsibility.

Role	Responsibility and skills	SOAP Gateway tasks and skills
System administrator	Responsible for managing systems and software, and for installing operating system upgrades and middleware products. Additionally, the system administrator typically monitors and maintains the configured system, adds and delete users, and manages networks and connectivity.	Work with the security administrator, and system programmer if installing on z/OS, to: • Install, configure, maintain and troubleshoot issues with SOAP Gateway. • Deploy web services.
System programmer	Responsible for the availability and acceptable performance of z/OS systems. The system programmer installs, configures, migrates, and customizes software on z/OS, automates operations, maintains system performance, plans for future releases, engineers system to accommodate new functions, and researches and applies preventive and corrective service.	Work with the system administrator and security administrator to: • Install, configure, maintain and troubleshoot issues with SOAP Gateway. • Deploy web services. The following skills or knowledge are required: • z/OS UNIX System Services and the hierarchical file system (HFS) • SMP/E and JCL • IMS
Security administrator	The system administrator and system programmer might need to work with the security administrator to obtain the necessary security information for user authentication and authorization to access the server or services hosted on the server.	Provide security related information and assistance for implementing web service security. The following skills are required: • Resource Access Control Facility (RACF), or the System authorization facility (SAF) product you use, to authenticate SOAP Gateway clients and servers, and authorize access to resources • Secure Sockets Layer (SSL), to enable security (recommended)
Application developer	Creates client applications that access services hosted on the SOAP Gateway server, or web services that IMS applications invoke. The application developers are generally familiar with the SOAP protocol, XML and web services, and the business logic.	Develop client applications to access web services hosted by the SOAP Gateway server. The following skills are required: • Java and related technologies required by your applications or web services • SOAP and XML

Table 14. Skill requirements for installing SOAP Gateway by role and responsibility. (continued)

Role	Responsibility and skills	SOAP Gateway tasks and skills
Enterprise architect and Solution architect	The enterprise architect provides overall leadership for all architectural and technological matters with respect to the enterprise's IT environment. The solution architect designs and coordinates a new solution, application or component with end-to-end responsibility including both hardware and software elements.	Identify solutions needed and the setup and configuration of the environment in which SOAP Gateway is to be installed or operates.

To gain familiarity with Multiple Virtual Storage (MVS), Time Sharing Option/Extensions (TSO), and Interactive System Productivity Facility (ISPF), or the z/OS UNIX System Services, see the z/OS topics in the z/OS basic skills information center

Related information:

z/OS basic skills information center
The information center provides the essential information on z/OS, MVS, and z/OS UNIX System Services.

Learning z/OS and UNIX

To install and maintain SOAP Gateway on z/OS, an understanding of both z/OS and UNIX concepts and skills is required.

The z/OS operating system provides the interfaces and system services of the original MVS operating system. z/OS also adds a UNIX environment through the z/OS UNIX System Services system component. To install, configure, and maintain SOAP Gateway, you use the z/OS skills to allocate and mount a file system. You use the UNIX commands to create directories, navigate through the directory structure, and transfer files through FTP. You modify JCL jobs or create JCL started tasks that execute UNIX shell scripts or commands.

The sample jobs provided in the IMS Enterprise Suite Base Services can be used as learning tools. They demonstrate the following tasks:

- Allocating a partitioned data set (PDS) to store data
- Allocating a hierarchical file system (HFS) or z/OS Distributed File Service zSeries file system (zFS)
- Creating directories for a file system (the UNIX MKDIR command)
- Mounting a file system
- Opening an FTP session, logging in to an FTP server, and changing to the correct directories on the local system and the remote system to get or put a file.

You can store a sequence of shell commands in a text file that can be executed. For system programmers who are familiar with MVS, Time Sharing Option/Extensions (TSO), and Interactive System Productivity Facility (ISPF), the ISHELL command starts the ISPF panel interface to z/OS UNIX System Services. If you are familiar with UNIX, the OMVS command is used to start the z/OS UNIX shell.

The provided AEWIOGBP sample job demonstrates how to run the SOAP Gateway management utility iogmgmt command by using the BPXBATCH utility.

Related information:

A brief comparison of z/OS and UNIX
The z/OS basic skill information center provides the essential information about z/OS, MVS, and z/OS UNIX System Services.

Using z/OS UNIX from batch, TSO/E, and ISPF
The z/OS V1R12 information center provides the essential information about z/OS UNIX System Services.

Installing SOAP Gateway on z/OS

Use the provided installation roadmap to guide your installation. Installation must be completed by a system programmer who is familiar with installation tasks on a z/OS platform.

Prerequisites:

1. Ensure that you meet the "System requirements" on page 41 and the "Software requirements" on page 43.
2. Review the "Planning for installation" on page 45 information to understand your configuration and setup requirements, server upgrade path, skill requirements, and the difference between the driving system and the target system. Proper user access setup for RACF is required for SMP/E processing and SOAP Gateway installation and configuration.
3. Order a CBPDO for the IMS Enterprise Suite product through Shopz. The package also includes IBM Installation Manager for z/OS FMID HGIN140 and its *Program Directory* (GI11-9852).
4. Print the *Program Directory for IMS Enterprise Suite V3.01* (GI10-8964), and the *IBM Installation Manager for z/OS V1.4 Program Directory* (GI11-9852).

   The *Program Directory* contains detailed block size requirements and operational requirements, such as the required version of IBM SMP/E for z/OS for installation. The *Program Directory* also provides information about the function modification identifiers (FMIDs), the installation instructions, and any required service process.

   Important: The IBM Java SDK in the IMS Enterprise Suite Base Services component, which is referred to in the program directory, has been deprecated (APAR PI33917). See the PSP bucket for the latest supported Java version and download instructions. The instructions also specify how to order Java for z/OS through Shopz at no charge.
5. Check for any SOAP Gateway APARs that need to be installed.

Installation roadmap for SOAP Gateway on z/OS

Use the following installation roadmap to guide your installation. For more information on applying services, see "Applying maintenance services on z/OS" on page 115.

Table 15. Roadmap for installing and configuring SOAP Gateway on the z/OS platform

Step	Description of task	Role
1	Determine your installation scenario. Follow the installation instructions for your scenario: • "Scenario 1. IBM Installation Manager for z/OS is not installed on the target system" on page 62. 1. Install IBM Installation Manager for z/OS on the target system. 2. Install SOAP Gateway on z/OS by using IBM Installation Manager. • "Scenario 2. IBM Installation Manager for z/OS is installed on the target system" on page 73. 1. If the target system for SOAP Gateway installation is different from the SMP/E driving system, transfer the sample installation jobs to the target system. 2. Install SOAP Gateway on z/OS by using IBM Installation Manager.	System programmer and SOAP Gateway server administrator
2	Modify the provided procedures for runtime configuration, starting and stopping SOAP Gateway, and running the SOAP Gateway management utility: 1. Configure SOAP Gateway for z/OS. 2. Define the OMVS segment and user ID to set up RACF for the SOAP Gateway started task	SOAP Gateway server administrator
3	Configure the SOAP Gateway run time: • Configure the Java SDK location. • (Optional) Configure the SOAP Gateway server port numbers. • (Optional) Configure the SOAP Gateway log file location. • (Optional) Specify how you want to log SOAP Gateway messages by setting the trace level.	SOAP Gateway server administrator
4	(Optional) Configure the SOAP Gateway to run on a zAAP.	SOAP Gateway server administrator
5	Configure IMS Connect for SOAP Gateway. IMS Connect manages the translation of message headers on input and output messages and provides a point of control to modify, route, and check security for messages from and to SOAP Gateway.	System programmer
6	Verify the installation by using the SOAP Gateway Installation Verification Program (IVP).	SOAP Gateway server administrator and system programmer

Table 15. Roadmap for installing and configuring SOAP Gateway on the z/OS platform (continued)

Step	Description of task	Role
7	• If you are upgrading from IMS Enterprise Suite Version 2.2 SOAP Gateway, migrate your web services. See "Migrating from IMS Enterprise Suite Version 2.2 SOAP Gateway" on page 106. • If you are upgrading from Version 2.1 SOAP Gateway, migrate your web services, see "Migrating from IMS Enterprise Suite Version 2.1 SOAP Gateway" on page 104. • If you are upgrading from Version 1.1 SOAP Gateway, migrate to V2.1 first, and then migrate from V2.1 to V3.1.	SOAP Gateway server administrator and system programmer
8	(Optional) Clone the SOAP Gateway server.	SOAP Gateway server administrator and system programmer
9	(Optional) Install multiple SOAP Gateway server instances that share one JVM.	SOAP Gateway server administrator and system programmer

Scenario 1. IBM Installation Manager for z/OS is not installed on the target system

The Installation Manager must be installed on the target system or LPAR where SOAP Gateway is to be installed, or on a different LPAR that is in the same sysplex.

Prerequisites:

1. Review the prerequisites in "Installing SOAP Gateway on z/OS" on page 60.
2. If you do not have IBM Installation Manager for z/OS:
   a. Use SMP/E to RECEIVE, APPLY, and ACCEPT the IBM Installation Manager (HGIN140). See the *IBM Installation Manager for z/OS V1.4 Program Directory* (GI11-9852) for the instructions.
   b. Apply at least PTF UK79476 to upgrade the Installation Manager installation kit to V 1.5.3 or later.

      Important: SOAP Gateway installation requires IBM Installation Manager V1.5.3 or later.

      The SMP/E processing of this FMID puts the IBM Installation Manager installation kit on the driving system. The Installation Manager is not yet installed.
3. Use SMP/E to RECEIVE, APPLY, and ACCEPT the IMS™ Enterprise Suite Base Services (HAHF310) and SOAP Gateway (JAHF311). See the *Program Directory for IMS Enterprise Suite V3.01* (GI10-8964) in the IBM Publication Center for instructions.

   Step result: A set of sample installation JCL jobs, IBM Java, and the SOAP Gateway repository are put on the driving system.
   - AAEWBASE: Contains sample installation jobs for SMP/E processing and three post-SMP/E installation jobs for installing SOAP Gateway by using the IBM Installation Manager. For a list of the sample jobs, see "Sample jobs for installation and configuration" on page 84.

- AAEWSAMP: Contains additional sample jobs for installing, configuring, and starting SOAP Gateway. Use these jobs to configure SOAP Gateway runtime properties, to start and stop the SOAP Gateway server,
- AAEWJV31, AAEWJV64: Contain the 31-bit and 64-bit IBM Java SDK.

  Important: The IBM Java SDK in the IMS Enterprise Suite Base Services component, which is referred to in the program directory, has been deprecated (APAR PI33917). See the PSP bucket for the latest supported Java version and download instructions. The instructions also specify how to order Java for z/OS through Shopz at no charge.
- SIOGSHFS: Contains the SOAP Gateway repository (for use with installation by using the Installation Manager). The SOAP Gateway repository file is located in the `/usr/lpp/InstallationManagerRepository/JAHF311/` directory.

4. If the target system where SOAP Gateway is to be installed is different from the SMP/E driving system, set up both the driving system and the target system to use FTP because you need to transfer files and sample jobs between the two systems. Both an FTP client and an FTP server are included as part of the base z/OS Communications Server functions.

Tips:

- The IMS Enterprise Suite Base Services FMID (HAHF310) includes a set of sample JCL jobs. Sample jobs for installing SOAP Gateway start with the prefix AEW, followed by the letter D or T. D indicates that the job is to be run on the driving system. T indicates that the job is to be run on the target system.
- A "Worksheet for installation scenario 1" on page 69 is provided to help you keep track of parameter values during the installation.

Next step: You are ready to install the Installation Manager from the installation kit.

Installing IBM Installation Manager for z/OS on the target system

IBM Installation Manager for z/OS is provided with IMS Enterprise Suite for z/OS. The Installation Manager must be installed in the same sysplex where SOAP Gateway will be installed.

Prerequisite: The IBM Installation Manager (HGIN140), with PTF UK79476 applied to upgrade to V 1.5.3 or later.

If the target system where SOAP Gateway is to be installed is the same as the SMP/E driving system, go to step 8.

If the target system is in a different sysplex, start with step 1. The IMS Enterprise Suite Base Services FMID includes extra sample JCL jobs in the AAEWSAMP data set to help you transfer the IBM Installation Manager installation kit and other sample JCL jobs from the driving system to the target system. These steps are described in steps 1 to 7.

Recommendation:

- IBM Installation Manager supports installation (or the creation of an Installation Manager) in admin mode, group mode, or user mode. When an Installation Manager is created in admin mode (installed by a superuser of UID=0), only the superuser can start the Installation Manager to install products. That is, the user who installs SOAP Gateway must be the same user that installs or creates the Installation Manager. The best approach is to install the Installation Manager in

group mode to allow any users connected to the UNIX System Services group that owns the Installation Manager to install SOAP Gateway.

- Choose or create a user ID to install and run Installation Manager. The user ID must have the following attributes:
  - Read/write access to the directory where the IBM Installation Manager is installed
  - Read access to theses FACILITY profiles:
    - BPX.FILEATTR.APF
    - BPX.FILEATTR.PROGCTL
    - BPX.FILEATTR.SHARELIB
    - BPX.SUPERUSER.

    For more information about setting up the user ID access, see "Setting up UID(0) access for SMP/E installation" on page 55.
  - Read access to these UNIXPRIV profiles:
    - SUPERUSER.FILESYS.CHOWN
    - SUPERUSER.FILESYS.CHANGEPERMS

1. **Driving** On the driving system, allocate a temporary 400-cylinder file system for storing the Installation Manager installation kit PAX file.

Sample job to edit	AEWDALPX (or AEWDZALP for zFS)
Description	This sample JCL job lets you allocate and mount a temporary HFS to store the PAX file for IBM Installation Manager.
Step result	A file system is allocated and mounted.

2. **Driving** On the driving system, compress the IBM Installation Manager installation kit into a PAX file.

Sample job to edit	AEWDIMPX
Description	This JCL lets you create a PAX file that contains the IBM Installation Manager installation kit.
Step result	Files in the installation kit directory structure are compacted into a PAX file and stored in the HFS mounted under the **-PAXPathPrefix-** mount point.

3. **Target** On the target system, prepare a 1-cylinder partitioned data set (PDS) to store the sample JCLs jobs for transferring the PAX file and additional sample installation jobs.

Sample job to edit	No job is provided.

Description	You can copy, edit, and submit the following sample job to allocate the PDS:
	`//ALLOCATE JOB <JOB PARAMETERS>`
	`//*`
	`//ALLOCATE EXEC PGM=IDCAMS,DYNAMNBR=200`
	`//SYSPRINT DD SYSOUT=*`
	`//SYSIN DD *`
	`ALLOCATE -`
	`DSNAME('hlqual.TARGET.JCL') -`
	`FILE(AGINJCL) -`
	`RECFM(F,B) -`
	`LRECL(80) -`
	`BLKSIZE(3200) -`
	`DSORG(PO) -`
	`DSNTYPE(PDS) -`
	`NEW CATALOG -`
	`SPACE(1,2) CYL -`
	`DIR(30) -`
	`UNIT(SYSALLDA)`
	`//*`

Step result	A PDS is created to store a set of sample jobs to transfer from the driving system.

4. **Driving** From the driving system, transfer a set of AEWTxxxx sample JCL jobs to the PDS created in step 3 on the target system.

Sample job to edit	AEWDFTP
Description	This sample JCL job lets you send sample JCL jobs for preparing the file systems and transferring the IBM Installation Manager installation kit PAX file.
Step result	The PDS allocated in step 3 stores the following sample JCL jobs: • AEWTALLO • AEWTFTP • AEWTUMDL • AEWTUNPX • AEWTZALC • AEWTZUMD These jobs are used in the next step (step 5) for transferring the Installation Manager PAX file and additional sample JCL jobs from the driving system to the target system. Sample jobs for cleaning up the temporary file systems are also included.

5. **Target** Prepare the target system to store the PAX file for IBM Installation Manager installation kit and the files after the PAX file is extracted. Allocate a PDS to store additional sample jobs to install the Installation Manager and SOAP Gateway.

Sample job to edit	AEWTALLO (or AEWTZALC for zFS)
Description	This sample JCL job helps you prepare for the file systems to store the IBM Installation Manager installation kit PAX file and the IBM Installation Manager installation after the PAX file is extracted. This job also prepares a PDS to store additional jobs to install SOAP Gateway.
Step result	Two file systems and a PDS are allocated on the target system.

6. **Target** On the target system, transfer the PAX file for Installation Manager and additional sample jobs from the driving system.

Sample job to edit	AEWTFTP
Description	This sample JCL job is run on the target system to get the IBM Installation Manager installation kit PAX file and the sample jobs for installation and configuration from the driving system through FTP.
Step result	The following files are now on the target system: • GIN*xxxxx* sample jobs provided by Installation Manager for installing of the product are in *hlqual1*.HGIN140.AGINJCL. • AEW*xxxxx* sample jobs and the configuration file for installing, uninstalling, and configuring SOAP Gateway and for cleaning up the file systems are in *hlqual3*.TARGET.JCL. • The AEWIOGPR JCL procedure for starting the SOAP Gateway server as a started task is in *hlqual5*.PROCLIB. • The PAX file for the Installation Manager installation kit is in the **`/local/directory`** directory.

7. **Target** On the target system, extract the PAX file for the Installation Manager installation kit by editing and submitting job AEWTUNPX.

Sample job to edit	AEWTUNPX
Description	This sample JCL job extracts the IBM Installation Manager installation kit PAX file.
Step result	The installation kit PAX file is extracted and ready for installation.

8. **Target** On the target system, install the Installation Manager.

Sample jobs to edit	GIN2ADMN, GIN2CFS, GIN2INST, GIN3CMD
Description	Follow the installation instructions in the Activating IBM Installation Manager for z/OS V1.4 section in the *IBM Installation Manager Program Directory* and the instructions in the JCL jobs to install IBM Installation Manager for z/OS
Step result	The Installation Manager is installed (or created) on the target system.

9. **Target** (Optional) Clean up the temporary file system on the target system.

Sample job to edit	AEWTUMDL (or AEWTZUMD for zFS)
Description	This sample JCL job helps you clean up the temporary file system allocated in sample job AEWTALLO (or AEWTZALC) in step 5.
Step result	The temporary file system on the target system for storing the Installation Manager PAX file is cleaned up.

10. **Driving** (Optional) If you do not plan to install Installation Manager on additional systems, you can clean up the temporary file system on the driving system.

Sample job to edit	AEWDUMDL (or AEWDZUMD for zFS)
Description	This sample JCL job helps you clean up the temporary file system created in AEWDALPX (or AEWDZALC) in step 1).
Step result	The temporary file system on the driving system for storing the Installation Manager PAX file is cleaned up.

The Installation Manager is now installed on the target system.

You are ready to install SOAP Gateway.

Installing SOAP Gateway on z/OS by using IBM Installation Manager

Edit and submit the provided sample jobs to install SOAP Gateway.

Prerequisites:

- Determine whether you want to install SOAP Gateway on one mount point or multiple mount points. Determine the disk space that you need based on the number of web services you plan to host, your trace level setting for logging, and whether transaction logging must be enabled. For more information, see "SOAP Gateway architecture" on page 27 and "Configuration and setup planning" on page 55.
- Determine the disk space requirement for the IMSBASE and IMSSOAP components. These components are where the server logs and web services artifacts are stored, and trace level setting, transaction log setting, and number of web services would impact the space requirement. For more information, see System requirements.
- IMS Enterprise Suite Base Services (HAHF310) and SOAP Gateway (JAHF311) must be already processed by SMP/E.
- IBM Installation Manager V 1.5.3 or later is installed on the target system where SOAP Gateway is installed.
- The Installation Manager must be able to access the file system where the SOAP Gateway repository file is located.
- The following SOAP Gateway installation sample files are on the target system:
  - AEWTSCFS
  - AEWTCKER
  - AEWTSINS
  - AEWTSUNI

Tip: Detailed instructions for each sample job are provided in the jobs. You might want to print a copy of each job before you start.

All steps are to be completed on target system where SOAP Gateway is being installed.

1. Target Allocate and mount a file system on the target system for SOAP Gateway installation.

Sample job to edit	AEWTSCFS

Description	This job, by default, install SOAP Gateway into one file system, either HFS or zFS, on one mount point. To install the three SOAP Gateway components on different mount points, make two copies of the zCreateFileSystem.sh command by following the instructions that are provided in the job. The value of **target.path** (or a **target.path** value for each of the SOAP Gateway component) that you specify here is later used in the AEWTSINS job to install SOAP Gateway.
Step result	The target system has a file system or three file systems ready for the SOAP Gateway installation.
Next step	If SOAP Gateway is to be installed on a different target system from where the AAEWBASE and AAEWSAMP data sets are located, continue with step 2. Step 2 is for preparing the required sample installation jobs and creating a keyring file to pass the FTP user name and password information for IBM Installation Manager to access the SOAP Gateway repository. If you are installing SOAP Gateway on the same system as the SMP/E driving system: • If the SOAP Gateway repository file is on the same system as IBM Installation Manager, go to step 3 to install SOAP Gateway. • If the SOAP Gateway repository file is not on the same system, go to step 2.

2. **Target** This step is needed only when the SOAP Gateway repository file is on a different system from where the Installation Manager is installed. IBM Installation Manager requires a keyring for secure installation.

Sample job to edit	AEWTCKER
Description	Use this job to create a keyring file to store the user name and password information. Use the imutilsc command in Installation Manager to create the keyring file. For more information about credentials and the imutilsc command, see the IBM Installation Manager information center.
Step result	A keyring file is created.

3. **Target** Install SOAP Gateway by using the Installation Manager imcl command.

Sample job to edit	AEWTSINS
Description	This job installs SOAP Gateway by using the Installation Manager imcl -install command. You must edit and submit the job for each SOAP Gateway installation component (also known as a package) by specifying the package name for each run: • com.ibm.ims.sgw.imsserver.v31 • com.ibm.ims.sgw.imsbase.v31 • com.ibm.ims.sgw.imssoap.v31
Step result	SOAP Gateway is installed.

You must configure SOAP Gateway next before you can start the SOAP Gateway server.

Related tasks:

Installing SOAP Gateway on z/OS (Installation roadmap)
Use the provided installation roadmap to guide your installation. Installation must be completed by a system programmer who is familiar with installation tasks on a z/OS platform.

Related information:

Keyring files and credentials (IBM Installation Manager V1.5 information center)

Worksheet for installation scenario 1

Use this worksheet to record the file system or directory names that you use or create during the installation. Scenario 1 assumes that you do not yet have IBM Installation Manager for z/OS on the target system.

Before you begin, ensure that you know the location where the required FMIDs are processed by using SMP/E. Use the following table to record the directories or file systems where the required files and data sets are located on the SMP/E driving system.

Table 16. Locations for IMS Enterprise Suite and IBM Installation Manager libraries after SMP/E

Component (Library DDName)	Location
The Installation Manager installation kit (SGINKIT)	The high-level qualifier is specified in the Installation Manager SMP/E processing GINSMKD job. Default path, without your high-level qualifier, is `/usr/lpp/InstallationManager/V1R4` Your directory path:
The Installation Manager sample installation jobs (AGINJCL)	Your file system for the AGINJCL data set:
IBM Java SDK **Important:** The IBM Java SDK in the IMS Enterprise Suite Base Services component, which is referred to in the program directory, has been deprecated (APAR PI33917). See the PSP bucket for the latest supported Java version and download instructions. The instructions also specify how to order Java for z/OS through Shopz at no charge.	Your directory path:
IMS Enterprise Suite sample installation jobs (AAEWBASE and AAEWSAMP)	Your file system for the AAEWBASE and AAEWSAMP data sets:
SOAP Gateway repository (SIOGSHFS)	Default path, without your high-level qualifier, is `/usr/lpp/InstallationManagerRepository/JAHF311/` Your directory path:

Installing IBM Installation Manager for z/OS on a target system different from the SMP/E driving system

The sequence of the following tables corresponds to the steps described for Scenario 1. The tables are grouped in two sections, one for installing the IBM Installation Manager for z/OS, and the other for installing SOAP Gateway.

1. **Driving** AEWDALPX (AEWDZALP for zFS)

Parameters	Your value
#hfsdsn: The temporary file system on the driving system to store a PAX file that will be created in the sample job AEWDIMPX sample job (next step) to store the IBM Installation Manager installation kit. A minimum of 400 cylinders is allocated.	
-PAXPathPrefix-: The high-level directory name (mount point) for the Installation Manager installation kit PAX file. For example, `/u/user1`.	

2. **Driving** AEWDIMPX

Parameters	Your value
-IMPathPrefix-: The IBM Installation Manager installation kit location, or the path prefix specified in the sample job GINISMKD during the SMP/E process.	
-PAXPathPrefix-: The same high-level directory name (mount point) that you specified in sample job AEWDALPX.	

3. **Target** PDS creation (no sample job is provided).

Parameters	Your value

A temporary PDS with a block size of 3200 on the target system to store the sample JCLs that will be transferred over from the driving system in sample job AEWDFTP in the next step. The suggested data set name is *#hlqual*.TARGET.JCL:
`...`
`//SYSIN DD *`
`ALLOCATE -`
`DSNAME('#hlqual.TARGET.JCL') -`
`...`
`BLKSIZE(3200) -`
`...`
`//*`	

4. **Driving** AEWDFTP

Parameters	Your value
target.site.com: The IP address or the fully qualified name of the target system.	

Parameters	Your value
#hlqual: The high-level qualifier of the data set where TCP/IP for z/OS is installed.	
username and **password**: The FTP username and password for the target system.	
#hlqual1: The high-level qualifier specified in AEWALLOC used in the SMP/E process to install IMS Enterprise Suite.	
#hlqual2.TARGET.JCL: The PDS created in step 3 to store sample JCL jobs on the target system.	

5. **Target** AEWTALLO (AEWTZALC for zFS)

Parameters	Your value
#hfsdsn1: The new file system for storing Installation Manager components after the PAX file is extracted. A minimum of 600 cylinders is allocated.	
#hfsdsn2: A temporary file system for storing the Installation Manager installation kit PAX file. A minimum of 400 cylinders is allocated.	
#volid: The VOLSER ID of the DASD that is used to store the Installation Manager installation kit.	
#hlqual: The high-level qualifier used to store sample JCL jobs for the installation of the Installation Manager.	
-PathPrefix1-: The high-level directory name (mount point) on `#hfsdsn1` for storing the installation kit directory structure.	
-PathPrefix2-: The high-level directory name (mount point) on `@hfsdsn2` for storing the PAX file.	

6. **Target** AEWTFTP

Parameters	Your value
driving.site.com: The IP address or the fully qualified name of the driving system.	
#hlqual: The high level qualifier of the data set where TCP/IP for z/OS is installed.	
username and **password**: The FTP username and password for the driving system	

Parameters	Your value
#hlqual3.TARGET.JCL: the PDS on the target system for storing the SOAP Gateway installation and runtime configuration sample jobs.	
#hlqual4: The high level qualifier of the AAEWSAMP data set created during the SMP/E APPLY process of SOAP Gateway on the driving system.	
#hlqual5: The high level qualifier for the PROCLIB for storing the SOAP Gateway configuration member and server startup JCL procedure.	

7. **Target** AEWTUNPX

Parameters	Your value
-PathPrefix1-: The same value for **-PathPrefix1-** that was specified in sample job AEWTALLO (step 5) for storing the Installation Manager installation kit directory structure after the PAX file is extracted.	
-PathPrefix2-: The same value for **-PathPrefix2-** that was specified in sample job AEWTALLO (step 5) for storing the PAX file.	

8. **Target** GIN2ADMN, GIN2CFS, GIN2INST, GIN3CMD

   See the instructions included in the jobs.

9. **Target** AEWTUMDL (or AEWTZUMD for zFS)

Parameters	Your value
#hfsdsn2: The temporary file system (**#hfsdsn2**) allocated in sample job AEWTALLO (AEWTZALC) in step 5) for storing the Installation Manager installation kit PAX file.	

10. **Driving** AEWDUMDL (or AEWDZUMD for zFS)

Parameters	Your value
#hfsdsn: The temporary file system **#hfsdsn** specified in AEWDALPX (or AEWDZALP) in step 1.	

Installing SOAP Gateway

1. **Target** AEWTSCFS (in AAEWBASE)

   AEWTSCFS:

Parameters	Your value
xxxxxx.dsn: The name of the data set where SOAP Gateway is to be installed, one for each SOAP Gateway component if the components are to be installed on different mount points.	
hhhhhh: The VOLSER ID to be used for the target file system.	
xxxxxx.path: The path to the mount point on the target system, one for each SOAP Gateway component if the components are to be installed on different mount points.	

If you are installing the IMSSERVER, IMSBASE, and IMSSOAP components on different file systems, record the data set name, VOLSER ID, and the path to the mount point for each of the component. This information is needed later for the sample installation job AEWTSINS and the sample configuration job AEWPOSIN.

2. **Target** AEWTCKER (If the SOAP Gateway repository file is on a different system from the system where IBM Installation Manager for z/OS is installed.)

Parameters	Your value
ipadress_of_driving_system: The IP address of the driving system that stores the SOAP Gateway repository file.	
path_to_soap_repository: Where on the driving system the SOAP Gateway repository file is located.	
your_user_name and **your_password**: The FTP user name and password to access the driving system where the SOAP Gateway repository file is located.	
keyring_filename: The path and the name of the keyring file.	
-PathPrefix-: The path to the root directory where IBM Installation Manager IBM Installation Manager for z/OS is installed.	

3. **Target** AEWTSINS (in AAEWBASE)

   See instructions that are provided in the sample job for information about where the required information is specified in previous jobs.

Scenario 2. IBM Installation Manager for z/OS is installed on the target system

Ensure PTF UK79476 is applied to upgrade to IBM Installation Manager V 1.5.3 or later. Then use SMP/E to RECEIVE, APPLY, and ACCEPT the IMS Enterprise Suite Base Services and SOAP Gateway.

Applying PTF UK79476 upgrades the IBM Installation Manager installation kit. You must:

1. Mount the upgraded installation kit on your system.

2. Change directory to the new level of the installation kit.
3. Reissue the same installation command (groupinstc, installc, or userinstc) that you used to create the Installation Manager.

Prerequisites:

- Review the prerequisites in "Installing SOAP Gateway on z/OS" on page 60.
- Use SMP/E to process IMS Enterprise Suite Base Services (HAHF310) and SOAP Gateway (JAHF311). See the *Program Directory for IMS Enterprise Suite V3.01* (GI10-8964) in the IBM Publication Center for instructions.
  1. Use SMP/E to RECEIVE, APPLY, and ACCEPT the IMS Enterprise Suite Base Services (HAHF310) and SOAP Gateway (JAHF311). Check for any latest APARs to install.

     Step result: A set of sample installation JCL jobs, IBM Java, and the SOAP Gateway repository are put on the driving system.
     - AAEWBASE: Contains sample installation jobs for SMP/E processing and three post-SMP/E installation jobs for installing SOAP Gateway by using the IBM Installation Manager. For a list of the sample jobs, see "Sample jobs for installation and configuration" on page 84.
     - AAEWSAMP: Contains additional sample jobs for installing, configuring, and starting SOAP Gateway. Use these jobs to configure SOAP Gateway runtime properties, to start and stop the SOAP Gateway server,
     - AAEWJV31, AAEWJV64: Contain the 31-bit and 64-bit IBM Java SDK.

       Important: The IBM Java SDK in the IMS Enterprise Suite Base Services component, which is referred to in the program directory, has been deprecated (APAR PI33917). See the PSP bucket for the latest supported Java version and download instructions. The instructions also specify how to order Java for z/OS through Shopz at no charge.
     - SIOGSHFS: Contains the SOAP Gateway repository (for use with installation by using the Installation Manager). The SOAP Gateway repository file is located in the `/usr/lpp/InstallationManagerRepository/JAHF311/` directory.
  2. If the target system where SOAP Gateway is to be installed is different from the SMP/E driving system, set up both the driving system and the target system to use FTP because you need to transfer files and sample jobs between the two systems. Both an FTP client and an FTP server are included as part of the base z/OS Communications Server functions.

Tips:

- The Base Services FMID (HAHF310) includes a set of sample JCL jobs that begin with the AEW prefix. The prefix is followed by the letter D or T. D indicates that the job is to be run on the driving system. T indicates that the job is to be transferred to the target system and run on the target system.
- A "Worksheet for installation scenario 2" on page 78 is provided to help you keep track of parameter values during the installation.

Next step:

- If the SMP/E driving system where the AAEWBASE, AAEWJAVA, and AIOGSHFS data sets are located is the same as the target system where SOAP Gateway is to be installed, follow the steps in "Installing SOAP Gateway on z/OS by using IBM Installation Manager" on page 67.

- If the SMP/E driving system is different from the target system, follow the steps in "Transferring the sample installation jobs to the target system" to prepare for installation.

Transferring the sample installation jobs to the target system

Before you can install SOAP Gateway, you must prepare the target system by transferring the sample installation jobs from the SMP/E driving system.

Prerequisite: Both the driving system and the target system must be set up to use FTP. Both an FTP client and an FTP server are included as part of the base z/OS Communications Server functions.

To transfer the sample installation jobs and customize the FTP job:

1. **Target** Prepare the target system to store the sample JCL jobs for installing SOAP Gateway. Allocate a temporary, 1-cylinder partitioned data set (PDS) to store the sample JCL jobs for installations.

Sample job to edit	No provided sample job
Steps	You can copy, edit, and submit the following sample job to allocate the PDS. You must customize with your job card, edit the DSNAME, and then submit the job.

```
//ALLOCATE JOB <JOB PARAMETERS>;
//*
//ALLOCATE EXEC PGM=IDCAMS,DYNAMNBR=200
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
   ALLOCATE -
     DSNAME('IMSES31.TARGET.JCL') -
     FILE(AGINJCL)                -
     RECFM(F,B)                   -
     LRECL(80)                    -
     BLKSIZE(3200)                -
     DSORG(PO)                    -
     DSNTYPE(PDS)                 -
     NEW CATALOG                  -
     SPACE(1,2) CYL               -
     DIR(30)                      -
     UNIT(SYSALLDA)
//*
```

Step result	A PDS is created to store a set of sample jobs to transfer from the driving system.

2. **Driving** From the driving system, transfer the AEWTFTP sample job to the target system by editing and submitting the sample job AEWDFTP.

Sample job to edit	AEWDFTP
Description	This JCL job sends sample the jobs for preparing the file systems and transferring the IBM Installation Manager installation kit PAX file to the target system.
Step result	The AEWTFTP sample job is stored in the PDS allocated in step 1.

3. **Target** On the target system, edit and submit the AEWTFTP job to transfer a set of sample jobs and configuration members for SOAP Gateway installation and runtime configuration from the driving system.

Sample job to edit	AEWTFTP
Description	
Step result	The following files are now on the target system: • AEW*xxxxx* sample jobs and a configuration file for installing, uninstalling, and configuring SOAP Gateway and for cleaning up the file systems are in *hlqual3*.TARGET.JCL. • The AEWIOGCF configuration member is in *hlqual3*.TARGET.JCL, and AEWIOGPR JCL procedure for starting the SOAP Gateway server as a started task is in *hlqual5*.PROCLIB.

You are ready to install SOAP Gateway.

Installing SOAP Gateway on z/OS by using IBM Installation Manager

Edit and submit the provided sample jobs to install SOAP Gateway.

Prerequisites:

- Determine whether you want to install SOAP Gateway on one mount point or multiple mount points. Determine the disk space that you need based on the number of web services you plan to host, your trace level setting for logging, and whether transaction logging must be enabled. For more information, see "SOAP Gateway architecture" on page 27 and "Configuration and setup planning" on page 55.
- Determine the disk space requirement for the IMSBASE and IMSSOAP components. These components are where the server logs and web services artifacts are stored, and trace level setting, transaction log setting, and number of web services would impact the space requirement. For more information, see System requirements.
- IMS Enterprise Suite Base Services (HAHF310) and SOAP Gateway (JAHF311) must be already processed by SMP/E.
- IBM Installation Manager V 1.5.3 or later is installed on the target system where SOAP Gateway is installed.
- The Installation Manager must be able to access the file system where the SOAP Gateway repository file is located.
- The following SOAP Gateway installation sample files are on the target system:
  - AEWTSCFS
  - AEWTCKER
  - AEWTSINS
  - AEWTSUNI

Tip: Detailed instructions for each sample job are provided in the jobs. You might want to print a copy of each job before you start.

All steps are to be completed on target system where SOAP Gateway is being installed.

1. **Target** Allocate and mount a file system on the target system for SOAP Gateway installation.

Sample job to edit	AEWTSCFS

Description	This job, by default, install SOAP Gateway into one file system, either HFS or zFS, on one mount point. To install the three SOAP Gateway components on different mount points, make two copies of the zCreateFileSystem.sh command by following the instructions that are provided in the job. The value of **target.path** (or a **target.path** value for each of the SOAP Gateway component) that you specify here is later used in the AEWTSINS job to install SOAP Gateway.
Step result	The target system has a file system or three file systems ready for the SOAP Gateway installation.
Next step	If SOAP Gateway is to be installed on a different target system from where the AAEWBASE and AAEWSAMP data sets are located, continue with step 2. Step 2 is for preparing the required sample installation jobs and creating a keyring file to pass the FTP user name and password information for IBM Installation Manager to access the SOAP Gateway repository. If you are installing SOAP Gateway on the same system as the SMP/E driving system: • If the SOAP Gateway repository file is on the same system as IBM Installation Manager, go to step 3 to install SOAP Gateway. • If the SOAP Gateway repository file is not on the same system, go to step 2.

2. **Target** This step is needed only when the SOAP Gateway repository file is on a different system from where the Installation Manager is installed. IBM Installation Manager requires a keyring for secure installation.

Sample job to edit	AEWTCKER
Description	Use this job to create a keyring file to store the user name and password information. Use the imutilsc command in Installation Manager to create the keyring file. For more information about credentials and the imutilsc command, see the IBM Installation Manager information center.
Step result	A keyring file is created.

3. **Target** Install SOAP Gateway by using the Installation Manager imcl command.

Sample job to edit	AEWTSINS
Description	This job installs SOAP Gateway by using the Installation Manager imcl -install command. You must edit and submit the job for each SOAP Gateway installation component (also known as a package) by specifying the package name for each run: • com.ibm.ims.sgw.imsserver.v31 • com.ibm.ims.sgw.imsbase.v31 • com.ibm.ims.sgw.imssoap.v31
Step result	SOAP Gateway is installed.

You must configure SOAP Gateway next before you can start the SOAP Gateway server.

Related tasks:

Installing SOAP Gateway on z/OS (Installation roadmap)
Use the provided installation roadmap to guide your installation. Installation must be completed by a system programmer who is familiar with installation tasks on a z/OS platform.

Related information:

Keyring files and credentials (IBM Installation Manager V1.5 information center)

Worksheet for installation scenario 2

Use this worksheet as you follow the installation steps to help keep track of the file system or directory names as necessary. Scenario 2 assumes that IBM Installation Manager for z/OS is already installed on the target system.

Before you begin, ensure that you have the location where the required FMIDs are processed by using SMP/E. Use the following table to record the directories or file systems where the required files and data sets are located on the SMP/E driving system.

Table 17. Locations for IMS Enterprise Suite and IBM Installation Manager libraries after SMP/E processing

Component (Library DDName)	Location
IBM Java SDK **Important:** The IBM Java SDK in the IMS Enterprise Suite Base Services component, which is referred to in the program directory, has been deprecated (APAR PI33917). See the PSP bucket for the latest supported Java version and download instructions. The instructions also specify how to order Java for z/OS through Shopz at no charge.	Your directory path:
IMS Enterprise Suite sample installation jobs (AAEWBASE and AAEWSAMP)	Your file system for the AAEWBASE and AAEWSAMP data sets:
SOAP Gateway repository (SIOGSHFS)	Default path, before your high-level qualifier, is `/usr/lpp/InstallationManagerRepository/JAHF311` Your directory path:

Transferring the sample jobs to the target system

1. **Target** (No sample job provided)

Parameters	Your value
A temporary PDS with a block size of 3200 on the target system to store the sample JCLs that will be transferred over from the driving system in sample job AEWDFTP in the next step	

2. Driving AEWDFTP

Parameters	Your value
target.site.com: The IP address or the fully qualified name of the target system.	
#hlqual: The high level qualifier of the data set where TCP/IP for z/OS is installed.	
username and **password**: The FTP username and password for the target system.	
#hlqual1: The high level qualifier specified in AEWALLOC used in the SMP/E process to install IMS Enterprise Suite..	
#hlqual2.TARGET.JCL: The PDS that you allocated on the target system in step 1 to store sample JCL jobs for installation.	

3. Target AEWTFTP

Parameters	Your value
driving.site.com: The IP address or the fully qualified name of the driving system.	
#hlqual: The high level qualifier of the data set where TCP/IP for z/OS is installed.	
username and **password**: The FTP username and password for the driving system	
#hlqual3.TARGET.JCL: the PDS on the target system for storing the SOAP Gateway installation and runtime configuration sample jobs.	
#hlqual4: The high level qualifier of the AAEWSAMP data set created during the SMP/E APPLY process of SOAP Gateway on the driving system.	
#hlqual5: The high level qualifier for the PROCLIB for storing the SOAP Gateway configuration member and server startup JCL procedure.	

Installing SOAP Gateway

1. Target AEWTSCFS (in AAEWBASE)

   AEWTSCFS:

Parameters	Your value
xxxxxx.dsn: The name of the data set where SOAP Gateway is to be installed, one for each SOAP Gateway component if the components are to be installed on different mount points.	
hhhhhh: The VOLSER ID to be used for the target file system.	
xxxxxx.path: The path to the mount point on the target system, one for each SOAP Gateway component if the components are to be installed on different mount points.	

If you are installing the IMSSERVER, IMSBASE, and IMSSOAP components on different file systems, record the data set name, VOLSER ID, and the path to the mount point for each of the component. This information is needed later for the sample installation job AEWTSINS and the sample configuration job AEWPOSIN.

2. **Target** AEWTCKER (If the SOAP Gateway repository file is on a different system from the system where IBM Installation Manager for z/OS is installed.)

Parameters	Your value
ipadress_of_driving_system: The IP address of the driving system that stores the SOAP Gateway repository file.	
path_to_soap_repository: Where on the driving system the SOAP Gateway repository file is located.	
your_user_name and **your_password**: The FTP user name and password to access the driving system where the SOAP Gateway repository file is located.	
keyring_filename: The path and the name of the keyring file.	
-PathPrefix-: The path to the root directory where IBM Installation Manager IBM Installation Manager for z/OS is installed.	

3. **Target** AEWTSINS (in AAEWBASE)

   See instructions that are provided in the sample job for information about where the required information is specified in previous jobs.

Configuring SOAP Gateway on z/OS

Before you can start the SOAP Gateway server, you must customize the provided sample jobs for runtime configuration and for starting and stopping the server.

The following sample jobs and configuration member must be customized:

- AEWPOSIN: Sample job to specify the location of the **imsserver**, **imsbase**, and **imssoap** components, the IBM Java SDK, and the server log file.
- AEWPARMF: Sample job to copy the Java load module into the partitioned data set extended (PDSE) used by SOAP Gateway.

- AEWIOGCF: Configuration member to configure SOAP Gateway runtime settings.
- AEWIOGPR: JCL procedure for starting and stopping SOAP Gateway.
- AEWIOGBP Sample job to specify the IBM Java SDK location and view the SOAP Gateway server properties by using the SOAP Gateway management utility commands from BPXBATCH.

Tip: If the correct authorization is not set as described in Section 6.1 in the *Program Directory for IMS Enterprise Suite*, the Java virtual machine (JVM) issues a JVMJ9VM082E error for lack of the required APF authorization to switch to an Integrated Facility for Applications (IFA) processor. The SOAP Gateway workload would be ineligible for a System z Application Assist Processor (zAAP). For more information about setting up user access, see "Setting up UID(0) access for SMP/E installation" on page 55.

To configure SOAP Gateway:

1. Edit and submit the AEWPOSIN sample job to specify the location of the **imsserver**, **imsbase**, and **imssoap** components, the IBM Java SDK, and the server log file.

Sample job to edit	AEWPOSIN

Steps	1. Change **aaaaa** to the directory that contains the **imsserver** component.
	2. Change **bbbbb** to the directory that contains the **imsbase** component.
	3. Change **ccccc** to the directory that contains the **imssoap** component.
	4. Change **ddddd** to the directory that contains the IBM Java SDK.
	5. If necessary, set the code page for server configuration and property files that are used by SOAP Gateway.
	The default value of `default` indicates that the server configuration and property files are converted from UTF-8 to the OMVS system code page.
	`export ENCODING=default; +`
	If you need to specify code pages because, for example, you use different code pages for OMVS and MVS, change the ENCODING value. The following example changes the ENCODING to IBM-277.
	`export ENCODING=IBM-277; +`
	6. Submit the job.

Step result	The SOAP Gateway component installation directory information is configured

2. Specify the **imsserver** component installation location in the AEWIOGCF configuration member.

Configuration member to edit	AEWIOGCF

Steps	1. Replace the variable *ddddd* with the value set for **-ServerPathPrefix-** in the AEWTSGCF sample job. The *ddddd* variable is in the following statement:
	`export AEWIOGPR_DIR=ddddd/imsserver`
	
	The `imsserver` directory name must remain intact. Do not modify this directory name.
	2. If you need to set up for FIPS 140-2 support:
	a. Remove the # sign in the beginning of the following line to uncomment it:
	`# IJO="$IJO -Dcom.ibm.jsse2.usefipsprovider=true"`
	b. Remove the # sign in the beginning of the following line to uncomment it to enable the SSL support for communications between SOAP and IMS Connect:
	`# IJO="$IJO -Dcom.ibm.ims.soap.sslProtocolType=TLSv1.2"`
	c. Remove the # sign in the beginning of the following line to uncomment it to enable HTTPS for communications between the external server and SOAP Gateway in the callout scenario:
	`# IJO="$IJO -Dcom.ibm.ims.soap.httpsProtocolType=TLSv1.2"`
	d. Remove the # sign in the beginning of the following line to uncomment it to assist troubleshooting security handshake issues:
	`# IJO="$IJO -Djavax.net.debug=ALL"`
	3. Save the file.

Step result	The **imsserver** component location is configured.

3. Edit and submit the AEWIOGBP sample job to specify the location of the **imsserver** component and the IBM Java SDK by using the SOAP Gateway management utility iogmgmt commands.

   Tip: This is an optional job that demonstrates how to issue the SOAP Gateway management utility -iogmgmt command by using an array.

Sample job to edit	AEWIOGBP
Steps	1. Change **-ServerPathPrefix-** to match what it is specified in the configuration file AEWTSGCF. 2. Change **-SMPPathPrefix-** to either the value specified for **-PathPrefix-** in sample job AEWJSMKD. 3. Submit the job.
Step result	The Java SDK location is set (by CMD[0]) and displayed (by CMD[1]). The SOAP Gateway server properties and their values are listed (by CMD[2]).

4. Edit the AEWIOGPR procedure for starting and stopping the SOAP Gateway server.

Procedure to edit	AEWIOGPR (in *hlqual5*.PROCLIB, as specified in sample job AEWTFTP)

Steps

1. Edit the AEWIOGPR procedure:
   a. Change *pppppppp* to the procedure name that meets your system requirement or convention. For example:
      ```
      //IMSESOAP PROC REGSIZE='0M',
      ```
   b. Change *#hlqual* to the high-level qualifier for the load library that contains the Java load module.
   c. Change *#LoadModPostfix* to 70 if you are using the 31-bit IBM Java, and to 76 if you are using the 64-bit Java. Ensure that this setting is consistent with what you specified to use in the AEWPOSIN sample job.
   d. Change *#hlqual2* to the value specified for DLIBPRE in the SEWALLOC sample job where the AAEWBASE data set is created.

      If the target system is different from the driving system, and sample jobs and installation and configuration files were transferred to the target system by using the AEWTFTP sample job, change *#hlqual2.AAEWBASE* to the PDS that you specified in AEWTFTP for storing the SOAP Gateway installation and runtime configuration sample jobs. For example:
      ```
      DSN=IMSES31.TARGET.JCL(AEWIOGCF)
      ```
2. Save the changes.

Step result The JCL procedure is configured to run as a started task.

SOAP Gateway runtime properties are configured, and the server startup and shutdown jobs are configured.

Next step: The user ID associated with the batch job that starts and stops the SOAP Gateway server requires that an OMVS segment be defined in RACF. You must define the OMVS segment and set up the RACF for the SOAP Gateway started task.

Related tasks:

"Configuring compliance for FIPS 140-2 and NIST SP800-131a" on page 97
You can configure SOAP Gateway to communicate with its clients and IMS Connect over secure sockets by using Java Secure Socket Extension files that are required by FIPS 140-2. In addition, NIST SP800-131a requires the use of TLS V1.2.

Installing SOAP Gateway on z/OS (Installation roadmap)
Use the provided installation roadmap to guide your installation. Installation must be completed by a system programmer who is familiar with installation tasks on a z/OS platform.

Defining the OMVS segment and user ID to set up RACF for the SOAP Gateway started task

z/OS

Use the RACF panels or TSO command prompts to define the user ID and OMVS segment to set up RACF for the SOAP Gateway JCL procedures.

If a unique UID is not set up, you might run into the following errors:

```
JVMJZBL2999T IOG00013E: Java DC5134 is not supported. Use the iogmgmt
command to update the Java path to point to the supported version 1.7.
```

To define the OMVS segment and user ID:

1. In the ISPF panel, select option 6.

2. Define the user ID and OMVS segment for installing and running SOAP Gateway in the home directory if they do not exist yet. The home directory must already exist.
   - If you are defining a new user ID, issue the following command:

     ```
     AU jjjjjj DFLTGRP(SYS1) OMVS(UID(nnnnn) HOME('hhhhh')
        PROGRAM( '/bin/sh'))
     ```

     jjjjjj Name of the started JCL procedure

     nnnnn
     OMVS user ID

     hhhhh
     Home directory for the OMVS segment
   - If you are making changes to an existing user (in PROC *jjjjjj*), such as adding an OMVS segment, or defining the HOME or PROGRAM directories for an existing OMVS segment, issue the following command:

     ```
     ALU jjjjjj DFLTGRP(SYS1) OMVS(UID(nnnnn) HOME('hhhhh')
        PROGRAM( '/bin/sh'))
     ```
3. Define the started task by issuing the following command:

   ```
   RDEF STARTED jjjjjj.* STDATA(USER(nnnnn) GROUP(gggg))
   ```

   jjjjjj Name of the started JCL procedure. For example, if AEWIOGPR is the name of the JCL procedure to start the SOAP Gateway server, replace *jjjjjj* with AEWIOGPR.

   nnnnn
   OMVS user ID

   gggg Default group for the specified user ID
4. Refresh the started class by issuing the following command:

   ```
   SETR RACLIST(STARTED) REFR
   ```

The user ID and the OMVS segment are properly set up for RACF. You can now start and stop the SOAP Gateway server by using the MVS /START command:

```
/START AEWIOGPR
```

If the *pppppppp* variable in the AEWIOGPR sample JCL procedure is set to IMSESOAP, the command would be:

```
/START IMSESOAP
```

Next step: Check the installation roadmap for additional configuration tasks for your environment, such as configuring IMS Connect, changing the SOAP Gateway server port number, configuring SOAP Gateway to run on zAAP, and migrating your web services from previous releases.

Sample jobs for installation and configuration

After the SMP/E processing of the IMS Enterprise Suite Base Services and SOAP Gateway FMIDs, a set of sample jobs and configuration members are provided in the SAEWBASE and SAEWSAMP data sets (or the AAEWBASE and AAEWSAMP data sets after SMP/E ACCEPT).

Sample jobs to transfer installation jobs for installing IBM Installation Manager and SOAP Gateway on a different LPAR

If SOAP Gateway is to be installed on a different LPAR from where SMP/E executes and processes the FMIDs, sample installation jobs for both the IBM Installation Manager and SOAP Gateway must be transferred to the target installation system.

For the steps and the sequence to edit and submit the jobs, choose your installation scenario from the installation roadmap in the "Installing SOAP Gateway on z/OS" on page 60 topic.

Table 18. Sample jobs in the AAEWSAMP data set to transfer installation jobs to a different LPAR

Job name	Description
AEWDALPX	Sample job to allocate, create the mount point, and mount a temporary file system (*hfsdsn*) to store the IBM Installation Manager PAX file that will be created later in AEWDIMPX. Use AEWDZALP for zFS.
AEWDZALP	Sample job to allocate, create mount point, and mount a temporary file system (*zfsdsn*) to store the IBM Installation Manager PAX file that will be created later in AEWDIMPX. Use AEWDALPX for HFS.
AEWDIMPX	Create a PAX file on the temporary file system allocated in AEWDALPX or AEWDZALP that contains the IBM Installation Manager installation kit after SMP/E processing. This PAX file needs to be transferred to each LPAR that SOAP Gateway is to be installed.
AEWDFTP	Sample job to transfer the following sample JCL jobs for preparing the file systems and transferring the Installation Manager installation PAX file through FTP to the target system: • AEWTALLO • AEWTFTP • AEWTUMDL • AEWTUNPX • AEWTZALC • AEWTZUMD
AEWTALLO	Sample job to allocate and mount two file systems (HFS) on the target system to store the Installation Manager installation kit and to hold the Installation Manager kit PAX file. This job also allocates a PDS to store additional sample JCL jobs for installing the Installation Manager and SOAP Gateway. Use AEWTZALC for zFS.
AEWTZALC	Sample job to allocate and mount two file systems (zFS) on the target system to store the Installation Manager installation kit and to hold the Installation Manager kit PAX file. This job also allocates a PDS to store additional sample JCL jobs for installing the Installation Manager and SOAP Gateway. Use AEWTALLO for HFS.

Table 18. Sample jobs in the AAEWSAMP data set to transfer installation jobs to a different LPAR (continued)

Job name	Description
AEWTFTP	Sample that runs the FTP program on the target system to: • Get the Installation Manager PAX file (`impax.pax`) to the temporary file system created in AEWTALLO. • Get the following sample installation JCL jobs from the driving system to the PDS created by AEWTALLO: – For installing and uninstalling Installation Manager: - GIN2ADMN - GIN2CFS - GIN2INST - GIN2UNIN - GIN3CMD – For installing and uninstalling SOAP Gateway: - AEWTSCFS - AEWTSINS - AEWTSUNI - AEWTCKER (optional, if SOAP Gateway repository is on a different system from where the IBM Installation Manager is installed) - AEWTHVJA, AEWTMVJA: These jobs are obsolete because the Java SDK that is provided with the IMS Enterprise Suite Base Services component has been deprecated. See the PSP bucket for the latest supported Java version and download instructions. – For configuring SOAP Gateway runtime and starting or stopping the SOAP Gateway server: - AEWIOGBP - AEWIOGCF - AEWIOGPR - AEWPARMF: This job is obsolete because the Java SDK that is provided with the IMS Enterprise Suite Base Services component has been deprecated. - AEWPOSIN
AEWTUNPX	Sample job to unpax the Installation Manager PAX file into the directory structure created in AEWTALLO.
AEWTUMDL	Sample job to unmount and delete the temporary file system allocated in AEWTALLO (*hfsdsn2*). Use AEWTZUMD for zFS.
AEWTZUMD	Sample job to unmount and delete the temporary file system allocated in AEWTZALC (*zfsdsn2*). Use AEWTUMDL for HFS.
AEWDUMDL	Sample job to unmount and delete the temporary file system allocated in AEWDALPX (*hfsdsn*).

Sample installation jobs to install SOAP Gateway

The following sample jobs and configuration member are provided to assist your configuration and installation of SOAP Gateway by using the IBM Installation Manager command line.

Table 19. Sample installation jobs to install SOAP Gateway

Job name	Data set	Description
AEWTSCFS	AAEWBASE	Sample job to create the file system, HFS or zFS, for installing SOAP Gateway on one mount point.
AEWTCKER	AAEWSAMP	Sample job to create a keyring file that is required by the Installation Manager to pass the username and password information when it accesses a repository through FTP.
AEWTSINS	AAEWBASE	Sample job to install SOAP Gateway using the Installation Manager. You must modify the sample job to install the three SOAP Gateway packages in three separate imcl -install commands.
AEWTSUNI	AAEWBASE	Sample job to uninstall SOAP Gateway. You must modify the sample job to uninstall the three SOAP Gateway packages in separate imcl -uninstall commands.

Sample runtime and configuration jobs for SOAP Gateway

Table 20. Sample runtime and configuration jobs in the AAEWSAMP data set for SOAP Gateway

Job name	Description
AEWIOGCF	Sample configuration member specify the SOAP Gateway server runtime settings.
AEWIOGBP	Sample job to execute SOAP Gateway management utility commands from BPXBATCH and to configure the IBM Java SDK location.
AEWIOGPR	Sample job to start the SOAP Gateway server.
AEWPOSIN	Sample job to specify the configuration information for the locations of the **imsserver** component, **imsbase** component, the **imssoap** component, and Java.

Installing SOAP Gateway on distributed platforms

Use the provided installation roadmap to guide your installation. Install SOAP Gateway on distributed platforms by using the IBM Installation Manager.

Prerequisites:

1. Check the "System requirements" on page 41 and "Software requirements" on page 43.
2. Review the "Planning for installation" on page 45 information to understand your configuration and setup requirements, server upgrade path, and skill requirements.
3. Save a backup copy of your web services files, or any customized scripts or server properties before you upgrade to IMS Enterprise Suite Version 3.1.

   Although you can use the SOAP Gateway management utility iogmgmt -migrate command to migrate web services and server properties that are managed by the utility, it is always a good idea to save a copy of the following files:

- *install_dir*/imsbase/conf/log4j.properties
- *install_dir*/imssoap/WEB-INF/wsjaas.conf
- Files under the *install_dir*/imssoap/xml and *install_dir*/imssoap/wsdl directories
- Your .aar files in the *install_dir*/imssoap/WEB-INF/services directory.

Installation roadmap for SOAP Gateway on distributed platforms

Use the following installation roadmap to guide your installation. For more information on applying services, see "Applying maintenance services on distributed platforms" on page 116.

Table 21. SOAP Gateway installation roadmap for distributed platforms

Step	Description of task	Role
1	Install SOAP Gateway: 1. Prepare to install SOAP Gateway using the IBM Installation Manager. 2. Install SOAP Gateway by using the IBM Installation Manager. 3. Optionally, install SOAP Gateway as a Windows service. .	SOAP Gateway server administrator
2	Apply any maintenance update as described in the "Release notes for IMS Enterprise Suite V3.1 SOAP Gateway" on page 1.	SOAP Gateway server administrator
3	Configure the SOAP Gateway run time: • If the IBM Java SDK is not installed with SOAP Gateway, configure the Java SDK location. • (Optional) Configure the SOAP Gateway server port numbers. • (Optional) Configure the SOAP Gateway log file location. • (Optional) Specify how you want to log SOAP Gateway messages by setting the trace level.	SOAP Gateway server administrator
4	Configure IMS Connect for SOAP Gateway. IMS Connect manages the translation of message headers on input and output messages and provides a point of control to modify, route, and check security for messages from and to SOAP Gateway.	System programmer
5	Verify the installation by using the SOAP Gateway Installation Verification Program (IVP).	SOAP Gateway server administrator and system programmer
6	• If you are upgrading from IMS Enterprise Suite Version 2.2 SOAP Gateway, migrate your web services. See "Migrating from IMS Enterprise Suite Version 2.2 SOAP Gateway" on page 106. • If you are upgrading from Version 2.1 SOAP Gateway, migrate your web services, see "Migrating from IMS Enterprise Suite Version 2.1 SOAP Gateway" on page 104. • If you are upgrading from Version 1.1 SOAP Gateway, migrate to V2.1 first, and then migrate from V2.1 to V3.1.	SOAP Gateway server administrator

Table 21. SOAP Gateway installation roadmap for distributed platforms (continued)

Step	Description of task	Role
7	Optionally, you can install multiple SOAP Gateway server instances that share one JVM.	SOAP Gateway server administrator

Preparing to install SOAP Gateway using IBM Installation Manager

Before you install IMS Enterprise Suite SOAP Gateway, you must download the SOAP Gateway installation repository file, and configure the installation repository.

Prerequisite:

1. Go to the IMS Enterprise Suite download site and log in.
2. Select **IMS Enterprise Suite SOAP Gateway** and follow the instructions to go to the IMS Enterprise Suite SOAP Gateway download page.
3. Follow the instructions on the page to download the latest supported Java.
4. You must have IBM Installation Manager Version 1.5.3 or later installed. If you do not yet have IBM Installation Manager, follow the link to download IBM Installation Manager.
5. Select the installation repository file for SOAP Gateway. The repository file is a compressed file.
6. If you plan to install SOAP Gateway in silent mode, select the provided response file and its data type definition (DTD) file for download.
7. Click **Download now** to download the selected files.

To prepare your environment to install SOAP Gateway with IBM Installation Manager:

1. If you are installing the Installation Manager:
   a. Extract the downloaded compressed file for IBM Installation Manager.
   b. Run the IBM Installation Manager installer to install IBM Installation Manager.
      - Windows On Windows, run the `install.exe` file.
      - Linux On Linux on System z, run the `install` file.
   c. Follow the pages in the installer wizard to install the Installation Manager.

   Tip: For silent installation of the Installation Manager, see the information about silent installation of the Installation Manager in the IBM Installation Manager information center.
2. Store the compressed SOAP Gateway repository file in a location that is accessible by the workstation that SOAP Gateway is to be installed on.
3. To install SOAP Gateway by using the graphical user interface (wizard mode) of the Installation Manager:
   a. Start the Installation Manager.
      - Windows On Windows, click **Start** > **All Programs** > **IBM Installation Manager** > **IBM Installation Manager**
      - Linux On Linux on System z, go to `/opt/IBM/InstallationManager/eclipse` and run the IBMIM application.

a. Add a repository and specify the location, including the file name, where the SOAP Gateway repository file is located. The compressed file does not need to be extracted.
   1) Click **File** > **Preferences**. The Preferences window opens.
   2) On the Repositories page, click **Add Repository**, and then click **Browse** to navigate to the SOAP Gateway repository file.
   3) Click **OK** to add the repository location to the list.
   4) Click **OK** again to go back to the IBM Installation Manager main screen.

Tip: To install SOAP Gateway in silent mode, configuration of the repository location is specified as part of the response file setup for silent installation. See "Installing SOAP Gateway in silent mode" on page 91.

You can now proceed to "Installing SOAP Gateway using IBM Installation Manager."

Installing SOAP Gateway using IBM Installation Manager

Use the IBM Installation Manager in wizard mode to install SOAP Gateway on distributed platforms.

Prerequisite: Complete the steps in Preparing to install.

If you cannot use the IBM Installation Manager graphical user interface (also known as the wizard mode) and must use the silent mode, instead of following the steps in this topic, follow the steps in "Installing SOAP Gateway in silent mode" on page 91.

To install SOAP Gateway:

1. In IBM Installation Manager, click **Install**. IMS Enterprise Suite SOAP Gateway shows as four packages to install.

   Important: All three parts (IMSSERVER, IMBASE, and IMSSOAP) must be selected to be installed at the same time. The three parts are provided as separate packages so that you can install them on different partitions or file systems for flexibility and ease of maintenance.
2. Select the check boxes for all three parts of SOAP Gateway.
3. Click **Next**.
   a. Click **Yes** to install the updated version.
   b. When the update is complete, click **OK**. The new version of the IBM Installation Manager is automatically restarted.
   c. Go back to step 1 to install SOAP Gateway.
4. Click to accept the terms in the license agreement and click **Next**.
5. On the Location page, specify the installation directory for each of the packages.

   By default, all packages are installed under one directory:

   - Windows For Windows, the default is `C:\Program Files\IBM\IMS Enterprise Suite V3.1\SOAP Gateway\[imsserver, imsbase, imssoap]`

     On Windows 7, the default path is `C:\Program Files (x86)\IBM\IMS Enterprise Suite V3.1\SOAP Gateway\[imsserver, imsbase, imssoap]`.

Because parentheses in path names are not supported by SOAP Gateway, you must manually change the installation path to not include any parentheses.

- Linux For Linux on System z, the default is `/opt/IBM/IMS_Enterprise_Suite_V3.1/SOAP_Gateway/[imsserver, imsbase, imssoap]`

If you want to change the default location, or install the packages on different directories, you must complete additional steps:

a. Click to select each package.
b. In the Installation Directory field, specify the new location, or click **Browse** to navigate to the new location.

   Important:

   1) The installation path cannot include parentheses. The SOAP Gateway management utility commands do not work when the path it needs to access contains parentheses.
   2) The three SOAP Gateway parts must be installed in the `imsserver`, `imsbase`, and `imssoap` directories. After you specify your location, the last directory names must be `imsserver`, `imsbase`, and `imssoap`, respectively.

   For example, the following directory paths are valid:

   - `c:\file_path1\imses31\soap\imsserver`
   - `d:\file_path2\imses31\soap\imsbase`
   - `e:\file_path3\imses31\soap\imssoap`

   The following directory paths are not valid:

   - `c:\file_path1\imses31\soap\server`
   - `d:\file_path2\imses31\soap\base`
   - `e:\file_path3\imses31\soap\soap`

6. Click **Next**.
7. On the Features page, click **Next**.
8. Verify your selection of features and click **Next**.
9. On the Summary page, review the summary information for the installation directory, the package to install, available disk space, and required disk space. Click **Install** to start the installation. A status bar shows the installation status.
10. When the installation is complete, click **Finish**.

IMS Enterprise Suite SOAP Gateway is now installed.

Windows For Windows, a program group is added under **Start** > **All Programs** > **IBM IMS Enterprise Suite V3.1** > **SOAP Gateway**.

Go back to the installation roadmap to continue the installation steps. You must configure IMS Connect and OTMA for SOAP Gateway before you can verify that the SOAP Gateway installation is successful. You might also want to configure the SOAP Gateway log file location, or change the default server listening port and shutdown port.

Installing SOAP Gateway in silent mode

If you cannot use the IBM Installation Manager graphical user interface (wizard mode) to install SOAP Gateway, you must install in silent mode. Silent mode is

based on a defined response file and is launched from the command line or a batch file. Response files are XML files that contain the responses or input to the Installation Manager.

Prerequisites:

- The required version of IBM Installation Manager must be already installed.
- The requires SOAP Gateway repository file must be downloaded and placed in a location that is accessible to the Installation Manager.
- The required response file (`IMSES31SOAPResponseFile.xml`) and its data type definition (`IMSES31SOAPConfig.dtd`) file must be downloaded from the IMS Enterprise Suite download site. The downloaded compressed file must be extracted.
- The `IMSES31SOAPConfig.dtd` file must be stored in the `eclipse\tools\` directory where IBM Installation Manager is installed, for example, `C:\Program Files\IBM\Installation Manager\eclipse\tools`.

To install SOAP Gateway, edit `IMSES31SOAPConfig.dtd` to specify the repository location and the installation directories.

1. Open `IMSES31SOAPConfig.dtd` in a text editor.
2. Specify the SOAP Gateway repository by modifying this line of code:

   ```
   <!-- path to the SOAP Gateway repository-->
   <!ENTITY repolocation "path to the SOAP Gateway Repository here">
   ```

   For example:

   ```
   <!-- path to the SOAP Gateway repository-->
   <!ENTITY repolocation "C:\IBM\IMSES31SOAP\soap_gateway_repository_filename.zip">
   ```

3. Customize the following lines to specify the absolute path to the directory to install each of the SOAP Gateway component:

   ```
   <!-- Paths to install each of the SOAP Gateway Components -->
   <!ENTITY imsserverpath "C:\Program Files\IBM\IMS Enterprise Suite V3.1\SOAP Gateway\imsserver">
   <!ENTITY imsbasepath "C:\Program Files\IBM\IMS Enterprise Suite V3.1\SOAP Gateway\imsbase">
   <!ENTITY imssoappath "C:\Program Files\IBM\IMS Enterprise Suite V3.1\SOAP Gateway\imssoap">
   ```

   Important: The last directory name must remain `imsserver`, `imsbase`, and `imssoap`. Changing the directory name would result in installation errors.

   By default, all packages are installed under one directory:

   - Windows For Windows, the default is `C:\Program Files\IBM\IMS Enterprise Suite V3.1\SOAP Gateway\[imsserver, imsbase, imssoap]`
   - Linux For Linux on System z, the default is `/opt/IBM/IMS_Enterprise_Suite_V3.1/SOAP_Gateway/[imsserver, imsbase, imssoap]`

4. Specify the absolute path to the directory where the IBM Java SDK is installed by customizing the following line:

   ```
   <!-- Path to install Java -->
   <!ENTITY javaPath "C:\Program Files\IBM\IMS Enterprise Suite V3.1\SOAP Gateway\java">
   ```

   Comment out this Java SDK offering in the `IMSES31SOAPResponseFile.xml` file:

   ```
   <!--Comment the next line if you do not want to install the IBM Java SDK V7.0-->
   <!-- <offering id='com.ibm.ims.sgw.java.sdk.v7'
     profile='IBM Java SDK V7.0' features='javasdk' installFixes='none'/> -->
   ```

   Important:

   - The version of the IBM Java SDK that is installed by IBM Installation Manager has been deprecated (IMS Enterprise Suite Version 3.1 APAR

PI33917). Visit the IMS Enterprise Suite downloads page and navigate to the page for SOAP Gateway. Follow the instructions on the page to download the latest supported Java version.

- All other lines in the `IMSES31SOAPConfig.dtd` and `IMSES31SOAPResponseFile.xml` files must remain intact.

5. Linux If you are installing on Linux on System z, comment out the system configuration section for Windows, and uncomment the section for Linux on System z.

   The DTD has Windows installation configuration as the default, with the Linux on System z configuration information commented out. Comment out the first section for Windows, and uncomment the section for Linux on System z:

```
<!-- system configuration windows-->
<!-- Operating System, windowing system, architecture, and language -->
<!--
<!ENTITY sysconfig "<data key='cic.selector.os' value='win32'/>
 <data key='cic.selector.ws' value='win32'/>
 <data key='cic.selector.arch' value='x86'/>
 <data key='cic.selector.nl' value='en'/>">
-->

<!-- example zLinux system configuration -->
<!ENTITY sysconfig "<data key='cic.selector.os' value='linux'/>
<data key='cic.selector.ws' value='gtk'/>
<data key='cic.selector.arch' value='s390'/>
<data key='cic.selector.nl' value='en'/>">
```

6. Save your changes.
7. Go to the `eclipse\tools\` directory in the Installation Manager.
8. Install SOAP Gateway:

   - Windows On Windows, enter the following command:

     `imcl.exe input path_To_responseFile/IMSES31SOAPResponseFile.xml -acceptLicense`

   - Linux On Linux on System z, enter the following command:

     `./imcl input path_To_responseFile/IMSES31SOAPResponseFile.xml -acceptLicense`

Installing additional instances

If you are installing additional instances of SOAP Gateway, you must change the profile ID for each of the SOAP components in the `IMSES31SOAPResponseFile.xml` file. Otherwise, the Installation Manager will install the new instance over the existing instance, ignoring the locations you specify for the new instance.

To install an additional instance:

9. Make a copy of the XML and DTD file for installing a different instance of SOAP Gateway. Rename the files to ease association of the files with the SOAP Gateway instance.
10. Modify the DTD file name accordingly in the following `DOCTYPE includes` statement in the XML file so that the correct DTD file is referenced by the XML file.

```
<!DOCTYPE includes [
<!ENTITY % entity SYSTEM "IMSES31SOAPConfig.dtd">
```

11. In the XML file, change the profile name for each of the component. The following example appends the characters `_2` to each of the profiles to indicate that the profile is for the second SOAP Gateway installation and yet it clearly identifies the SOAP Gateway component that the profile is for.

```
<!ENTITY imsserverProfile "IMS Enterprise Suite V3.1 SOAP Gateway - Part 1 of 3, IMSSERVER_2">
<!ENTITY imsbaseProfile "IMS Enterprise Suite V3.1 SOAP Gateway - Part 2 of 3, IMSBASE_2">
<!ENTITY imssoapProfile "IMS Enterprise Suite V3.1 SOAP Gateway - Part 3 of 3, IMSSOAP_2">
```

12. Modify the XML file to specify the absolute path to the directory to install each of the SOAP Gateway components.
13. If you do not want to install the IBM Java SDK, comment out the Java SDK offering in the XML file.
14. Use the Installation Manager command line imcl command to install SOAP Gateway.

For information about obtaining SOAP Gateway fixes or enhancements, see the topic on Obtaining and installing IMS Enterprise Suite product updates on distributed platforms

After the installation, you must configure IMS Connect and OTMA before you can verify that the SOAP Gateway installation is successful. You might want to configure the SOAP Gateway log file location, and change the default port 8080 where SOAP Gateway listens for SOAP requests.

1. Configure IMS Connect for SOAP Gateway.
2. Configure the Java SDK location.
3. Optionally, register the SOAP Gateway server as a Windows service.
4. Optionally, configure the SOAP Gateway log file location.
5. Optionally, configure the SOAP Gateway server port number.
6. Verify the installation by using the SOAP Gateway Installation Verification Program (IVP).

Installing SOAP Gateway as a Windows service

Windows

After SOAP Gateway is installed by using the IBM Installation Manager, use the SOAP Gateway management utility to register the SOAP Gateway server as a Windows service.

Prerequisite:

SOAP Gateway must be installed before you can proceed with the following steps.

To install SOAP Gateway as a Windows service:

1. Go to the *install_dir*/imsserver/deploy subdirectory under the SOAP Gateway installation directory.
2. Issue the following command:

```
iogmgmt -service –install
```

An IOG30010I message is displayed, indicating that the installation was successful, and that SOAP Gateway is registered as a Windows service.

Configuring SOAP Gateway

For z/OS, you must specify the IBM Java SDK location before you can start the SOAP Gateway server. Optionally, for all supported platforms, you can change the default log file location, server listening port, or shutdown port. For z/OS, you can enable a System z Application Assist Processor (zAAP).

Specifying the Java SDK location

Before starting the SOAP Gateway server, the Java SDK installation location must be identified to the server. Use the SOAP Gateway management utility to specify the location.

z/OS For the z/OS platform, the IBM Java SDK in the IMS Enterprise Suite Base Services component has been deprecated. See the PSP bucket for the latest supported Java version and download instructions. The instructions also specify how to order Java for z/OS through Shopz at no charge.

If you configured sample job AEWIOGBP to run the SOAP Gateway management utility `iogmgmt -prop -u -java -h $IOGJH` command, your Java SDK location is already set.

Windows Linux For distributed platforms, specifying the Java SDK location is required only if the Java SDK is moved after installation or you are sharing the JVM with multiple SOAP Gateway server instances.

To specify the Java SDK location:

1. Go to the location of the SOAP Gateway management utility: *install_dir*/imsserver/deploy.
2. Issue the following command to specify the Java location. For example, for the z/OS platform:

   ```
   iogmgmt -prop -u -java -h -PathPrefix-/usr/lpp/ims/imses/V3R1/java170/SRx/FPx/J7.0/
   iogmgmt -prop -u -java -h -PathPrefix-/usr/lpp/ims/imses/V3R1/java170/SRx/FPx/J7.0_64
   ```

   The location differs depending on whether you are using 64-bit or 31-bit Java.

Configuring the SOAP Gateway log file location

By default, the log files are written to the *install_dir*/imsbase/logs directory. To change the log file location, use the SOAP Gateway management utility.

To change the log file location:

1. Go to the directory for the SOAP Gateway management utility. Change directory to *install_dir*/imsserver/deploy.
2. Set the full path to the server log by issuing the following command:

   ```
   iogmgmt -prop -u -f C:\full_path_to\soap\logs
   ```

   If the log file path contains a space, you must enclose the entire path in quotation marks:"*C:\full_path_with spaces to\soap\logs*"SOAP Gateway must have the permission to write to the directory.
3. Verify that the update was successful by reviewing the IOGD0123I message in the log.

Related reference:

"-prop: Set SOAP Gateway properties" on page 450
Use the -prop command to modify the SOAP Gateway server properties.

Configuring the SOAP Gateway server port numbers

To run the SOAP Gateway server on a port other than the default port 8080, or to change the default shutdown port of 8005, use the SOAP Gateway management utility and then restart the server.

Important: Each instance of the SOAP Gateway server must have a unique listening port and a unique shutdown port. The shutdown port must be available when the server is starting up.

To change the port numbers:

1. Change directory to the *install_dir*/imsserver/deploy directory, where the SOAP Gateway management utility is located.
2. To specify the new port number, use the following SOAP Gateway management utility command:

   ```
   iogmgmt -prop -u -p port_number
   ```
3. To specify the new shutdown port number, use the following SOAP Gateway management utility command:

   ```
   iogmgmt -prop -u -d shutdown_port_number
   ```

   Tip: You can specify both ports in one command:

   ```
   iogmgmt -prop -u -p port_number -d shutdown_port_number
   ```
4. Restart the server.
   a. To stop the SOAP Gateway server, see "SOAP Gateway server shutdown options" on page 293 for the command or the shell script to run, depending on your platform.
   b. To start the SOAP Gateway server, see "SOAP Gateway server startup options" on page 292 for the command or the shell script to run, depending on your platform.

The server port number is changed.

You can now verify the installation by using the SOAP Gateway Installation Verification Program (IVP).

Related reference:

"-prop: Set SOAP Gateway properties" on page 450
Use the -prop command to modify the SOAP Gateway server properties.

Configuring SOAP Gateway to run on a zAAP

z/OS

The SOAP Gateway server runs in the Java Virtual Machine (JVM) environment. The IBM Java SDK includes an option to run the eligible server workload on a System z Application Assist Processor (zAAP), if one is available, rather than on a central processor. To reduce the processing costs associated with the SOAP Gateway server workload, you can configure SOAP Gateway to run on a zAAP.

Prerequisites:

- SOAP Gateway must be already installed.
- Ensure that your System z hardware configuration includes an available zAAP.

To configure SOAP Gateway to run on a zAAP:

1. Add the required RACF security permissions for your administrative user ID. Your user ID must have the same permissions as those required for the initial installation of the IBM IMS Enterprise Suite V3.1 Base Services. See the for more information.

2. Configure the IBM SDK for z/OS, Java Technology Edition. The Java Virtual Machine must be configured with the command-line option `-Xifa:on`, which is the default value.
3. Activate the IFA option for SOAP Gateway with the following command:

   ```
   iogmgmt -prop -u -java -i on
   ```

   You can view the IFA setting with the following command:

   ```
   iogmgmt -view -java -i
   ```
4. Restart the SOAP Gateway server.

The eligible SOAP Gateway workload now runs on available zAAPs.

Optionally, you can modify the IEAOPTxx member of the SYS1.PARMLIB data set to tune how z/OS prioritizes and routes zAAP workloads.

Related information:

-Xifa
For more information about the -Xifa command line option, see *IBM User Guide for Java V7 on z/OS*.

 zSeries Application Assist Processor (zAAP) Implementation (pdf)
For more information about zAAP requirements and configuration, see *zSeries Application Assist Processor (zAAP) Implementation* from IBM Redbooks®.

Configuring compliance for FIPS 140-2 and NIST SP800-131a

You can configure SOAP Gateway to communicate with its clients and IMS Connect over secure sockets by using Java Secure Socket Extension files that are required by FIPS 140-2. In addition, NIST SP800-131a requires the use of TLS V1.2.

Prerequistie:

To enable FIPS, you need to specify to use the IBM Java Cryptographic Extension (JCE) FIPS Provider, IBMJCEFIPS. This cryptographic module supports FIPS-approved cryptographic operations through the Java APIs.

1. Stop the SOAP Gateway server if it is running.
2. Specify to use the IBMJCEFIPS FIPS provider. To do so, modify the `java.security` file in the IBM Java SDK to enable FIPS.

   Important: The following steps are required each time you update the IBM Java SDK.

   a. Go to *java_install_dir*`/jre/lib/security` directory.
   b. Save a copy of the existing `java.security` file as a backup. For example, name the backup copy `java.security.nofips`.
   c. In a text editor, open the `java.security` file and make the following changes.
      1) Find the following the list of security providers. The list might look as follows:

         ```
         security.provider.1=com.ibm.jsse2.IBMJSSEProvider2
         security.provider.2=com.ibm.crypto.provider.IBMJCE
         security.provider.3=com.ibm.security.jgss.IBMJGSSProvider
         security.provider.4=com.ibm.security.cert.IBMCertPath
         security.provider.5=com.ibm.security.sasl.IBMSASL
         security.provider.6=com.ibm.xml.crypto.IBMXMLCryptoProvider
         ```

```
security.provider.7=com.ibm.xml.enc.IBMXMLEncProvider
security.provider.8=com.ibm.security.jgss.mech.spnego.IBMSPNEGO
security.provider.9=sun.security.provider.Sun
```

2) Add the following line as the first line to enable FIPS:

```
security.provider.1=com.ibm.crypto.fips.provider.IBMJCEFIPS
```

3) Because the IBMJCEFIPS provider is now the first provider, increase the existing provider numbers by 1:

```
security.provider.1=com.ibm.crypto.fips.provider.IBMJCEFIPS
security.provider.2=com.ibm.jsse2.IBMJSSEProvider2
security.provider.3=com.ibm.crypto.provider.IBMJCE
security.provider.4=com.ibm.security.jgss.IBMJGSSProvider
security.provider.5=com.ibm.security.cert.IBMCertPath
security.provider.6=com.ibm.security.sasl.IBMSASL
security.provider.7=com.ibm.xml.crypto.IBMXMLCryptoProvider
security.provider.8=com.ibm.xml.enc.IBMXMLEncProvider
security.provider.9=com.ibm.security.jgss.mech.spnego.IBMSPNEGO
security.provider.10=sun.security.provider.Sun
```

4) Towards the end of the file, locate the section that specifies the default key and trust manager factory algorithms for SSL.

```
ssl.KeyManagerFactory.algorithm=IbmX509
ssl.TrustManagerFactory.algorithm=PKIX
```

5) Append the following two lines:

```
ssl.SocketFactory.provider=com.ibm.jsse2.SSLSocketFactoryImpl
ssl.ServerSocketFactory.provider=com.ibm.jsse2.SSLServerSocketFactoryImpl
```

The resulting section might look as follows:

```
ssl.KeyManagerFactory.algorithm=IbmX509
ssl.TrustManagerFactory.algorithm=PKIX
ssl.SocketFactory.provider=com.ibm.jsse2.SSLSocketFactoryImpl
ssl.ServerSocketFactory.provider=com.ibm.jsse2.SSLServerSocketFactoryImpl
```

6) For NIST SP800-131a compliance, check if the following lines are present. If not, add them to the end of the file. These lines disable cryptographic algorithms that are deemed unacceptable by the SP800-131a standard.

```
jdk.tls.disabledAlgorithms = RSA keySize < 2048, DSA keySize < 2048, EC keySize < 224, MD5
jdk.certpath.disabledAlgorithms = RSA keySize < 2048, DSA keySize < 2048, EC keySize < 224, SHA1, MD5
```

d. Save your changes.

3. Specify to use the FIPS provider module, and for NIST SP800-131a, to set TLS v1.2 as the protocol for both HTTPS and SSL communications in the `server.xml` file.

a. Edit the `server.xml` file in the *install_dir*`/imsbase/conf/master` directory. In the Connector element, add the following attributes and values if they do not exist yet:

```
SSLEnabled="true"
sslEnabledProtocols="TLSv1.2"
```

Tip: TLS v1.2 is required for NIST SP800-131a. If the sslEnabledProtocols attribute value contains a spelling error or the cases do not match, the server automatically adopts a lower SSL if multiple protocols are specified.

b. Remove `sslProtocol="SSL"` if this attribute exists in the `server.xml` file.

c. Go to the *install_dir*`/imsserver/bin` directory.

d. Take one of the following steps, depending on your platform.

- z/OS Modify the AEWIOGCF sample JCL job as described in Step 3 in "Configuring SOAP Gateway on z/OS" on page 80, if you have not yet done so.

This file contains several settings to enable the FIPS provider and TLS V1.2 for communication wit IMS Connect and SOAP Gateway clients.

1) Remove the # sign in the beginning of the following line to uncomment it:

```
# IJO="$IJO -Dcom.ibm.jsse2.usefipsprovider=true"
```

2) Remove the # sign in the beginning of the following line to uncomment it to enable the SSL support for communications between SOAP and IMS Connect:

```
# IJO="$IJO -Dcom.ibm.ims.soap.sslProtocolType=TLSv1.2"
```

3) Remove the # sign in the beginning of the following line to uncomment it to enable HTTPS for communications between the external server and SOAP Gateway in the callout scenario:

```
# IJO="$IJO -Dcom.ibm.ims.soap.httpsProtocolType=TLSv1.2"
```

4) Remove the # sign in the beginning of the following line to uncomment it to assist troubleshooting security handshake issues:

```
# IJO="$IJO -Djavax.net.debug=ALL"
```

5) Add the following line for NIST SP800-131a compliance:

```
IJO="$IJO -Dcom.ibm.jsse2.sp800-131=strict"
```

6) Save your changes.

- Linux Windows Replace the SOAP Gateway startup batch file or shell script with the version that supports FIPS settings.

  The FIPS-enabled versions are:

  – `iogstart.sh.secure`

  – `iogstart.bat.secure`

  These files are provided so minimal manual modification is needed, which could be error-prone.

  1) Linux For Linux for System z:

     a) Rename `iogstart.sh` to `iogstart.sh.nonfips`.

     b) Rename the FIPS-compliant version of the server startup shell script `iogstart.sh.secure` to `iogstart.sh`.

  2) Windows For Windows:

     a) Rename `iogstart.bat` to `iogstart.bat.nonfips`.

     b) Rename the FIPS-compliant version of the server startup batch file `iogstart.bat.secure` to `iogstart.bat`.

  3) For NIST SP800-131a compliance, in the new `iogstart.sh` or `iogstart.bat`, search for the following line:

     ```
     -Dcom.ibm.ims.soap.sslProtocolType=TLSv1.2
     ```

     There are two instances of this line. Append the following line to both instances:

     ```
     -Dcom.ibm.jsse2.sp800-131=strict
     ```

     The modified line would look as follows:

     ```
     ... -Dcom.ibm.ims.soap.sslProtocolType=TLSv1.2 -Dcom.ibm.jsse2.sp800-131=strict ...
     ```

  4) Save your changes.

4. Restart the server.

SOAP Gateway is enabled for FIPS.

In addition to enabling FIPS on the SOAP Gateway server:

- IMS Connect must be configured to match the required version of TLS (TLS v1.2 for NIST SP800-131a) and required cipher suites strength.

  For more information about IMS Connect SSL setup, see the "Configuring IMS Connect for SOAP Gateway" on page 101 topic.
- Connection bundles must be created or updated to use the STRONG encryption type for SSL connections with IMS Connect.

  For more information about IMS Connect SSL setup, see the "-conn: Create, update, or delete a connection bundle" on page 435 topic.
- For the callout scenario, the connection bundle must include the provider keystore and truststore information.
- For client applications:
  - Java clients must be run with FIPS enabled. If you are using the IBM Java SDK in IMS Enterprise Suite, turn on the Java FIPS provider flag by using the following system property:

    ```
    -Dcom.ibm.jsse2.usefipsprovider=true
    ```

    You must also use the `java.security.fips` file for the pre-configured security settings.

    For other versions of Java, check their documentation.
  - For NIST SP800-131a, the keystore and truststore that you create must meet the minimum requirements of 112-bit key strength or a key length of 2048. Java clients must be run with the HTTPS protocol flag turned on by using the following system property:

    ```
    -Dhttps.protocols=TLSv1.2
    ```
  - For ease of troubleshooting, you might want to turn on debugging:

    ```
    -Djavax.net.debug=ALL
    ```

Related concepts:

"FIPS 140-2 and NIST SP800-131a" on page 39
Federal Information Processing Standards (FIPS) are standards and guidelines issued by the United States National Institute of Standards and Technology (NIST) for federal government computer systems. FIPS can be enabled for SOAP Gateway.

Related tasks:

"Creating the server keystore for SOAP Gateway and exporting the public key as a certificate" on page 150
Create a server keystore for SOAP Gateway and export the public key as a server certificate that the SOAP Gateway client can use to verify that the server is trusted.

"Creating the server truststore for SOAP Gateway" on page 152
Create a truststore for SOAP Gateway to store the HTTPS client certificates, or the SSL server certificate (from IMS Connect).

"Exporting the certificate from IMS Connect" on page 152
Use the RACDCERT command to export the certificate to a data set.

"Example: Configuring the client authentication and basic authentication security scheme" on page 192
This example demonstrates how to create self-signed certificates to configure client authentication and basic authentication when the web service is hosted on an Apache Tomcat server on Windows. The actual location of the key management utility might be different based on your server environment.

Configuring IMS Connect for SOAP Gateway

You must configure IMS Connect to allow SOAP Gateway to access IMS transactions.

Prerequisite: Ensure that IMS Connect and IMS Open Transaction Manager Access (OTMA) are properly configured and enabled. Ensure that IMS Connect and OTMA are set up with the correct level of security.

To configure IMS Connect for SOAP Gateway:

1. Configure IMS Connect with user exit routine HWSSOAP1.

   The HWSSOAP1 exit routine is an IMS Connect exit routine that manages the translation of message headers on input and output messages and provides a point of control to modify, route, and check security for messages from and to SOAP Gateway.

   Ensure that the HWSSOAP1 exit routine is specified within the IMS Connect configuration member EXIT keyword within the TCPIP statement, as shown in the following sample code for the IMS Connect configuration member HWSCFGxx:

   ```
   HWS=(ID=HWS8,RACF=Y,XIBAREA=20)
   TCPIP=(HOSTNAME=MVSTCPIP,RACFID=RACFID,
   PORTID=(9999,LOCAL),MAXSOC=2000,TIMEOUT=8800,
   EXIT=(HWSSMPL1,HWSSOAP1))
   ADAPTER=(XML=Y)
   ```

2. Configure IMS Connect with user exit routine HWSSMPL1.

   The HWSSMPL1 exit routine internally manages SOAP Gateway ping support in IMS Connect. It is provided in the IMS.SDFSSMPL data set. Add the HWSSMPL1 exit routine to the EXIT parameter of the TCPIP statement of the HWSCFGxxx member in the IMS.PROCLIB data set. For more information about the exit routine, see IMS Connect user message exit routines in *IMS Version 13 Exit Routines*.

3. For NIST SP800-131a, configure IMS Connect to enable TLSv1.2 support.

   a. Turn on TLS v1.2 support in the IMS Connect SSL configuration member by setting the GSK_PROTOCOL_TLSV1_2 variable to GSK_PROTOCOL_TLSV1_2_ON:

      ```
      GSK_PROTOCOL_TLSV1_2=GSK_PROTOCOL_TLSV1_2_ON
      ```

   b. Specify the cipher suite to enable. For example:

      ```
      GSK_V3_CIPHER_SPECS=3C0906030201
      ```

      In this example, the first two characters, 3C, indicate 128-bit AES encryption with SHA-256 message authentication and RSA key exchange. For more information about cipher suite definitions for TLS v1.2, see Cipher Suite Definitions in z/OS Cryptographic Services System SSL Programming information.

   For more information about IMS Connect SSL setup and related variables, see the SSL initialization topic in *IMS Version 13 Communications and Connections*.

4. After changing the IMS Connect configuration member, restart IMS connect.

5. Optional: Configure SSL communications. You can use either z/OS Communications Server Application Transparent Transport Layer Security feature (AT-TLS) or IMS Connect to manage your SSL communications. AT-TLS is recommended because it provides greater flexibility with respect to the use of ports in addition to simplifying the IMS Connect security implementation.

   - For more information about configuring IMS Connect for SSL, see IMS Connect SSL connections in *IMS Version 13 Communications and Connections*.

- For more information about configuring z/OS AT-TLS, see z/OS: Communications Server IP Configuration Guide.

6. To use the XML conversion function in IMS Connect, you must configure the IMS Connect XML adapter function.

Related tasks:

Installing SOAP Gateway on z/OS (Installation roadmap)
Use the provided installation roadmap to guide your installation. Installation must be completed by a system programmer who is familiar with installation tasks on a z/OS platform.

Installing SOAP Gateway on distributed platforms (Installation roadmap)
Use the provided installation roadmap to guide your installation. Install SOAP Gateway on distributed platforms by using the IBM Installation Manager.

"Diagnosing Installation Verification Program errors" on page 349
Running the IVP for SOAP Gateway is usually trouble free. However, if you do experience an error, this topic lists the possible errors and recommended solutions for these problems.

Related reference:

 HWSCFGxx configuration member (IMS Version 13)
For more information about the HWSCFGxx configuration member, see IMS Version 13 system definition information.

Configuring XML conversion support for IMS Connect clients (IMS V13 System Definition)
For more information about configuring XML conversion support in IMS Connect, see the IMS V13 System Definition information.

Verifying the installation of SOAP Gateway

To verify the installation of SOAP Gateway, use the SOAP Gateway Installation Verification Program (IVP).

Prerequisites:

- SOAP Gateway must be installed by following the installation roadmap for your platform:
  - Installation roadmap for the z/OS platform
  - Installation roadmap for the distributed platforms
- The default port number for the SOAP Gateway server is 8080. If you did not change the default setting, check to see if port 8080 is used by other processes or programs. If you need to change this port number, see "Configuring the SOAP Gateway server port numbers" on page 95 for further information.

  Tip: To check if the port is reserved, use the UNIX System Services `netstat -o` command to check if the port is reserved.

The SOAP Gateway IVP is a web service that is installed with SOAP Gateway. The IVP verifies the installation when you successfully access the target IMS environment. The IVP does not require a host IMS application.

To verify the installation:

1. Update the connection bundle properties for use by the IVP to connect to your target IMS. The connection bundle contains information about the IMS host system, port number, and data store in order for SOAP Gateway to connect to

the backend IMS system. The SOAP Gateway management utility provides a command to update the connection bundle.

a. Change the directory to the *install_dir*/imsserver/deploy directory where the SOAP Gateway management utility is to run the SOAP Gateway management utility.

   z/OS For example, on z/OS:

   ```
   cd -PathPrefix-/usr/lpp/ims/imses/V3R1/soap_gw/deploy
   ```

b. Update the connection bundle called imssoapivp. The connection bundle imssoapivp is provided with the SOAP Gateway IVP, but you must update it for the correct host name, port number and data store in your environment. Use the SOAP Gateway management utility iogmgmt -conn -u command.

   The following example demonstrates how to update (-u) the named connection bundle (-n) with the new host name (-h), port number (-p), and data store (-d).

   z/OS Linux

   ```
   ./iogmgmt -conn -u -n imssoapivp -h YourHost -p YourIMSConnectPort -d YourDataStore
   ```

   Windows

   ```
   iogmgmt -conn -u -n imssoapivp -h YourHost -p YourIMSConnectPort -d YourDataStore
   ```

2. Start the SOAP Gateway server.

   - z/OS On z/OS, use the z/OS START command:

     ```
     /START AEWIOGPR
     ```

   - Linux On Linux on System z, go to the *install_dir*/imsserver/deploy directory, and issue the following command:

     ```
     ./iogmgmt -start
     ```

   - Windows On Windows, from the Windows Start menu, click **Start** > **All Programs** > **IBM Enterprise Suite Version 3.1** > **SOAP Gateway** > **Start Server**.

   Wait until the IOG30001I message displays.

3. Run the IVP:

   a. From a web browser, enter a URL in the following format to invoke the SOAP Gateway IVP web client.

      ```
      http://hostname:port/imssoap/imssoapivp.html
      ```

      The *hostname* is the host name of the target system and *port* is the port number where SOAP Gateway is running. The default port number is 8080.

      Tip: If you invoke the IVP from the same workstation where SOAP Gateway is installed, you can use the default port number. The URL is `http://localhost:8080/imssoap/imssoapivp.html`.

   b. Click **Submit**. The IVP is run successfully if you see one of the following messages:

      - `DFS058I hh:mm:ss START COMMAND COMPLETED`
      - `DFS1292E SECURITY VIOLATION`

      The objective of the IVP is to receive a message from IMS in reply to the /STA OTMA command. One of these messages is returned, depending on the level of security checking in place for your installation, and whether the user name, password, or group name is permitted to issue the /STA command. Either message indicates successful execution of the IVP, because IMS issues the DFS message. The IVP does not run an IMS application

program, nor verify that IMS transactions can run successfully. It only verifies that the path to IMS is available.

If you are upgrading from IMS Enterprise Suite Version 2.2 SOAP Gateway, see "Migrating from IMS Enterprise Suite Version 2.2 SOAP Gateway" on page 106.

If you are upgrading from IMS Enterprise Suite Version 2.1 SOAP Gateway, see "Migrating from IMS Enterprise Suite Version 2.1 SOAP Gateway."

Related tasks:

Installing SOAP Gateway on z/OS (Installation roadmap)
Use the provided installation roadmap to guide your installation. Installation must be completed by a system programmer who is familiar with installation tasks on a z/OS platform.

Installing SOAP Gateway on distributed platforms (Installation roadmap)
Use the provided installation roadmap to guide your installation. Install SOAP Gateway on distributed platforms by using the IBM Installation Manager.

"Diagnosing Installation Verification Program errors" on page 349
Running the IVP for SOAP Gateway is usually trouble free. However, if you do experience an error, this topic lists the possible errors and recommended solutions for these problems.

Migrating from IMS Enterprise Suite Version 2.1 SOAP Gateway

After you install IMS Enterprise Suite Version 3.1 SOAP Gateway, migrate your web service files and server properties by using the SOAP Gateway management utility iogmgmt -migrate command.

Prerequisites:

- Install SOAP Gateway as described in "Installing SOAP Gateway on z/OS" on page 60 or "Installing SOAP Gateway on distributed platforms" on page 87.
- You must upgrade to the latest maintenance release before you can upgrade to V3.1. For the latest maintenance release, see the IMS Enterprise Suite V2.1 release notes.

To migrate your existing web services and server properties:

1. Use the SOAP Gateway management utility iogmgmt -migrate *path_to_source_installation* command in the new installation (target) to migrate all web services and server properties from the previous installation (source).
   a. Go to the directory where the V3.1 SOAP Gateway management utility is at: *install_dir*/imsserver/deploy.
   b. Issue the following command:

      z/OS Linux

      ```
      ./iogmgmt -migrate path_to_2.1_install_dir
      ```

      Windows

      ```
      iogmgmt -migrate path_to_2.1_install_dir
      ```

      The migration utility handles the following migration tasks:
      - The correlator entries are migrated over and checked to ensure that the files are up to date.
      - The connection bundle files are copied to the new installation.
      - The following server configuration information is copied over to the new installation:

- Server properties that are set through the SOAP Gateway management utility.

  Important: The iogmgmt -migrate command does not validate the existing values. If an invalid value was previously manually added, the problem would not surface until during run time.
- The `wsjaas.conf` file and the server policy and bindings files for web services security are copied over.
- The WS-Security binding and policy files are migrated only if the source binding/policy files are different from the target binding and policy files.
- The following properties that are set in the `log4j.properties` file are migrated to the `native.env.properties` file:
  - log4j.appender.CONSOLE.Threshold
  - log4j.appender.LOGFILE.Threshold
  - log4j.appender.LOGFILE.encoding

- For web services migration, the tool parses through the correlator entries on the source (V2.2) server to identify all the existing web services and their corresponding WSDL files:
  - For the provider scenario, for each WSDL file found in the correlator entries:
    - If the WSDL file and the XSD files it references do not exist in the `wsdl/` directory, but the web service .aar file is found in the `WEB-INF/services/` directory, the web service .aar file is copied over to the target server (V3.1).
    - If the WSDL file and the XSD files it references exist in the `wsdl/` directory, a web service .aar file is generated on the target server.
  - For the consumer and business event scenarios, the WSDL and XML files must exist on the source server. These files are copied over to the target server.

  Log file location and trace level setting are not migrated. A migration report (`migration.log`) is saved in the designated V3.1 SOAP Gateway log file directory. The report contains the following information:
  - Server configuration information that is migrated and configured.
  - Web services security-related files that are migrated.
  - Connection bundle information that is migrated.
  - Correlator files that are migrated and updated to the new schema.
  - Web services that are deployed.

2. Check the migration report.

   For any reported issues, correct the issues, and rerun the iogmgmt -migrate command until all errors are resolved.

   Important: You can run the migration tool as many times as necessary. For steps to correct correlator-related issues and rerunning the migration, see "Migrating correlator files to schema version 3.0" on page 302.

   For web services that fail to be deployed, after the issues are corrected, manually deploy the services by using the iogmgmt -deploy command. For example:

   z/OS Linux

   ```
   ./iogmgmt -deploy -w /path/to/yourWSDLFile.wsdl -r /path/to/yourCorrelatorFile.xml
   ```

   Windows

```
iogmgmt -deploy -w /path/to/yourWSDLFile.wsdl -r /path/to/yourCorrelatorFile.xml
```

3. Adjust your server properties if necessary. Customize your log file location and trace level setting.
4. Start the SOAP Gateway server. If the server is already running, you must restart the server for server property changes to take effect.

Related tasks:

Installing SOAP Gateway on distributed platforms (Installation roadmap)
Use the provided installation roadmap to guide your installation. Install SOAP Gateway on distributed platforms by using the IBM Installation Manager.

"Migrating correlator files to schema version 3.0" on page 302
IMS Enterprise Suite Version 3.1 SOAP Gateway requires correlator schema version 3.0. To migrate an existing correlator file from older versions to version 3.0, use the SOAP Gateway management utility iogmgmt -migrate correlator command.

"Configuring the SOAP Gateway log file location" on page 95
By default, the log files are written to the *install_dir*/imsbase/logs directory. To change the log file location, use the SOAP Gateway management utility.

"Setting the trace level for SOAP Gateway" on page 304
You can turn on internal tracing for SOAP Gateway to help diagnose problems. The trace level can be changed to control the amount of logging.

Related reference:

"-migrate: Migrate and upgrade SOAP Gateway" on page 448
The -migrate command upgrades SOAP Gateway artifacts and settings to the latest version and generates a migration log.

"-deploy: Deploy a web service or callout application" on page 444
The -deploy command deploys a web service, callout application, or business event application to the active configuration of the SOAP Gateway server.

Migrating from IMS Enterprise Suite Version 2.2 SOAP Gateway

After you install IMS Enterprise Suite Version 3.1 SOAP Gateway, migrate your web service files and server properties by using the SOAP Gateway management utility iogmgmt -migrate command.

Prerequisites:

- Install SOAP Gateway as described in "Installing SOAP Gateway on z/OS" on page 60 or "Installing SOAP Gateway on distributed platforms" on page 87.
- You must upgrade to the latest maintenance release before you can upgrade to V3.1. For the latest maintenance release, see the IMS Enterprise Suite V2.2 release notes.

To migrate your existing web services and server properties:

1. Use the SOAP Gateway management utility iogmgmt -migrate *path_to_source_installation* command in the new installation (target) to migrate all web services and server properties from the previous installation (source).
   a. Go to the directory where the V3.1 SOAP Gateway management utility is at: *install_dir*/imsserver/deploy.
   b. Issue the following command:

      z/OS Linux

      ```
      ./iogmgmt -migrate path_to_2.2_install_dir/imsserver
      ```

      Windows

      ```
      iogmgmt -migrate path_to_2.2_install_dir/imsserver
      ```

The migration utility handles the following migration tasks:

- The correlator entries are migrated over and checked to ensure that the files are up to date.
- The connection bundle files are copied to the new installation.
- The following server configuration information is copied over to the new installation:
  - Server properties that are set through the SOAP Gateway management utility.

    Important: The iogmgmt -migrate command does not validate the existing values. If an invalid value was previously manually added, the problem would not surface until during run time.
  - The `wsjaas.conf` file and the server policy and bindings files for web services security are copied over.
- For web services migration, the tool parses through the correlator entries on the source (V2.2) server to identify all the existing web services and their corresponding WSDL files:
  - For the provider scenario, for each WSDL file found in the correlator entries:
    - If the WSDL file and the XSD files it references do not exist in the `wsdl/` directory, but the web service .aar file is found in the `WEB-INF/services/` directory, the web service .aar file is copied over to the target server (V3.1).
    - If the WSDL file and the XSD files it references exist in the `wsdl/` directory, a web service .aar file is generated on the target server.
  - For the consumer and business event scenarios, the WSDL and XML files must exist on the source server. These files are copied over to the target server.

Log file location and trace level setting are not migrated. A migration report (`migration.log`) is saved in the designated V3.1 SOAP Gateway log file directory. The report contains the following information:

- Server configuration information that is migrated and configured.
- Web services security-related files that are migrated.
- Connection bundle information that is migrated.
- Correlator files that are migrated and updated to the new schema.
- Web services that are deployed.

2. Check the migration report.

   For any reported issues, correct the issues, and rerun the iogmgmt -migrate command until all errors are resolved.

   Important: You can run the migration tool as many times as necessary. For steps to correct correlator-related issues and rerunning the migration, see "Migrating correlator files to schema version 3.0" on page 302.

   For web services that fail to be deployed, after the issues are corrected, manually deploy the services by using the iogmgmt -deploy command. For example:

   z/OS Linux

   ```
   ./iogmgmt -deploy -w /path/to/yourWSDLFile.wsdl -r /path/to/yourCorrelatorFile.xml
   ```

   Windows

   ```
   iogmgmt -deploy -w /path/to/yourWSDLFile.wsdl -r /path/to/yourCorrelatorFile.xml
   ```

3. Adjust your server properties if necessary. Customize your log file location and trace level setting.
4. Start the SOAP Gateway server. If the server is already running, you must restart the server for the changes to take effect.

Related tasks:

Installing SOAP Gateway on distributed platforms (Installation roadmap)
Use the provided installation roadmap to guide your installation. Install SOAP Gateway on distributed platforms by using the IBM Installation Manager.

"Migrating correlator files to schema version 3.0" on page 302
IMS Enterprise Suite Version 3.1 SOAP Gateway requires correlator schema version 3.0. To migrate an existing correlator file from older versions to version 3.0, use the SOAP Gateway management utility iogmgmt -migrate correlator command.

"Configuring the SOAP Gateway log file location" on page 95
By default, the log files are written to the *install_dir*/imsbase/logs directory. To change the log file location, use the SOAP Gateway management utility.

"Setting the trace level for SOAP Gateway" on page 304
You can turn on internal tracing for SOAP Gateway to help diagnose problems. The trace level can be changed to control the amount of logging.

Related reference:

"-migrate: Migrate and upgrade SOAP Gateway" on page 448
The -migrate command upgrades SOAP Gateway artifacts and settings to the latest version and generates a migration log.

"-deploy: Deploy a web service or callout application" on page 444
The -deploy command deploys a web service, callout application, or business event application to the active configuration of the SOAP Gateway server.

Cloning the SOAP Gateway server

Use the SOAP Gateway management utility iogmgmt -migrate command to propagate web services and server properties to another instance of the server on the same system or direct access storage device (DASD).

Prerequisites:

1. The master copy of the SOAP Gateway server must be properly installed and configured.
2. All web services are properly migrated or deployed on the master copy of the server.
3. For each clone instance of the server, SOAP Gateway must be properly installed. Installation steps for each additional instance of SOAP Gateway are the same as those for the master copy. However, if you are using the IBM Installation Manager silent mode for installation, each instance must be installed with a different profile name for each of the SOAP Gateway installation component. For more information, see "Installing SOAP Gateway in silent mode" on page 91.

Important: The iogmgmt -migrate command is designed to facilitate cloning of web services and server properties for initial server setup. It is not designed to support subsequent deployment of additional web services to all instances of the SOAP Gateway server.

To clone the web services and server properties from a master instance:

1. On the clone instance (the target), go to the *install_dir*/imsserver/deploy directory.

2. Issue the SOAP Gateway management utility iogmgmt -migrate *path_to_source_installation* command to migrate all web services and server properties from the master installation. For the master installation directory, specify the path to where the **imsserver** component is located:

   ```
   ./iogmgmt -migrate /opt/IBM/IMS_Enterprise_Suite_V3.3/imsserver/
   ```

3. For any reported issues, correct the issues, and rerun the iogmgmt -migrate *path_to_source_installation* command until all errors are resolved.

   Tip: You can run the migration tool as many times as necessary.

4. Manually copy any security files, such as custom authentication classes files for WS-Security, to the new installation.
5. Adjust your server properties, such as the server listening port and shutdown port. Both port numbers must be unique for each instance of the server. To specify the ports, issue the following SOAP Gateway management utility command:

   ```
   ./iogmgmt -prop -u -p server_port_number -d shutdown_port_number
   ```

   Each instance of the SOAP Gateway server must have a unique shutdown port. The shutdown port must be available when the server is started.

6. If client authentication or server authentication is set to true in the master server copy, after you clone the server properties, you must set the client authentication or server authentication to false first. Copy the keystore or truststore information to the cloned server instance, and then set the **-clientauth** or **-serverauth** properties to true. The HTTPS port must also be unique. For example, to set client authentication to false:

   ```
   ./iogmgmt -prop -u -clientauth false
   ```

   To reset the client authentication to true:

   ```
   ./iogmgmt -prop -u -clientauth true -s https_port -k keystore_loc
   -w keystore_pwd -t truststore_loc -a truststore_pwd
   ```

7. Start the SOAP Gateway server. If the server is already running, you must restart the server for the changes to take effect.

Related tasks:

Installation roadmap for z/OS
Use the provided installation roadmap to guide your installation. Installation must be completed by a system programmer who is familiar with installation tasks on a z/OS platform.

Installation roadmap for distributed platforms
Use the provided installation roadmap to guide your installation. Install SOAP Gateway on distributed platforms by using the IBM Installation Manager.

Related reference:

"-migrate: Migrate and upgrade SOAP Gateway" on page 448
The -migrate command upgrades SOAP Gateway artifacts and settings to the latest version and generates a migration log.

"-prop: Set SOAP Gateway properties" on page 450
Use the -prop command to modify the SOAP Gateway server properties.

Installing multiple SOAP Gateway server instances that share one JVM

Multiple SOAP Gateway server instances can share a single instance of the Java Virtual Machine (JVM). This option reduces the amount of storage required for each additional server instance and improves the serviceability because a SDK service update needs to be installed only once to apply to all SOAP Gateway servers sharing that SDK.

Every server instance must have access to the directory where the IBM SDK is installed.

To install multiple SOAP Gateway server instances that share one JVM:

1. Install the first SOAP Gateway server. See Chapter 3, "Installing and configuring SOAP Gateway," on page 41.
2. Install the subsequent instances of the SOAP Gateway server, but exclude the IBM Java SDK from installation. .
   - Linux Windows On the distributed platform:
     - If you are using the wizard mode, clear the checkbox for **IBM Java SDK V*x.x.x.x*** when you choose the packages to install.
     - If you are using the silent mode, comment out the Java SDK offering in the `IMSES31SOAPResponseFile.xml` file:

       ```
       <!--Comment the next line if you do not want to install the IBM Java SDK V7.0>
       <!-- <offering id='com.ibm.ims.sgw.java.sdk.v7'
       profile='IBM Java SDK V7.0' features='javasdk' installFixes='none'/> -->
       ```

       See "Installing SOAP Gateway in silent mode" on page 91 for more information.
3. Issue the command `iogmgmt -prop -u -java -h` *SDK_path*. You must issue this command to the SOAP Gateway in the *`install_dir`*`/imsserver/deploy` directory of the new server instance. Each SOAP Gateway server instance has an independent SOAP Gateway management utility.

   This command binds the new SOAP Gateway server instance to the existing IBM SDK. You must issue this command before you can issue other SOAP Gateway management utility commands.

   You can view the current Java home directory with the following command:

   ```
   iogmgmt -view -java -h
   ```
4. Use the SOAP Gateway management utility to configure the new server instance.

Related tasks:

Installing SOAP Gateway on z/OS (Installation roadmap)
Use the provided installation roadmap to guide your installation. Installation must be completed by a system programmer who is familiar with installation tasks on a z/OS platform.

Installing SOAP Gateway on distributed platforms (Installation roadmap)
Use the provided installation roadmap to guide your installation. Install SOAP Gateway on distributed platforms by using the IBM Installation Manager.

"Configuring SOAP Gateway" on page 94
For z/OS, you must specify the IBM Java SDK location before you can start the SOAP Gateway server. Optionally, for all supported platforms, you can change the default log file location, server listening port, or shutdown port. For z/OS, you can enable a System z Application Assist Processor (zAAP).

Related reference:

"-prop: Set SOAP Gateway properties" on page 450
Use the -prop command to modify the SOAP Gateway server properties.

Verifying the setup for the consumer (callout) usage scenario

To verify that SOAP Gateway is set up property to pull IMS callout requests from IMS OTMA hold queues, create a connection bundle in SOAP Gateway with the correct connection information, and run the provided IMS Installation Verification Program (IVP) jobs to issue the callout requests from the provided IMS program.

An IMSSOAPCalloutIVPService web service is shipped with SOAP Gateway that receives and processes the callout requests from the IMS IVP jobs. When the SOAP Gateway starts, this web service is automatically deployed.

Prerequisites:

1. You must complete the configuration steps in "Configuring IMS Connect for SOAP Gateway" on page 101.
2. For IMS Version 12, you must apply the callout IVP sample APAR PM31536. For IMS Version 11, you must apply the callout IVP sample APAR PM31226.

   The callout IVP sample provides the IVP jobs and tasks in the S series to set up the environment. IV_S230J and IV_S231J are jobs that run the IMS BMP applications by using the IMS DL/I testing program, DFSDDLT0, to issue the callout messages.

   - Job IV_S230J, when run, issues an asynchronous callout request to a predefined tpipe (IVPPIPE5).
   - Job IV_S231J, when run, issues a synchronous callout request to a predefined tpipe (IVPPIPE6).

   The callout requests are placed in the resume tpipes (hold queues) until they are pulled by SOAP Gateway. Introduction information for the S series is in IV_S001T. Task IV_229T contains information about the COBOL converter samples that you must compile and link-edit into the IMS.SDFSRESL data set before submitting jobs in IV_S230J and IV_S231J.

Table 22. The callout IVP samples on the IMS host system

Type of callout requests	Job	Tpipe	Part name for the COBOL converter source in SDFSSMPL
Asynchronous	IV_S230J	IVPPIPE5	DFSACCBL
Synchronous	IV_S231J	IVPPIPE6	DFSSCCBL

   To compile and link-edit the converters into a data set concatenated with IMS Connect STEPLIB:

   - The converter must have an alias that is linked with the converter code, using the same name as the converter, with an X suffix. If the converter name is eight-character long, the last character for the alias must be changed to an X.
   - The converter and the load module name must end with the letter D.

   For example, by default the program ID for the provided asynchronous COBOL converter source file is IOGIVPAD. Your link-edit JCL would contain ENTRY, ALIAS, and NAME statements as follows:

```
//LKED.SYSIN DD *
INCLUDE RESLIB(CBLTDLI)
INCLUDE RESLIB(DFSLI000)
ENTRY IOGIVPAD
ALIAS IOGIVPAX
NAME IOGIVPAD(R)
```

3. Verify that the static OTMA destination descriptors for IVPPIPE5 and IVPPIPE6 in DFSYDTx reflect the correct converter module names generated above. Changes require an IMS cold start, or use the following command to make effective:

   ```
   UPDATE OTMADESC NAME(IVPDTOR5) SET(TYPE(IMSCON)  +
   ADAPTER(HWSXMLA0) CONVRTR(IOGIVPAD))
   ```

4. If you have changed the default SOAP Gateway server HTTP port number, the callout IVP WSDL file, `IMSSOAPCalloutIVP.wsdl`, must be updated with the port number. Modify the `IMSSOAPCalloutIVP.wsdl` file in the server WSDL directory and restart the server.

To verify the proper setup for the consumer scenario, you need to update the provided connection bundle to instruct the SOAP Gateway server where to pull the callout request from the tpipe, start the SOAP Gateway server, and then run the two jobs, IV_S230J and IV_S231J, in IMS to issue the callout request.

Verifying the setup for the IMS asynchronous callout function

To verify that SOAP Gateway is set up property to pull IMS callout requests from IMS OTMA hold queues, update the provided connection bundle in SOAP Gateway, and run the provided IMS Installation Verification Program (IVP) jobs to issue an asynchronous callout request.

Prerequisites: See the prerequisites described in "Verifying the setup for the consumer (callout) usage scenario" on page 111.

To verify the proper setup for the IMS asynchronous callout function:

1. In SOAP Gateway, update the provided imssoapcalloutivp connection bundle to specify the IMS host system, port number, data store, and the tpipe name.
   a. Navigate to the *install_dir*`/imsserver/deploy` directory.
   b. Issue the following SOAP Gateway management utility command:

      ```
      iogmgmt -conn -u -n imssoapcalloutivp -h YourHostName -p YourPortNumber
      -d YourDataStore -i  IVPPIPE5
      ```

      - The connection bundle must be named `imssoapcalloutivp`. This connection bundle name is already specified in the callout web service that is automatically deployed when the SOAP Gateway server is started.
      - *YourHostName* is the name or IP address of the host system where IMS Connect is running.
      - *YourPortNumber* is the port number to access IMS Connect.
      - *YourDataStore* is the data store name.
      - The tpipe name must be set to IVPPIPE5. This tpipe name is specified in the OTMA destination description that is provided in the IMS callout IVP sample.
2. Start the SOAP Gateway server.
   - z/OS On z/OS, use the z/OS START command:

     ```
     /S AEWIOGPR
     ```

- Windows Linux On other platforms, issue the following SOAP Gateway management utility command:

  ```
  iogmgmt -start
  ```

  Important: If the server is already started, you must restart the server for the changes to take effect.

  Wait until the `SOAP Gateway server is now up and running` message displays.

3. In IMS, run the IMS callout application by running job IV_S230J. Job IV_S230J runs the JCL that uses the DFSDDLT0 test program to issue an ISRT ALTPCB call to send a message to tpipe IVPPIPE5.

   The IVP callout application input is:

   ```
   THIS IS A ONE-WAY ASYNCHRONOUS CALLOUT MESSAGE FROM IMS
   ```

   Because this request is a one-way asynchronous callout, there is no response from the web service.

From the SOAP Gateway log, if you see the following message, you have successfully issued an asynchronous callout request from IMS, through SOAP Gateway, to an external web service.

```
INFO: Callout Job (IVPPIPE5:0): No response was received from the external
web service because it is an asynchronous notification (one-way) service.
```

Tips:

- By default, the SOAP Gateway log file `imssoap.log` is stored in *install_dir*`/imsbase/logs`.
- In this sample, the web service is hosted on the same SOAP Gateway server.

Related concepts:

Callout requests for external services or data (IMS Version 13)
For more information about the callout function, see IMS Version 13 Application Programming information.

Samples for the callout function (IMS Version 13)
For more information about the IVP samples for the callout function, see IMS Version 13 Installation information.

Related reference:

"-conn: Create, update, or delete a connection bundle" on page 435
Use the -conn command to create, update, or delete a connection bundle.

Steps Sx for callout samples (IMS Version 13)
For more information about the S series jobs and tasks in the IVP for the callout samples, see IMS Version 13 Installation information.

Verifying the setup for the IMS synchronous callout function

To verify that SOAP Gateway is set up property to pull IMS callout requests from IMS OTMA hold queues, update the provided connection bundle in SOAP Gateway, and run the provided IMS Installation Verification Program (IVP) jobs to issue a synchronous callout request.

Prerequisites: See the prerequisites described in "Verifying the setup for the consumer (callout) usage scenario" on page 111.

To verify the proper setup for the IMS synchronous callout function:

1. In SOAP Gateway, update the imssoapcalloutivp connection bundle to specify the IMS host system, port number, and data store in your environment.
   a. Navigate to the *install_dir*/imsserver/deploy directory.
   b. Issue the following SOAP Gateway management utility command:

      ```
      iogmgmt -conn -u -n imssoapcalloutivp -h YourHostName -p YourPortNumber
       -d YourDataStore -i  IVPPIPE6
      ```

      - The connection bundle named imssoapcalloutivp is provided with the SOAP Gateway installation. This connection bundle name is already specified in the callout web service that is automatically deployed when the SOAP Gateway server starts.
      - *YourHostName* is the name or IP address of the host system where IMS Connect is running.
      - *YourPortNumber* is the port number to access IMS Connect.
      - *YourDataStore* is the data store name.
      - The tpipe name must be set to IVPPIPE6. This tpipe name is specified in the OTMA destination description that is provided in the IMS callout IVP sample.
2. Start the SOAP Gateway server.
   - z/OS On z/OS, use the z/OS START command:

     ```
     /S AEWIOGPR
     ```

   - Windows Linux On other platforms, issue the following SOAP Gateway management utility command:

     ```
     iogmgmt -start
     ```

   Important: If the server is already started, you must restart the server for the changes to take effect.

   Wait until the `IMS Enterprise Suite SOAP Gateway server is now up and running` message displays.
3. In IMS, run the IMS callout application by running job IV_S231J. Job IV_S231J runs the JCL that uses the DFSDDLT0 test program to issue a DL/I ICAL call to send a message to tpipe IVPPIPE6. The callout request looks as follows:

   ```
   ICAL  SENDRECV IVPDTOR6 001000 00028 00030
   ```

   IVPDTOR6 is the OTMA destination descriptor name that is also provided by the IMS callout IVP sample.

   The IVP callout application input is:

   ```
   THIS IS A SYNCHRONOUS CALLOUT MESSAGE SENT FROM IMS
   ```

If you receive the following response message in the SOAP Gateway log, you have successfully issued a synchronous callout request from an IMS application, through SOAP Gateway, to an external web service. The SOAP Gateway imssoapcalloutivp web service returns:

```
HELLO FROM SOAP GATEWAY
```

Tips:

- By default, the SOAP Gateway log file `imssoap.log` is stored in *install_dir*/imsbase/logs.
- In this sample, the imssoapcalloutivp web service is hosted on the same SOAP Gateway server.

.

Related concepts:

Callout requests for external services or data (IMS Version 13)
For more information about the callout function, see IMS Version 13 Application Programming information.

Samples for the callout function (IMS Version 13)
For more information about the IVP samples for the callout function, see IMS Version 13 Installation information.

Related reference:

"-conn: Create, update, or delete a connection bundle" on page 435
Use the -conn command to create, update, or delete a connection bundle.

Steps Sx for callout samples (IMS Version 13)
For more information about the S series jobs and tasks in the IVP for the callout samples, see IMS Version 13 Installation information.

Applying maintenance services on z/OS

Maintenance services on z/OS are made available as PTFs for APARs. After the PTFs are applied, modify the AEWTSINS sample job to point to the new repository.

Check the APAR ++HOLD card for steps to apply the service or potential migration steps before you upgrade.

In the general, the steps involve modifying the AEWTSINS sample job to point to the repository file that is provided in the maintenance service to install the updated component or components.

1. Set the @package ID name@ to each of the components that is changed in the PTFs. For example, if the IMSSERVER component is changed, set @package ID name@ variable to `com.ibm.ims.sgw.imsserver.v31` and run the job. If all three components are changed, repeat the step for each of the changed component by changing the @package ID name@ variable to `com.ibm.ims.sgw.imsbase.v31` and then `com.ibm.ims.sgw.imssoap.v31`.
2. Change the installation directory to the directory that contains the IMSSERVER component that you are updating. For example:

   ```
   -installationDirectory
   /usr/lpp/ims/imses/V3R1/soap_gateway/imsserver +
   ```
3. Change the repository name to point to the new repository file. For example,

   ```
   -repositories /usr/lpp/InstallationManagerRepository/JAHF311/ +
   IMSES_SOAPGateway_zOS_V3.1.0.4.zip
   ```
4. Submit the AEWTSINS job.

You would receive a message that indicates the update was successful. The following message indicates that the IMSSERVER component was updated successfully.

```
Modified com.ibm.ims.sgw.imsserver.v31 in the
/usr/lpp/ims/imses/V3R1/soap_gateway/imsserver directory.
```

If you change the Java installation location with updates of the IBM Java SDK, ensure that you specify the new Java location, and if FIPS is required, re-enable the FIPS settings. For more information, see "Specifying the Java SDK location" on page 95 and "Configuring compliance for FIPS 140-2 and NIST SP800-131a" on page 97.

Applying maintenance services on distributed platforms

Maintenance services can be downloaded from the IMS Enterprise Suite download site. Use either the **Upgrade** function in IBM Installation Manager or the silent installation approach to install the service.

Go to the IMS Enterprise Suite download page to download the SOAP Gateway installation repository file. Installation instructions are provided on the download site, with information on obtaining the latest Java from IBM.

Store the downloaded repository file in a location that is specified in IBM Installation Manager as the repository. Then, click **Upgrade** and follow the prompt in the user interface to upgrade.

If you use the silent installation approach, take the following steps.

1. Open `IMSES31SOAPConfig.dtd` in a text editor.
2. Specify the SOAP Gateway repository by modifying this line of code:

   ```
   <!-- path to the SOAP Gateway repository-->
   <!ENTITY repolocation "path to the SOAP Gateway Repository here">
   ```

   For example:

   ```
   <!-- path to the SOAP Gateway repository-->
   <!ENTITY repolocation "C:\IBM\IMSES31SOAP\soap_gateway_repository_filename.zip">
   ```
3. Customize the following lines to specify the absolute path to the directory to install the SOAP Gateway component that is being updated in the service: For example, in V3.1.0.1, the IMSSERVER component is updated, so the imsserverpath entity must be specified.

   ```
   <!-- Paths to install each of the SOAP Gateway Components -->
   <!ENTITY imsserverpath "C:\Program Files\IBM\IMS Enterprise Suite V3.1\SOAP Gateway\imsserver">
   ```
4. Specify the absolute path to the directory where the IBM Java SDK is installed by customizing the following line:

   ```
   <!-- Path to install Java -->
   <!ENTITY javaPath "C:\Program Files\IBM\IMS Enterprise Suite V3.1\SOAP Gateway\java">
   ```
5. Save your changes.
6. Go to the `eclipse\tools\` directory in the Installation Manager.
7. Install SOAP Gateway:
   - Windows On Windows, enter the following command:

     ```
     imcl.exe input path_To_responseFile/IMSES31SOAPResponseFile.xml -acceptLicense
     ```
   - Linux On Linux on System z, enter the following command:

     ```
     ./imcl input path_To_responseFile/IMSES31SOAPResponseFile.xml -acceptLicense
     ```

You would receive a message that indicates the update was successful. The following message indicates that the IMSSERVER component was updated successfully.

```
Modified com.ibm.ims.sgw.imsserver.v31 in the C:\Program Files (x86)\IBM\
IMS Enterprise Suite V3.1\SOAP Gateway\imsserver directory.
```

If you change the Java installation location with updates of the IBM Java SDK, ensure that you specify the new Java location, and if FIPS is required, re-enable the FIPS settings. For more information, see “Specifying the Java SDK location” on page 95 and “Configuring compliance for FIPS 140-2 and NIST SP800-131a” on page 97.

Removing SOAP Gateway

To remove a SOAP Gateway installation, edit and submit the AEWTSUNI job.

- z/OS On z/OS, edit and submit the AEWTSUNI job. For more information, see the instruction that is provided in the sample job.

  Tips:
  - If the **imsserver** component is mounted as read-only, it must be unmounted first and remounted in read/write mode before you can submit the AEWTSUNI job.
  - You need the same permission and authority as during installation.
- Linux Windows On distributed platforms:
  - In wizard mode, use the **Uninstall** option in the Installation Manager and following the instructions in the wizard to uninstall SOAP Gateway.
  - In silent mode:
    1. Edit the `IMSES31SOAPResponseFile.xml` file and look for the `install` tag:

```
<install modify='false'>
<!-- Version numbers must correspond to version numbers from IM build offerings-->
<offering id='com.ibm.ims.sgw.imsserver.v31'  profile='&imsserverProfile;'
features='com.ibm.im.ims.sgw installFixes='none'/>
<offering id='com.ibm.ims.sgw.imsbase.v31'  profile='&imsbaseProfile;'
features='com.ibm.ims.sgw.imsbase' installFixes='none'/>
<offering id='com.ibm.ims.sgw.imssoap.v31'  profile='&imssoapProfile;'
features='com.ibm.ims.sgw.imssoap' installFixes='none'/>
<!--Comment the next line if you do not want to install the IBM java SDK V7.0-->
<offering id='com.ibm.ims.sgw.java.sdk.v31' profile='&javaProfile;'
features='javasdk' installFixes='none'/>
</install>
```

    2. Change the `install` tag to `uninstall`:

```
<uninstall modify='false'>
<!-- Version numbers must correspond to version numbers from IM build offerings-->
<offering id='com.ibm.ims.sgw.imsserver.v31'  profile='&imsserverProfile;'
features='com.ibm.ims.sgw.imsserver installFixes='none'/>
<offering id='com.ibm.ims.sgw.imsbase.v31'  profile='&imsbaseProfile;'
features='com.ibm.im.ims.sgw.imsbase' installFixes='none'/>
<offering id='com.ibm.ims.sgw.imssoap.v31'  profile='&imssoapProfile;'
features='com.ibm.ims.sgw.imssoap' installFixes='none'/>
<!--Comment the next line if you do not want to install the IBM Java SDK V7.0-->
<offering id='com.ibm.ims.sgw.java.sdk.v31' profile='&javaProfile;'
features='javasdk' installFixes='none'/>
</uninstall>
```

    3. Go to the `eclipse\tools\` directory in the Installation Manager.
    4. Uninstall SOAP Gateway by using the Installation Manager imcl command:
       - Windows On Windows: `imcl.exe input` *`path_To_responseFile/`*`IMSES31SOAPResponseFile.xml -acceptLicense`
       - Linux On Linux on System z: `./imcl input` *`path_To_responseFile/`*`IMSES31SOAPResponseFile.xml -acceptLicense`

The SOAP Gateway installation is removed.

Important:

- Log files and temporary files that are generated by SOAP Gateway run time in the `imsbase/` directory cannot be removed by the Installation Manager. These files must be manually removed.
- User files under the `imssoap/` directory are left untouched. If you plan on reinstalling SOAP Gateway in the same directory, all three directories (`imsserver/`, `imssoap/` and `imsbase/`) must be empty.

Chapter 4. Design and implementation by usage scenario

SOAP Gateway supports three usage scenarios: web service provider, web service consumer, and business event.

The following table describes, for each of the SOAP Gateway usage scenario, the corresponding development approach in Rational Developer for System z, the application languages supported, and whether multi-operation or multi-segment messages are supported.

Table 23. Supported usage scenarios, application languages, and the corresponding Rational Developer for System z development scenarios.

Usage scenario	Rational Developer for System z Enterprise Services Tools development scenario	Application language supported	Multi-operation support	Multi-segment support
IMS applications as web service providers	*Top-down*: Generate the IMS application, the correlator file, and the XML converter driver (batch processing mode only) from the web service WSDL file.	PL/I	Yes	Yes
	Bottom-up: Generate the web service WSDL, the correlator file, and the XML converter driver from an IMS application.	COBOL	No	Yes
		PL/I	No	No
IMS callout applications as web service consumers	*Top-down*: For synchronous callout only, generate a COBOL copybook, mapping session files, a correlator file, and the XML converter driver for one or more operations in a WSDL file.	COBOL	Yes	No
	Meet-in-middle: Generate web service files from the source (IMS callout application) and the target (web service) XML schema definitions; defining mapping between high-level-language data structures and the web service XML schema for request mapping and response mapping.	COBOL asynchronous	No	No
		COBOL synchronous	No	Yes
		PL/I asynchronous	No	No
		PL/I synchronous	No	Yes

Table 23. Supported usage scenarios, application languages, and the corresponding Rational Developer for System z development scenarios (continued).

Usage scenario	Rational Developer for System z Enterprise Services Tools development scenario	Application language supported	Multi-operation support	Multi-segment support
IMS applications emitting business events	When the target business event processing engine is WebSphere Business Events: • Use the bottom-up approach to generate the XML schema file (XSD file). • Use WebSphere Business Events Design Data to generate the WSDL file. • Use the meet-in-middle approach to generate the data mapping files, the correlator file, and the XML converter driver.	COBOL	No	No
	When the target business event processing engine is WebSphere Business Monitor: • Use the bottom-up approach to generate the XML schema file (XSD file). • Use the meet-in-middle approach to generate the data mapping files, the correlator file, and the XML converter driver.	COBOL	No	No

Web service provider scenario

SOAP Gateway enables IMS applications as web service providers that receive inbound requests from external client applications. This scenario is called the *web service provider* scenario.

The following figure shows the SOAP Gateway runtime environment when IMS applications are enabled as web service providers and receive inbound requests from client applications. The numbers in the figure correspond to the descriptions that follow.

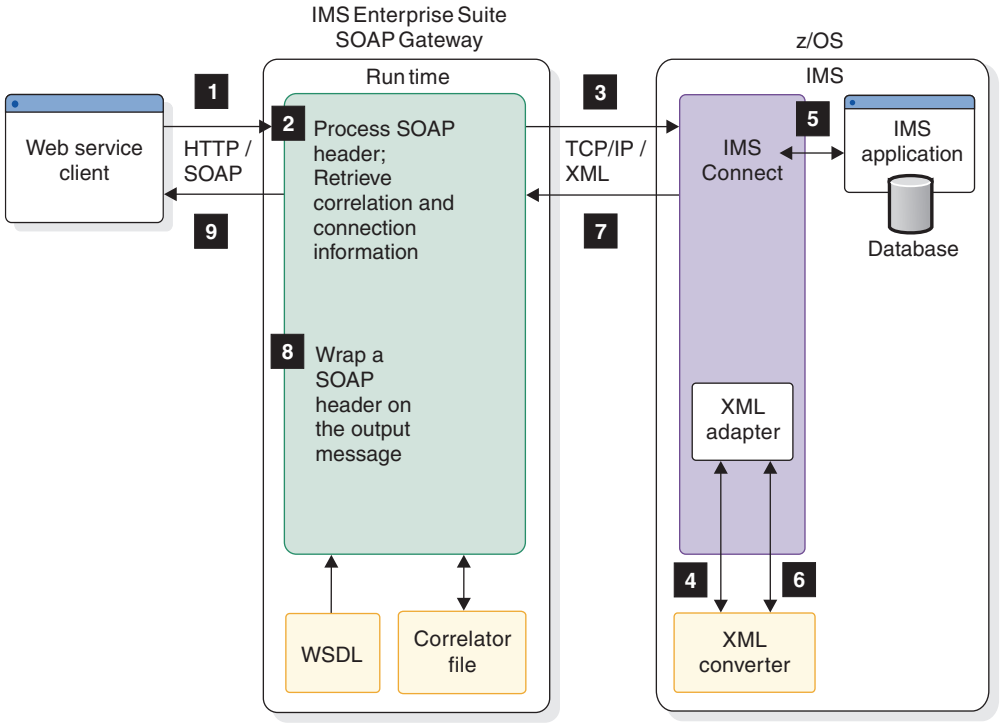


Figure 10. SOAP Gateway runtime environment for the IMS applications as web service providers scenario

1. The web service client application sends a SOAP message to SOAP Gateway that contains input to the IMS application in an XML format.
2. SOAP Gateway processes the SOAP header (XML) and retrieves the appropriate correlation and connection information for the input request.
3. SOAP Gateway sends the input XML data to IMS Connect by using TCP/IP after adding the appropriate IMS Connect header.
4. IMS Connect calls its XML adapter, which in turn calls the XML converter to transform the XML to IMS application format.
5. IMS Connect sends the message for further processing. From this point on, the processing is the same as a normal transaction flow. The transaction is executed and the output is queued.
6. IMS Connect calls the XML adapter to transform the IMS application format to XML.
7. IMS Connect sends the output XML message back to SOAP Gateway by using TCP/IP.
8. SOAP Gateway wraps a SOAP header on the output message.
9. SOAP Gateway sends the output message back to the client application.

If you handle the data transformation in your application without using the IMS Connect XML adapter, you do not need to invoke the XML adapter. In this case, the incoming XML message is sent directly to IMS Connect and then to the IMS application. The IMS application creates an XML output message that is sent to IMS Connect, then directly to SOAP Gateway, and lastly to the web service client.

Support for multi-segment messages

You can enable multi-segment IMS COBOL or PL/I applications as web service providers by identifying the layouts of the input and output messages by using IBM Rational Developer for System z.

Restriction: Multi-segment IMS applications are supported only for the following scenarios:

- When a PL/I application that is generated from a web service WSDL file using the top-down approach is enabled as a web service.
- When a COBOL application is enabled as a web service using the bottom-up approach.
- Synchronous callout applications (because the IMS Message Queue is bypassed).

For all other scenarios, the maximum size is 32 KB for a single segment.

Multi-segment IMS applications receive and send messages in multiple segments. Each segment in the message can have the same data structure or different data structures. The segment structure is referred to as the *language structure*.

For SOAP Gateway and IMS Connect to handle the multi-segment messages, both the input and output message structures must be defined and known ahead of time. You then use the Rational Developer for System z to identify the language structures in your COBOL source file that comprise the input to and output from your multi-segment IMS application to generate the XML converters.

In addition to identifying the language structures that comprise the request and response messages, you must also describe the order and the maximum number of times a pattern can repeat.

The following example is a segment, or a language structure:

```
01 INPUT-MSG.
   02 IN-LL   PICTURE S9(3) COMP.
   02 IN-ZZ   PICTURE S9(3) COMP.
   02 IN-TRCD PICTURE X(10).
   02 IN-DATA PICTURE X(8).
```

Multiple language structures consist of multiple segments. The order of these segments is important and segments can repeat.

Restrictions: The following restrictions apply to the IMS multi-segment message layouts:

- Only one input and output message layout is allowed for each web service.
- The size of a language structure is limited to the maximum size of a segment: 32 KB.
- Although a language structure can occur more than once, the occurrences must be consecutive. The occurrences cannot be interleaved with occurrences of other language structures.
- At most, one language structure can occur a variable number of times (variable-count) and can be followed only by a language structure with a minimum and maximum count of 1.
- The total maximum size of the message layout, when represented in either XML or binary format, cannot exceed 32 MB when IBM Enterprise COBOL for z/OS Version 3.4 or later is used, or 16 MB when IBM Enterprise COBOL for z/OS Version 3.3 or earlier is used.

The following examples show the supported message layouts. A, B, and C are language structures, and each is less than 32 KB in size.

Layout 1: Fixed count, order-important language structure instances.

- {A1, A2, B1, B2, C1, C2}
- {A1, B1, B2, C1}
- {A1, B1, C1}

Layout 2: Optional, fixed count, order-important language structure instances, followed by one variable count language structure, and ending with one optional language structure instance.

- {A1, [B1, B2,...,Bn], C1}
- {A1, B1, [C1, C2,...,Cn]}
- {A1, A2, B1, B2, [C1, C2,...,Cn]}
- {A1, A2, B1, [C1, C2,...,Cn], D1}
- {[B1, B2,...,Bn]}
- {[B1, B2,...,Bn], C1}

Related information:

Support for multi-segment message processing programs as web services for SOAP Gateway projects in IBM Rational Developer for System z
For more information on Rational Developer for System z support for multi-segment message processing programs as web services, see the Rational Developer for System z information center.

Creating a web service provider from a multi-segment IMS application
For more information on how to create a web service provider from a multi-segment IMS application, see the Rational Developer for System z information center.

Security for the web service provider scenario

SOAP Gateway provides support for both server authentication and client authentication and web-services security (WS-Security) for the web service provider scenario regardless of the platform that SOAP Gateway runs on.

SOAP Gateway clients can secure data exchanges with SOAP Gateway through HTTPS requests by using the SSL/TLS security protocol. Similarly, SSL/TLS connections are supported between SOAP Gateway and IMS Connect.

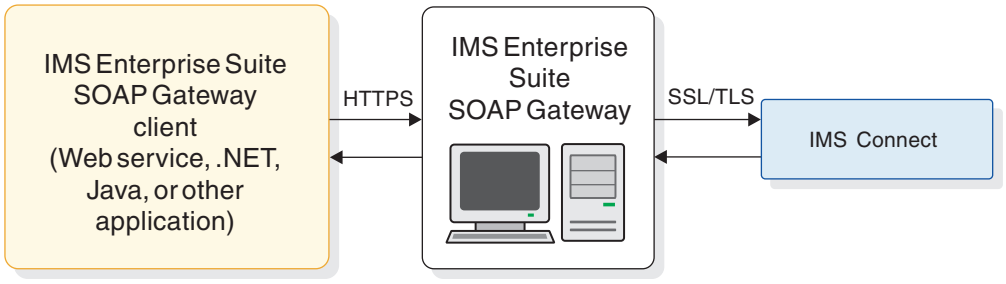


Figure 11. SOAP Gateway as the web service server and the SSL/TLS client

For the z/OS platform, you can use the IBM z/OS Communications Server Application Transparent Transport Layer Security (AT-TLS) feature using SAF to secure the connection. You can also take advantage of the additional security

features in AT-TLS or Quality of Service (QoS), such as security connection refresh settings, maximum connection settings, and revocation of certificates.

Important: For NIST SP800-131a, you must use System SSL between SOAP Gateway and IMS Connect. You must apply the following fix, depending on the IMS version.

- IMS V13 APAR PM96825
- IMS V12 APAR PM98017
- IMS V11 APAR PM98018

The following figure shows the process flow of the client authentication security scheme. With client authentication, both the server that hosts the web service and the client that requests the service require authentication from the other before data is exchanged.

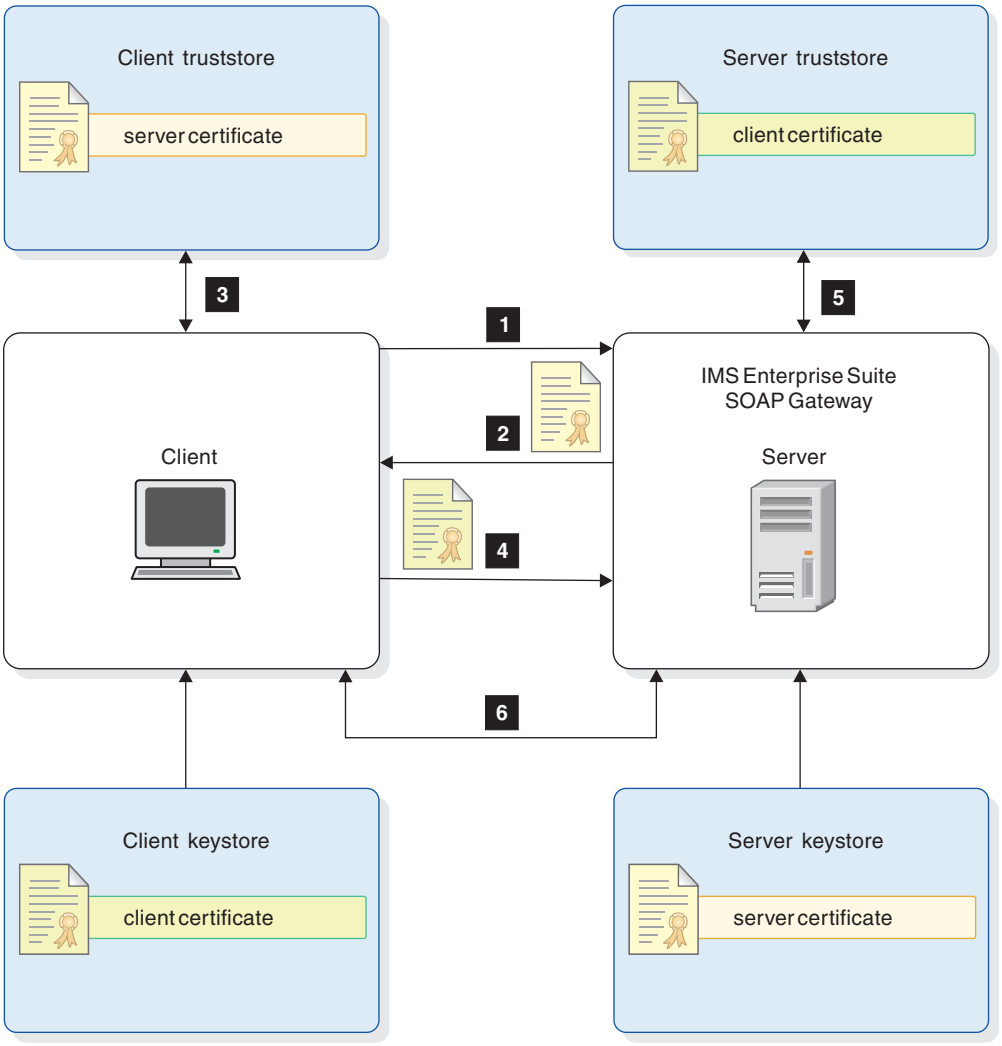


Figure 12. Client authentication for the web service provider scenario

1. The web service client initiates an HTTPS call.
2. The web service server (SOAP Gateway) returns its certificate stored in its server keystore.

3. The client verifies the server certificate with the certificates that are stored in the truststore on the client.
4. The client sends the server its certificate that is stored in the client keystore.
5. The server verifies the client certificate with the certificates that are stored in the truststore on the server.
6. After the transmission is secured, the client is authenticated and allowed to access protected services.

Related concepts:

"Security support in SOAP Gateway" on page 30
SOAP Gateway supports HTTPS communication with its clients, and SSL communications with its host, IMS Connect.

"Secure sockets layer (SSL) and Transport Layer Security (TLS)" on page 33
SSL provides security for your interactions by securing the TCP/IP connection between SOAP Gateway and IMS Connect.

"System SSL" on page 37
System SSL, a feature of the Cryptographic Services base element of z/OS, provides a complete SSL/TLS implementation and a full set of APIs that allow z/OS client and server applications to enable SSL/TLS protection for their TCP network traffic.

Security features supported for the web service provider scenario

SOAP Gateway supports server authentication, client authentication, and web services security (WS-Security). SOAP Gateway on the z/OS platform can further take advantage of the AT-TLS feature in IBM z/OS Communications Server for additional security features.

SOAP Gateway can be configured for server authentication and client authentication through Java keystore (JKS). If SOAP Gateway runs on a z/OS system, you can also use the System Authorization Facility (SAF) to store truststore and keystore information.

Important: z/OS Use of the IBM z/OS Communications Server Application Transparent Transport Layer Security (AT-TLS) feature requires the use of a SAF keyring.

Important: For NIST SP800-131a, you must use System SSL between SOAP Gateway and IMS Connect. You must apply the following fix, depending on the IMS version.

- IMS V13 APAR PM96825
- IMS V12 APAR PM98017
- IMS V11 APAR PM98018

The following table lists the supported security features by platform.

Table 24. Supported security features for the web service provider scenario

Security feature	Platform	Description	Key type (JKS or SAF with AT-TLS)
Server authentication	All SOAP Gateway supported platforms	The server provides server authentication information (certificate) to the client that binds the server identify to subsequent communications.	Both for the z/OS platform JKS for distributed platforms

Table 24. Supported security features for the web service provider scenario (continued)

Security feature	Platform	Description	Key type (JKS or SAF with AT-TLS)
Client authentication	All SOAP Gateway supported platforms	Also known as *mutual authentication* because in addition to server authentication, the client must send certification information to the server.	Both for the z/OS platform JKS for distributed platforms
WS-Security	All SOAP Gateway supported platforms	• UsernameToken Profile support does not require server authentication or client authentication, but use of either server or client authentication is recommended. • SAML Token Profile support requires client authentication.	Both for the z/OS platform JKS for distributed platforms
Custom authentication module	All SOAP Gateway supported platforms	You can plug in your own custom authentication module when WS-Security is enabled and client authentication is configured.	Both for the z/OS platform JKS for distributed platforms
Certificate selection	z/OS	The AT-TLS feature lets you specify a specific certificate in a SAF keyring to use, if more than one is present.	SAF on AT-TLS for the z/OS platform
Certificate revocation list (CRL)	z/OS	The AT-TLS feature lets you configure an LDAP directory that contains your CRLs. AT-TLS passes this list of revoked services into System SSL. Not available for JKS.	SAF on AT-TLS for the z/OS platform
Connection setting	z/OS	The QoS feature supports configuration of connection threshold. The AT-TLS feature supports configuration of connection refresh settings for periodically renegotiating the encryption key and expiring SSL sessions in the cache.	SAF on AT-TLS for the z/OS platform
Additional connectivity rules	z/OS	AT-TLS supports separate configuration of connectivity rules between SOAP Gateway clients to SOAP Gateway, and between SOAP Gateway (an IMS Connect client) and IMS Connect.	SAF on AT-TLS for the z/OS platform

Security process flow with AT-TLS for the web service provider scenario

z/OS

For the web service provider scenario, the IBM z/OS Communications Server AT-TLS feature can be set up to provide security for web services hosted by the SOAP Gateway server.

The following figure shows the security flow when AT-TLS is used to handle security when an IMS application is enabled as a web service, and client authentication is enabled.

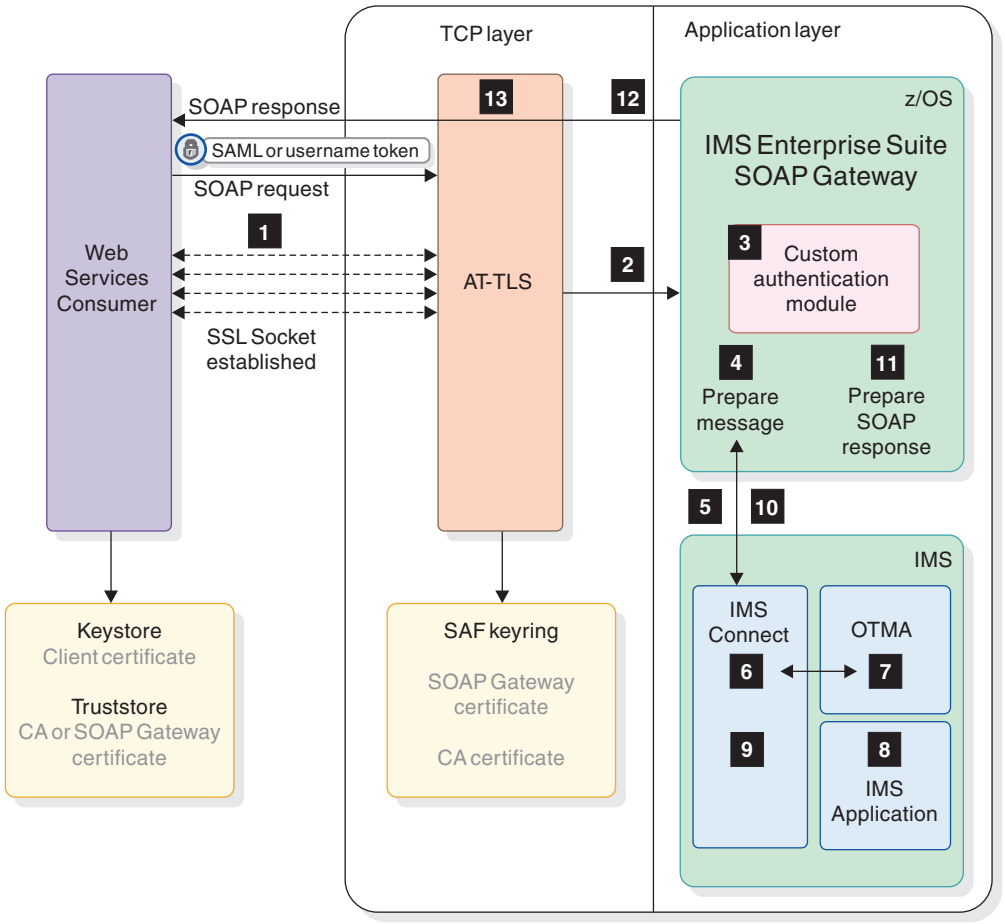


Figure 13. Security process flow with AT-TLS

1. The client sends a SOAP request over HTTPS with a SAML or user name token. The request triggers the following activities on AT-TLS:
   a. An SSL socket connection to AT-TLS is established.
   b. AT-TLS starts the handshake process.
   c. AT-TLS sends a server certificate from the SAF keyring that represents the SOAP Gateway server identity to the client.
   d. The client verifies the incoming server certificate with the CA certificate in the truststore.

e. The client sends the client certificate to AT-TLS (client authentication). AT-TLS verifies incoming client certificate with certificate authority (CA) in a SAF keyring.
f. Optionally, AT-TLS performs certificate revocation list (CRL) validation with the specified LDAP server

2. AT-TLS sends the decrypted message to SOAP Gateway.
3. Optionally, SOAP Gateway loads the custom authentication module if one is configured.
4. SOAP Gateway prepares the message. If WS-Security is enabled, SOAP Gateway extracts the user ID (and password, if using a user name token) from the WS-Security header to the IRM header of the IMS request message.
5. The request (user information and request message) is sent to IMS Connect.
6. IMS Connect authenticates the user based on the user ID and the password (if RACF security is turned on), converts the data from XML to bytes, and passes the user ID and data to OTMA.
7. OTMA authorizes the user ID if security is turned on (OTMASE={CHECK | PROFILE | FULL}). OTMA schedules the transaction to run.
8. The IMS transaction runs.
9. IMS Connect converts the IMS response from bytes to XML.
10. The XML response is sent back to SOAP Gateway.
11. SOAP Gateway prepares the SOAP response.
12. SOAP Gateway returns the SOAP response.
13. AT-TLS encrypts the message and sends the encrypted message to the SOAP Gateway client.

Configuring AT-TLS for SOAP Gateway

z/OS

To configure the IBM z/OS Communications Server AT-TLS feature for SOAP Gateway, use the IBM Configuration Assistant for z/OS Communications Server V1R13.

The Configuration Assistant can be downloaded from the IBM support website. The minimum required version is V1R11.

The high-level steps to configure the AT-TLS feature for SOAP Gateway are as follows:

1. In the Configuration Assistant, configure your z/OS image.
2. Enable the AT-TLS technology.
3. In the AT-TLS perspective:
   a. Configure the port, IP address, cipher, trace level, and keyring information for server authentication.
   b. Configure other settings such as:
      - Client authentication
      - Certificate selection
      - Certificate revocation list (CRL)
      - Connection settings such as cipher reset timer and SSL session timeouts
      - Additional authentication between SOAP Gateway and IMS Connect
4. Install the master AT-TLS policy configuration file.

Related concepts:

"Security process flow with AT-TLS for the web service provider scenario" on page 127
For the web service provider scenario, the IBM z/OS Communications Server AT-TLS feature can be set up to provide security for web services hosted by the SOAP Gateway server.

z/OS V1R13 Communications Server documentation
See the z/OS V1R13 Communications Server documenation in all supported formats.

Related information:

IBM Configuration Assistant for z/OS Communications Server download page
Download the IBM Configuration Assistant for z/OS Communications Server in all supported format.

Configuring the z/OS Communications Server AT-TLS feature for SOAP Gateway: z/OS

You must configure the IP address, handshake role, and port number to set up the IBM z/OS Communications Server AT-TLS feature for SOAP Gateway for the web service provider scenario.

Configuration begins with the creation of a new connectivity rule. A connectivity rule requires the specification of the server IP address and allowed IP addresses for incoming traffic. You must also define a requirement map, a reusable object that describes properties of the network traffic such as protocols and ports, and the security settings, such as cipher selection, client authentication, and SSL session caching.

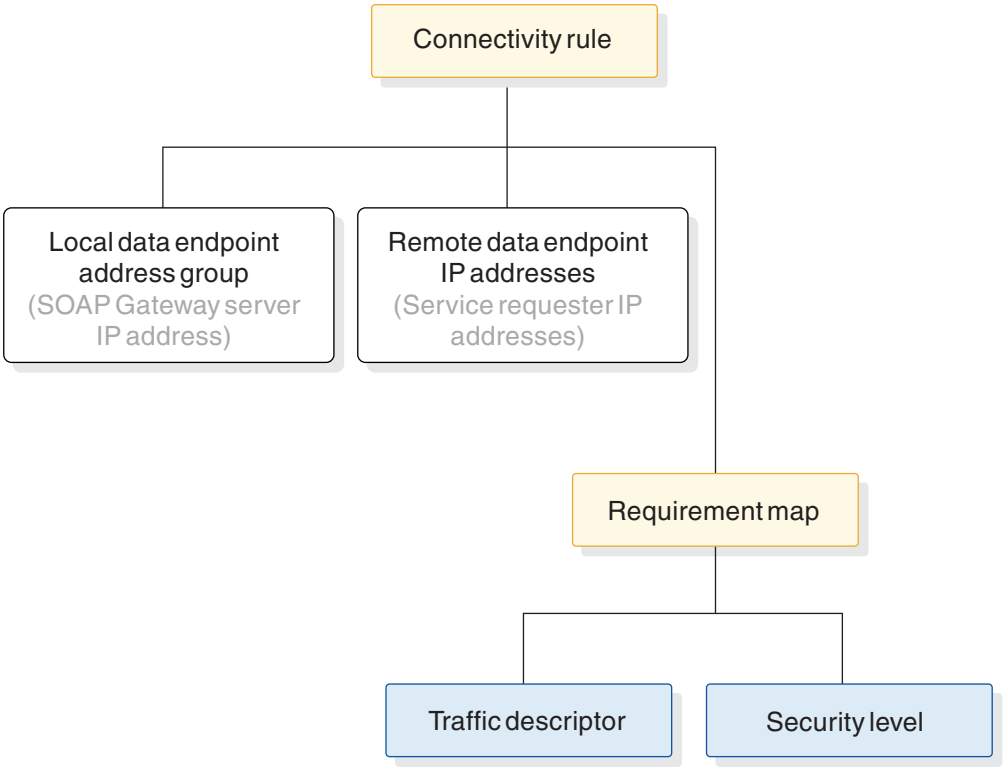


Figure 14. Connectivity rules

Server authentication is the required security setup if you want client authentication or other additional security features.

Configuring AT-TLS for SOAP Gateway for server authentication: z/OS

Use the AT-TLS perspective in IBM Configuration Assistant for z/OS Communications Server V1R13 to configure SOAP Gateway for server authentication.

You must have a z/OS image and an associated TCP/IP stack already configured for the system that SOAP Gateway runs on.

To configure AT-TLS for SOAP Gateway for server authentication:

1. In the Configuration Assistant, go to the AT-TLS perspective by clicking **Perspective** > **AT-TLS**.
2. Click **Add**. The connectivity rule wizard opens.

   A connectivity rule consists of the data endpoints and a requirement map. A requirement map is a set of mappings of traffic descriptors to security levels. A connectivity rule specifies which of those security levels protects that kind of traffic when it is flowing between certain endpoints.

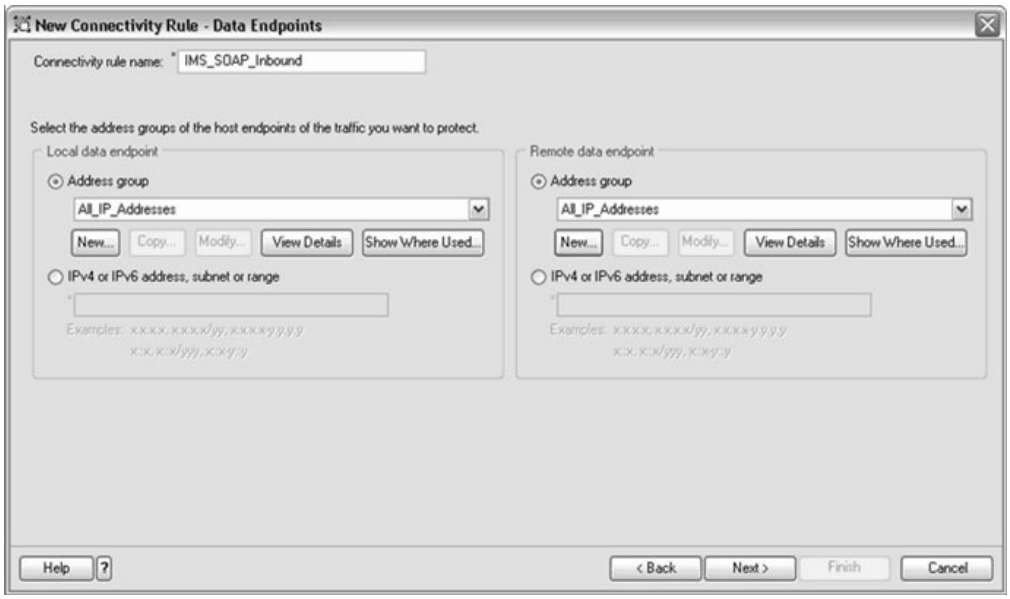


Figure 15. Creating a new connectivity rule in the Configuration Assistant

In this example, the connectivity rule name is IMS_SOAP_Inbound.

3. Click **Next** and specify a name for the connectivity rule. This rule is for inbound traffic to the SOAP Gateway server.
4. Specify the local data endpoint and remote data endpoint. The local data endpoint is where the SOAP Gateway server resides, and remote data endpoints are the clients.
   a. For the local data endpoint:
      1) Click **New** to specify the IP address of the server.
      2) Enter the name for this IP address group, and the IP address, and click **OK**.
   b. For the remote data endpoint:
      1) Specify the range of IP addresses that you want to accept requests for web services that are provided on the SOAP Gateway server. For example, if you want to accept any traffic from the Internet, choose **All_IP_Addresses**.
      2) Click **Next**.

   You are asked to create a new or use an existing requirement map.
5. Enter a name for the requirement map. Each requirement map must have a traffic descriptor.
6. Click **Traffic Descriptors**.
7. In the Traffic Descriptor Objects window, click **Add**.
8. In the New Traffic Descriptor window, enter a name for the descriptor and click **Add** to add a traffic type for this descriptor. The New Traffic Type window opens.

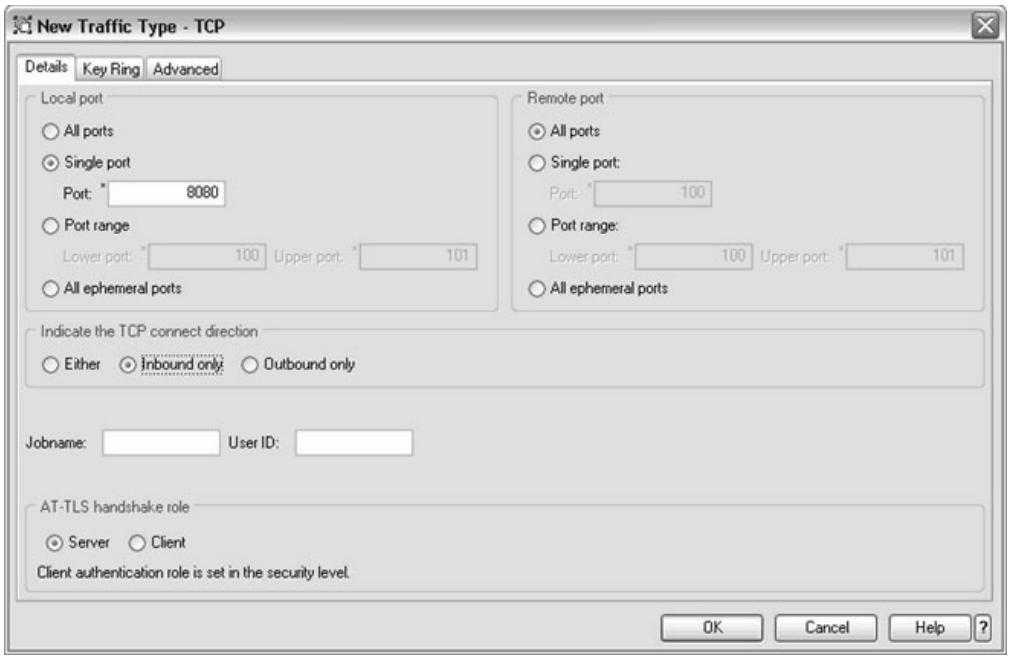


Figure 16. Creating a new traffic type in the Configuration Assistant

9. Specify the local port, the remote port, TCP connection direction, AT-TLS handshake role, and optionally, jobname and user ID.
   a. For local port, specify the port that SOAP Gateway runs on.
   b. For remote port, select the port you want to allow for incoming requests to SOAP Gateway. Typically, the remote port would be all ports.
   c. For TCP connect direction, select **Inbound only** as the direction that this traffic descriptor applies to.
   d. For AT-TLS handshake role, select **Server**. This option means SOAP Gateway performs the handshake as the server. The server must have the client certificate on its keyring.

      A jobname and a user ID are needed when this traffic descriptor is mapped to an AT-TLS security level, where a packet must be to or from an application with this job name for that packet to match this set of traffic characteristics.
   e. Click **OK**. The traffic type is added for the traffic descriptor.
   f. Click **OK**. The traffic descriptor is added.
   g. Click **Close**.
10. In the New Connectivity Rule window, click the **Traffic Descriptor** column header to select the traffic descriptor you created.

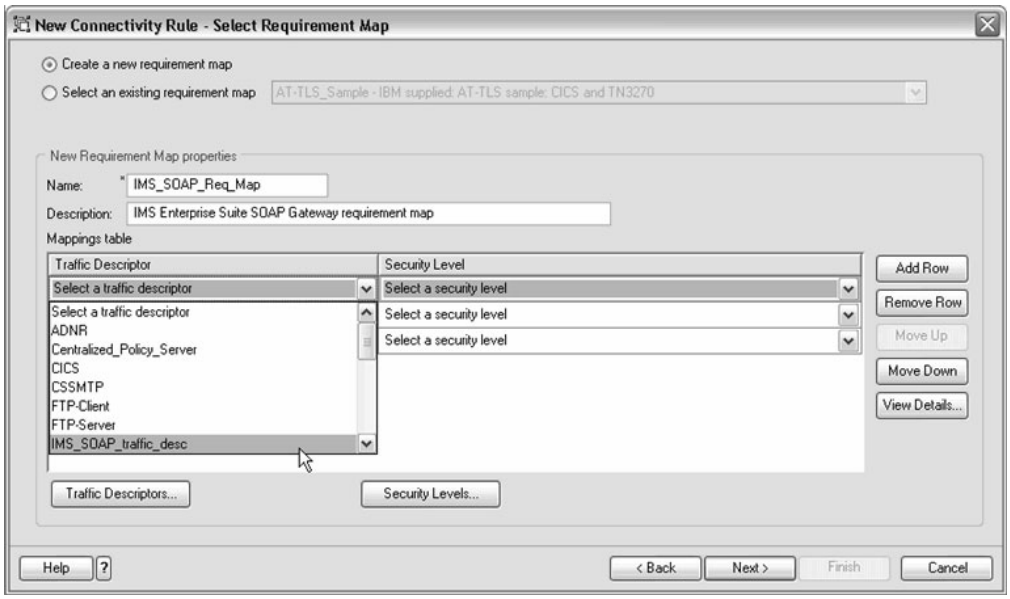


Figure 17. Creating a connectivity rule

11. Click **Security Levels** to add a security level object.
    a. Click **Add**.
    b. Enter a name for the security level and click **Next**.
    c. Select the supported versions of TLS.
    d. If you need to select a cipher:
        1) Click **Use only selected ciphers**.
        2) Click **Choose Ciphers**.
        3) Click to select your cipher options. For example, select TLS_RSA_WITH_AES_128_CBC_SHA for a 128-bit strong encryption cipher suite.
        4) Click **Next**.

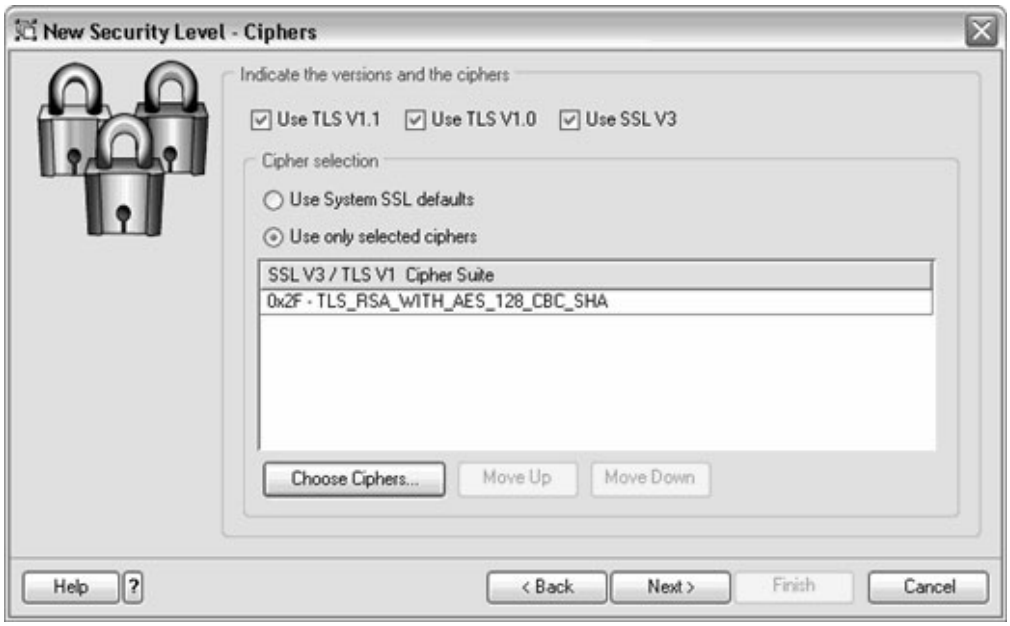


Figure 18. Creating a security level

12. Click **Advanced Settings** if you want to configure for client authentication, tune the cipher suite timer and caching, specify certificate validation mode, and others. For more information about client authentication, see the topic on configuring AT-TLS for client authentication.
13. Click **Finish**.
14. In the New Connectivity Rule window, click the **Security Level** column heading, and select the requirement map that you created.
15. Click **Next**, and then click **Finish**.

View the configuration file by right-clicking your image, selecting **Install Configuration File**, selecting the stack for your z/OS image, and clicking **Show Configuration File**. The following example shows a generated configuration file:

```
## AT-TLS Policy Agent Configuration file for:
##    Image: YRL
##    Stack: YRLTCP
##
## Created by the IBM Configuration Assistant for z/OS Communications Server
## Version 1 Release XX
## Backing Store = .\files\saveData
## FTP History:
##
## End of Configuration Assistant information
TTLSRule                          IMS_SOAP_Inbound~1
{
  LocalAddrGroupRef               IMS_SOAP
  RemoteAddr                      ALL
  LocalPortRangeRef               portR1
  Direction                       Both
  Priority                        255
  TTLSGroupActionRef              gAct1~IMS_SOAP_traffic_desc
  TTLSEnvironmentActionRef        eAct1~IMS_SOAP_traffic_desc
  TTLSConnectionActionRef         cAct1~IMS_SOAP_traffic_desc
}
TTLSGroupAction                   gAct1~IMS_SOAP_traffic_desc
{
  TTLSEnabled                     On
}
```

```
TTLSEnvironmentAction             eAct1~IMS_SOAP_traffic_desc
{
  HandshakeRole                   Server
  EnvironmentUserInstance         0
  TTLSKeyringParmsRef             keyR~YRL
}
TTLSConnectionAction              cAct1~IMS_SOAP_traffic_desc
{
  HandshakeRole                   Server
  TTLSConnectionAdvancedParmsRef  cAdv1~IMS_SOAP_traffic_desc
  CtraceClearText                 Off
  Trace                           2
}
TTLSConnectionAdvancedParms       cAdv1~IMS_SOAP_traffic_desc
{
  SecondaryMap                    Off
}
TTLSKeyringParms                  keyR~YRL
{
  Keyring                         tlsKeyring
}
TTLSCipherParms                   cipher1~IMS_SOAP_Security
{
  V3CipherSuites                  TLS_RSA_WITH_AES_128_CBC_SHA
}
IpAddrGroup                       IMS_SOAP
{
    IpAddr
    {
        Addr 9.9.9.9
    }
}
PortRange                         portR1
{
  Port                            8080
}
```

- TTLSRule contains the name of the connectivity rule, IMS_SOAP_Inbound.
- The IP address group name is IMS_SOAP, with the IP address and port number for theSOAP Gateway server
- The HandshakeRole parameter of the TTLSConnectionAction policy statement and TTLSEnvironmentAction statement is set to Server.
- Your SOAP Gateway traffic descriptor (IMS_SOAP_traffic_desc in this example) is the action associated with the TTLSConnectionAction policy statement.
- TTLSCipherParms is set to your SOAP Gateway security level (IMS_SOAP_Security in this example).
- V3CipherSuites is set to your cipher suite types (TLS_RSA_WITH_AES_128_CBC_SHA in this example)

Related tasks:

"Configuring AT-TLS for SOAP Gateway for client authentication"
Use the AT-TLS perspective in the IBM Configuration Assistant for z/OS Communications Server V1R13 to configure SOAP Gateway for client authentication.

Configuring AT-TLS for SOAP Gateway for client authentication: z/OS

Use the AT-TLS perspective in the IBM Configuration Assistant for z/OS Communications Server V1R13 to configure SOAP Gateway for client authentication.

You must have configured AT-TLS for server authentication.

1. In the Configuration Assistant, go to the AT-TLS perspective by clicking **Perspective** > **AT-TLS**.
2. Click the TCP/IP stack of your z/OS image.
3. Click the connectivity rule for SOAP Gateway, and click **Modify**.
4. Click the **Select Requirement Map** tab.
5. Click **Security Levels**.
6. In the Security Level Objects window, click the security level associated with SOAP Gateway, and click **Modify**.
7. Click the **Additional Settings** tab.
8. Click **Advanced Settings**.
9. Click the **Client Authentication** tab.
10. Click **Use client authentication**.

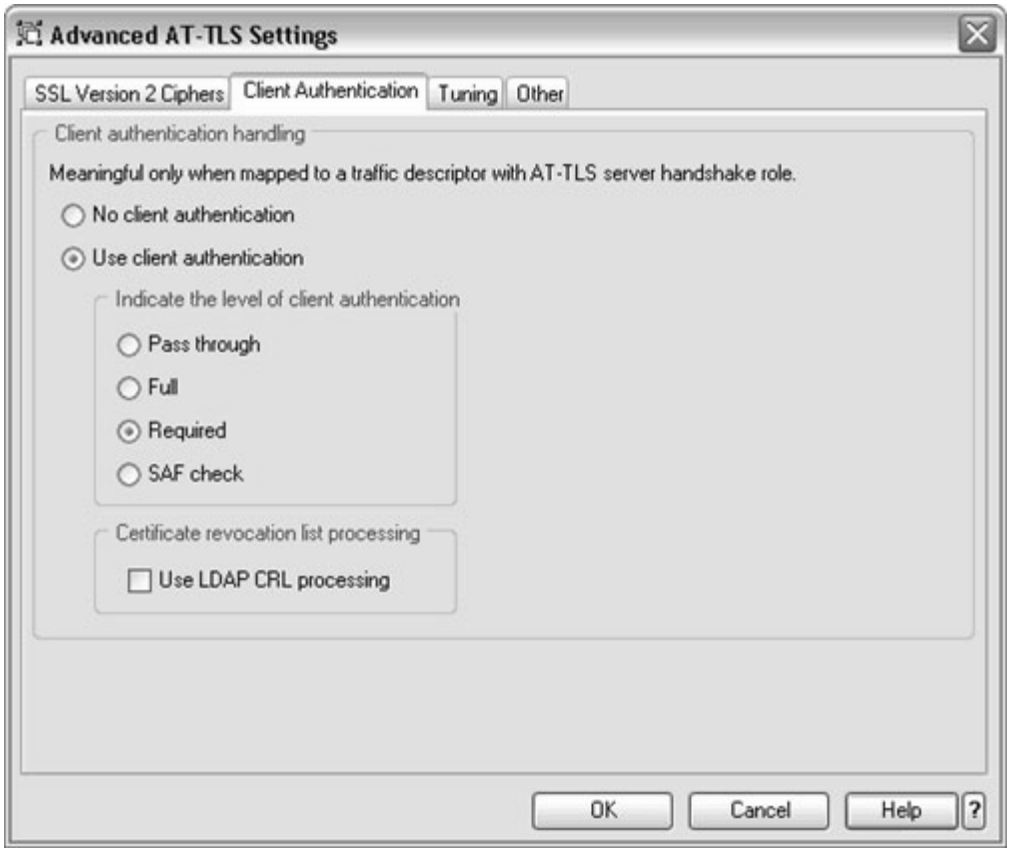


Figure 19. Configuring client authentication in the advanced AT-TLS settings

11. For the level of client authentication, click **Required**. This option requires the client to present a certificate and performs client certificate validation. The SAF check option is not supported for SOAP Gateway client authentication.
12. Click **OK** and **Close** multiple times until you return to the TCP/IP stack page in the AT-TLS perspective.
13. Click **Apply Changes**.

View the configuration file by right-clicking your image, selecting **Install Configuration File**, selecting the stack for your z/OS image, and clicking **Show Configuration File**.

The following example shows the part of the generated configuration file that relates to the authentication information. The client authentication information is reflected in the HandshakeRole parameter of the TTLSConnectionAction and TTLSEnvironmentAction policy statements, with the value set to ServerWithClientAuth.

```
TTLSEnvironmentAction             eAct1~IMS_SOAP_traffic_desc
{
  HandshakeRole                   ServerWithClientAuth
  EnvironmentUserInstance         0
  TTLSKeyringParmsRef             keyR~YRL
}
TTLSConnectionAction              cAct1~IMS_SOAP_traffic_desc
{
  HandshakeRole                   ServerWithClientAuth
  TTLSCipherParmsRef              cipher1~IMS_SOAP_Security
  TTLSConnectionAdvancedParmsRef  cAdv1~IMS_SOAP_traffic_desc
  CtraceClearText                 Off
  Trace                           7
}
```

Revoking certificates: z/OS

You can specify an LDAP server in the IBM Configuration Assistant for z/OS Communications Server V1R13 that contains your certificate revocation list (CRL) and passes that list of revoked services into System SSL to use.

System SSL validates peer certificates against a list of revoked certificates during SSL handshake. You can specify the TTLSGskLdapParms statement to define a set of LDAP parameters, including the LDAP connection, cache timeout, and security level.

Support for CRL requires client authentication.

To configure for CRL in IBM Configuration Assistant for z/OS Communications Server V1R13:

1. Go to the AT-TLS perspective.
2. In the Advanced ATL-TLS Image Settings window, click **Add** to add your LDAP server (IP address or host name) and the port number.

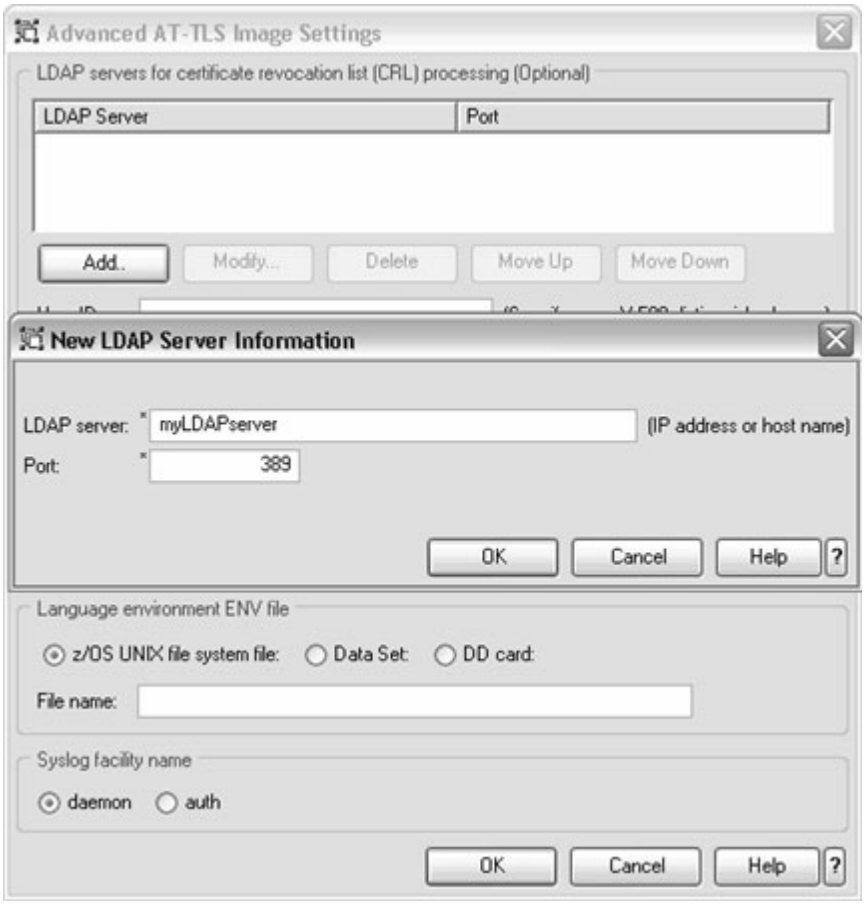


Figure 20. Adding the LDAP server information for the certificate revocation list

3. Click the **Client Authentication** tab and click **Required** to indicate the level of client authentication.

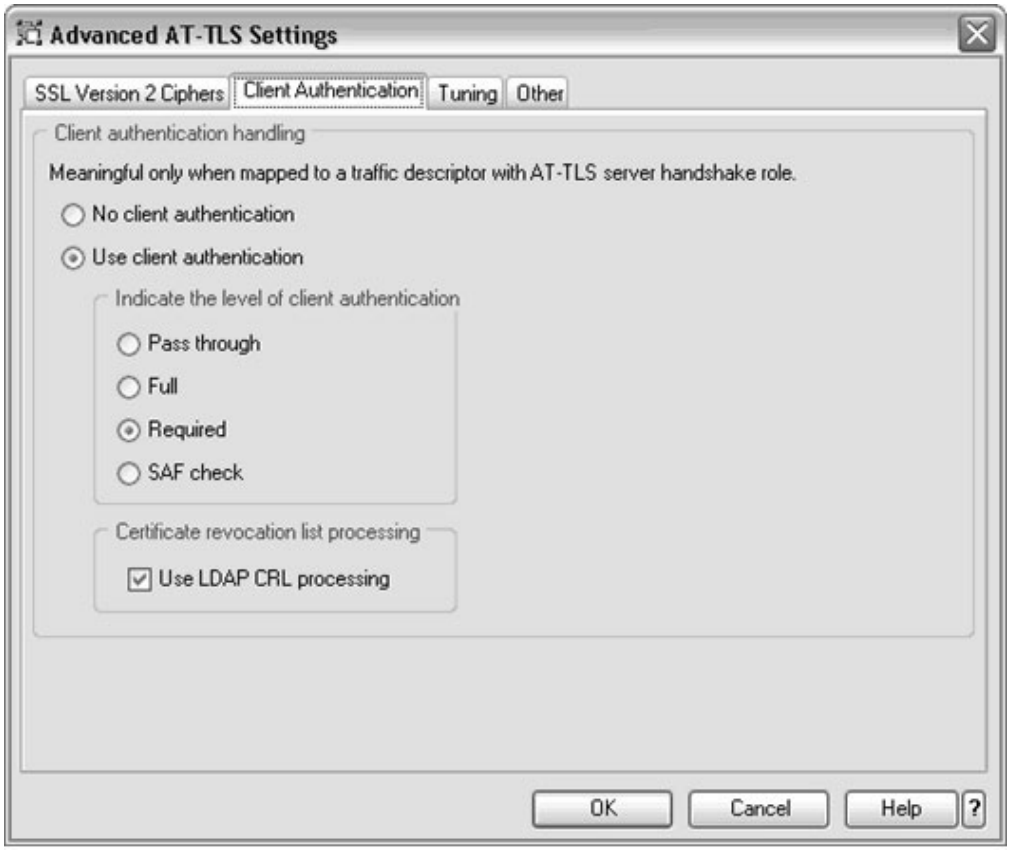


Figure 21. Enabling client authentication and CRL processing

4. Click the **Use LDAP CRL processing** check box.
5. Click **OK**.

Related concepts:

"System SSL" on page 37
System SSL, a feature of the Cryptographic Services base element of z/OS, provides a complete SSL/TLS implementation and a full set of APIs that allow z/OS client and server applications to enable SSL/TLS protection for their TCP network traffic.

Refer to the z/OS V1R13 Communications Server IP Configuration Guide for more information about CRL and LDAP configuration.
For more detail about the CRL function, refer to the z/OS V1R13 Communications Server IP Configuration Guide for LDAP server information.

Resetting the cipher key: z/OS

You can specify the timer to reset the cipher key in the IBM Configuration Assistant for z/OS Communications Server V1R13 for a new session key to be generated when the timer value is reached.

Use the Reset Cipher Key timer parameter to specify the number of minutes a secure connection can be active before a new session key is generated for the connection. The default is that the session key is not refreshed for the life of the connection.

Support for the Reset Cipher Key timer applies only to SSL V3 and TLS V1 connections.

To set the cipher key timer value:

1. In the AT-TLS perspective, click the TCP/IP stack of your z/OS image.
2. Click the connectivity rule for SOAP Gateway, and click **Modify**.
3. Click the **Select Requirement Map** tab.
4. Click **Security Levels**.
5. In the Security Level Objects window, click the security level associated with SOAP Gateway, and click **Modify**.
6. Click the **Additional Settings** tab.
7. Click **Advanced Settings**.
8. Click the **Tuning** tab.
9. Click **Reset the key every**.
10. Specify the number of minutes to reset the cipher key. The number must be an integer from 1 to 1440.
11. Click **OK** and **Close** multiple times until you are back to the TCP/IP stack page.
12. Click **Apply Changes**.

View the configuration file by right-clicking your image, selecting **Install Configuration File**, selecting the stack for your z/OS image, and clicking **Show Configuration File**.

The sample generated configuration information might look as follows:

```
TTLSConnectionAdvancedParms       cAdv1~IMS_SOAP_traffic_desc
{
  ResetCipherTimer                1440
}
```

The ResetCipherTimer value is set in the TTLSConnectionAdvancedParms policy statement.

Related concepts:

Refer to the z/OS V1R13 Communications Server IP Configuration Guide for more information about CRL and LDAP configuration.
For more detail about the CRL function, refer to the z/OS V1R13 Communications Server IP Configuration Guide for LDAP server information.

Setting the SSL session timeout value: z/OS

You can specify the number of seconds until a cached TLS V1 or SSL V3 session identifier expires.

To set the SSL session timeout value:

1. In AT-TLS perspective, click the TCP/IP stack of your z/OS image.
2. Click the connectivity rule for SOAP Gateway, and click **Modify**.
3. Click the **Select Requirement Map** tab.
4. Click **Security Levels**.
5. In the Security Level Objects window, click the security level associated with SOAP Gateway, and click **Modify**.
6. Click the **Additional Settings** tab.

7. Click **Advanced Settings**.
8. Click the **Tuning** tab.
9. Click the **Cache session identifier** button for the version of SSL or TLS that you use.
10. Specify the number of seconds the session key is to be cached.
    - For SSL V3 or TLS V1, the number must be an integer from 1 to 86400. The default is 86400.
    - For SSL V2, the number must be an integer from 1 to 100. The default is 100.
11. Click **OK** and **Close** multiple times until you are back to the TCP/IP stack page.
12. Click **Apply Changes**.

View the configuration file by right-clicking your image, selecting **Install Configuration File**, selecting the stack for your z/OS image, and clicking **Show Configuration File**. The new value you specify is set in the TTLSGskAdvancedParms policy statement.

The following example shows that an SSL V3 or TLS V1 session is cached for 50000 seconds.

```
TTLSGskAdvancedParms              gskAdv1~IMS_SOAP_Security
{
  GSK_V3_SESSION_TIMEOUT          50000
}
```

Related concepts:

Refer to the z/OS V1R13 Communications Server IP Configuration Guide for more information about CRL and LDAP configuration.
For more detail about the CRL function, refer to the z/OS V1R13 Communications Server IP Configuration Guide for LDAP server information.

Configuring additional connectivity rules for SOAP Gateway-to-IMS Connect communication:

In addition to the inbound message connection rule for connections from the web service client (service requester) to SOAP Gateway, you can configure an additional set of rules for the connections between SOAP Gateway and IMS Connect.

The following figure shows the two additional rules that are created to protect the traffic between SOAP Gateway and IMS Connect. Each connectivity rule can use a different set of certificates in the System Authorization Facility (SAF) keyring.

If AT-TLS is used to authenticate local traffic on the same z/OS image, as depicted in the following scenario, the SSL handshake is completed, but encryption and decryption processing is bypassed.

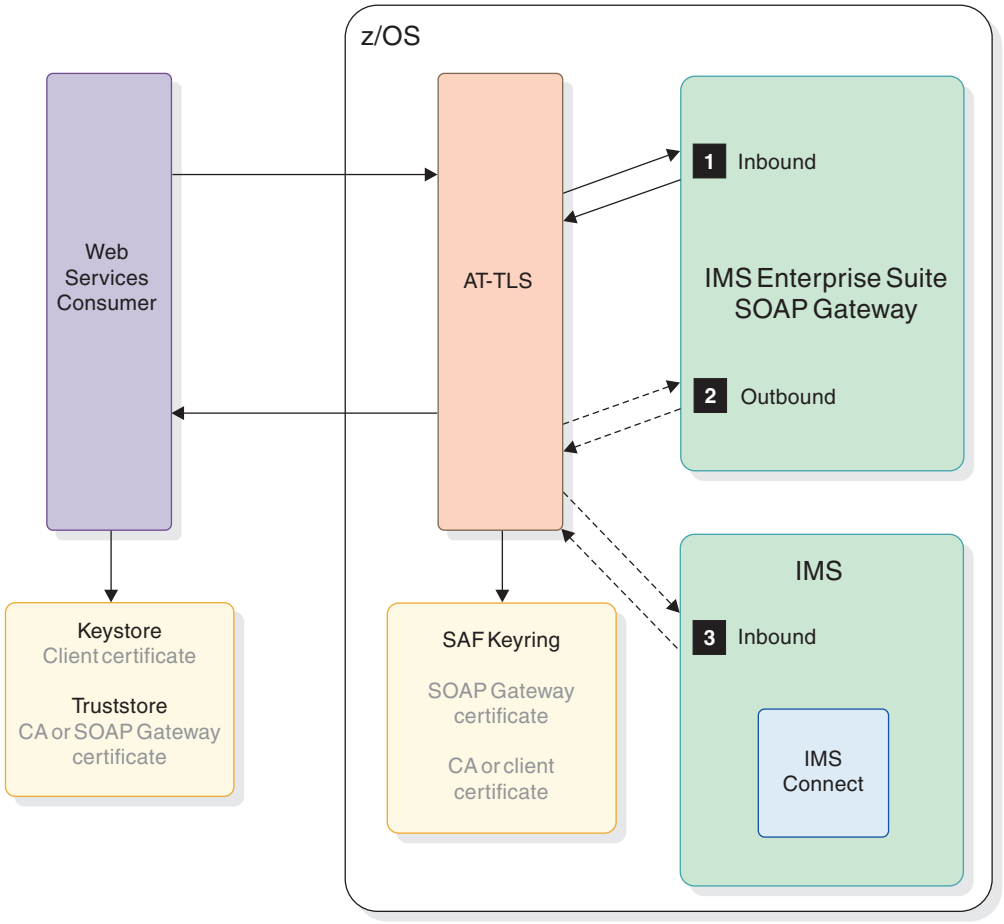


Figure 22. Connectivity rules when AT-TLS is used to protect IMS Connect and their TCP connection directions

1. The first connection rule is for inbound messages from the web service client (service requester) to SOAP Gateway.
2. The added second connection rule is for outbound messages from SOAP Gateway to IMS Connect.
3. The added third connection rule is for IMS Connect to process inbound messages from SOAP Gateway or other IMS Connect clients.

To create the second and the third connectivity rules to protect the traffic to IMS Connect, follow the similar steps to create the connectivity rules and requirement maps, as described in the "Configuring AT-TLS for SOAP Gateway for server authentication" on page 130 topic.

For the outbound communication from SOAP Gateway to IMS Connect, in the New Traffic Type window, specify the following information:

Local port
All the ports that SOAP Gateway sends outbound messages to IMS Connect.

Remote port
The port that IMS Connect runs on.

TCP connect direction
Outbound only.

For the inbound communication into IMS Connect from SOAP Gateway (or other IMS Connect clients), in the New Traffic Type window, specify the following information:

Local port
The port for IMS Connect.

Remote port
All the ports that IMS Connect accepts incoming requests

TCP connect direction
Inbound only.

Important: When you create the connection bundle for the web service by using the iogmgmt -conn command, do not specify any JKS SSL properties. Specifications of any JKS SSL properties in an AT-TLS environment will result in SSL handshake failure.

Configuring maximum connections for SOAP Gateway:

Use the Quality of Service (QoS) feature in IBM z/OS Communications Server to specify traffic thresholds and traffic priority to help manage traffic to the SOAP Gateway server.

Quality of Service refers to the overall service that a user or application receives from a network, in terms of throughput and delay.

To configure the maximum connections, create a traffic descriptor, a traffic shaping level, and a requirement map in the QoS perspective.

1. In the Configuration Assistant, enable QoS.
   a. In the main perspective, in the navigation tree, click your image, or the TCP/IP stack for which you want to enable QoS.
   b. In the z/OS Communications Server technologies section, if the QoS technology is not enabled, select **QoS** and click **Enable**.
   c. Open the QoS perspective by clicking **Perspective** > **QoS**.
2. Create a traffic descriptor.
   a. Click **Traffic Descriptors**.
   b. Click **Add**.

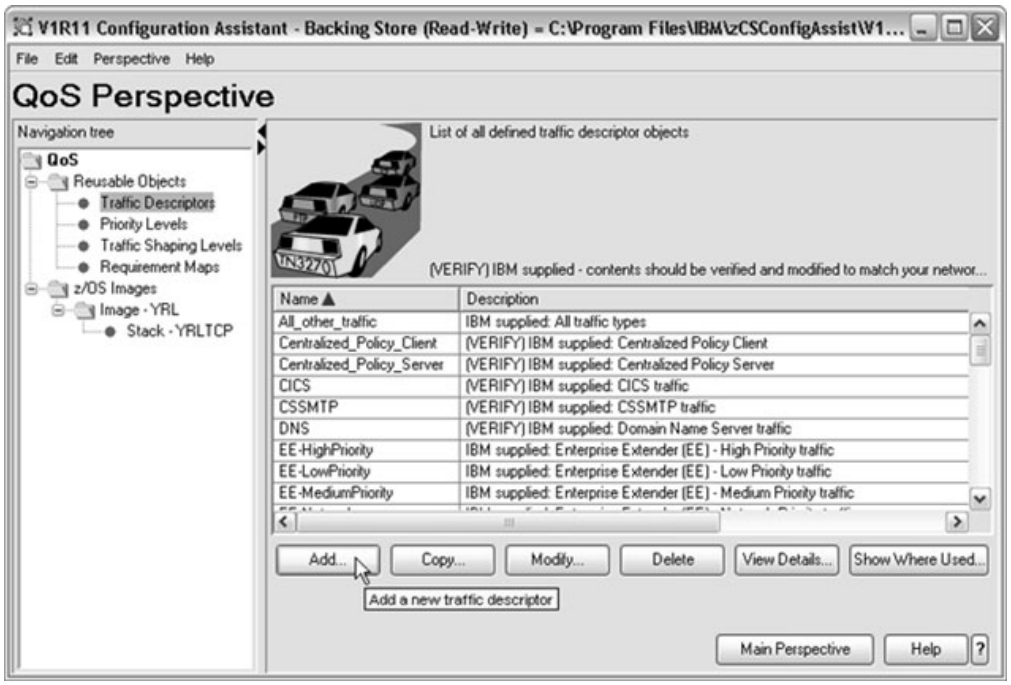


Figure 23. Adding a traffic descriptor in the QoS perspective

c. Specify a name and a description.
d. Select TCP as the protocol, and click **Add**.

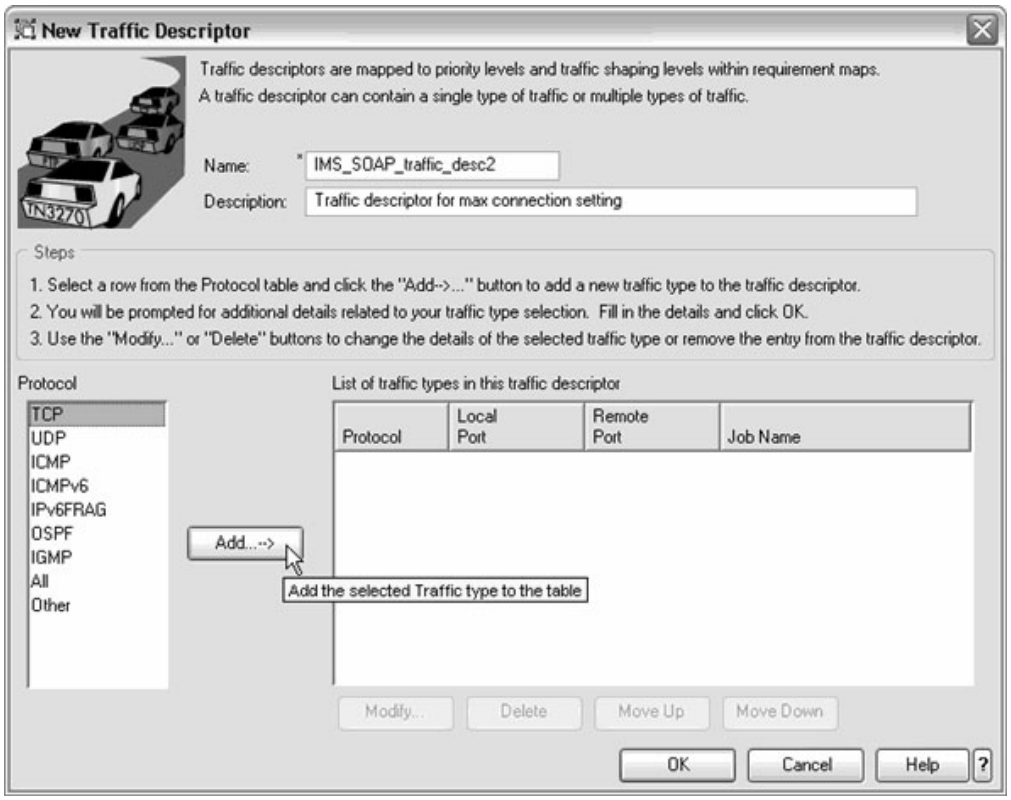


Figure 24. Specifying the traffic type by selecting a protocol.

e. In the New Traffic Type – TCP window, specify the port, or the port range.

Tip: The online help describes these fields and valid values.

Important: Specify the same values that you specified when you set up AT-TLS for server authentication or client authentication.

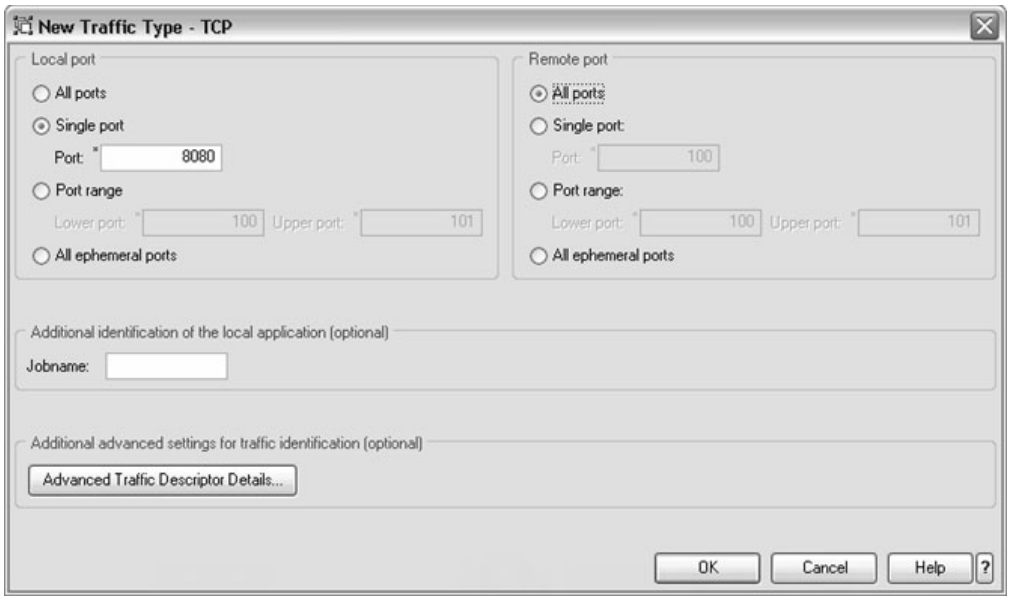


Figure 25. Specifying the local port and the remote port

f. Click **OK**.

g. Click **OK**. The new traffic descriptor is added to the descriptor list.

3. Create a traffic shaping level, where you specify the maximum number of connections allowed.

   a. Click **Reusable Objects** > **Traffic Shaping Levels** in the navigation tree, and then click **Add**.

   b. In the New Traffic Shaping Level window, specify the name and a description, and click **Next**.

   c. To set the connection threshold, click **Connection limit**, and enter the maximum number of connections allowed. Use the online help to guide you through the definition of these fields and their valid values. The default maximum connection value is 65535.

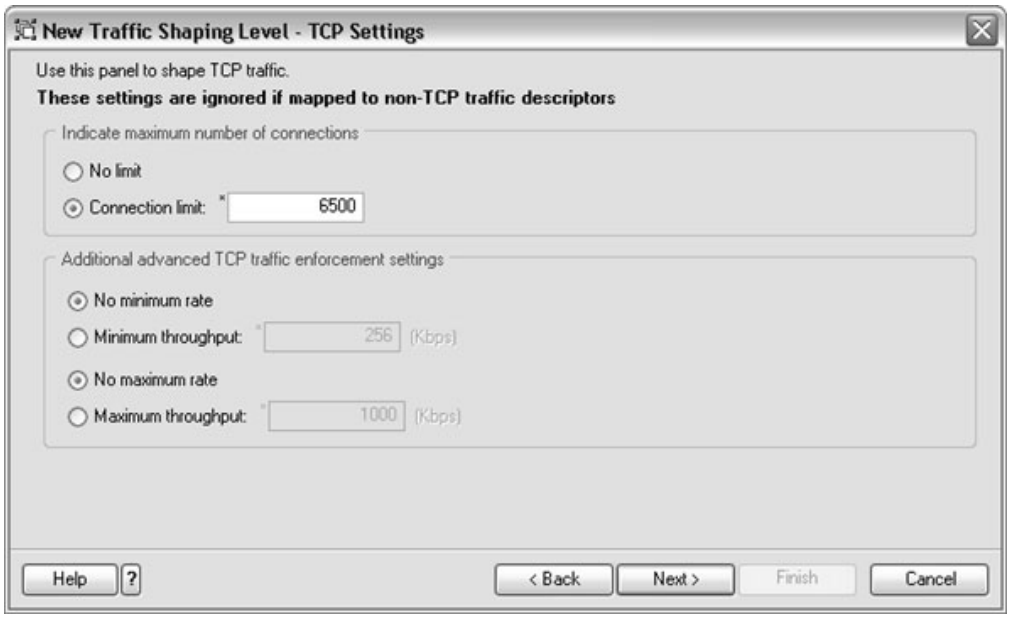


Figure 26. Specifying the connection limit

d. Click **Next** for advanced settings and configure based on your environment.
e. Click **Finish**. The newly defined traffic shaping level is added to list of traffic shaping level objects.

4. Create a requirement map.
   a. Click **Reusable Objects** > **Requirement Maps** in the navigation tree.
   b. Click **Add** to add a requirement map.
   c. Specify the name and a description for the requirement map.
   d. Select your traffic descriptor from the list of traffic descriptors, and click **Add** to add to the requirement map table.
   e. Select the traffic shaping level from the list.

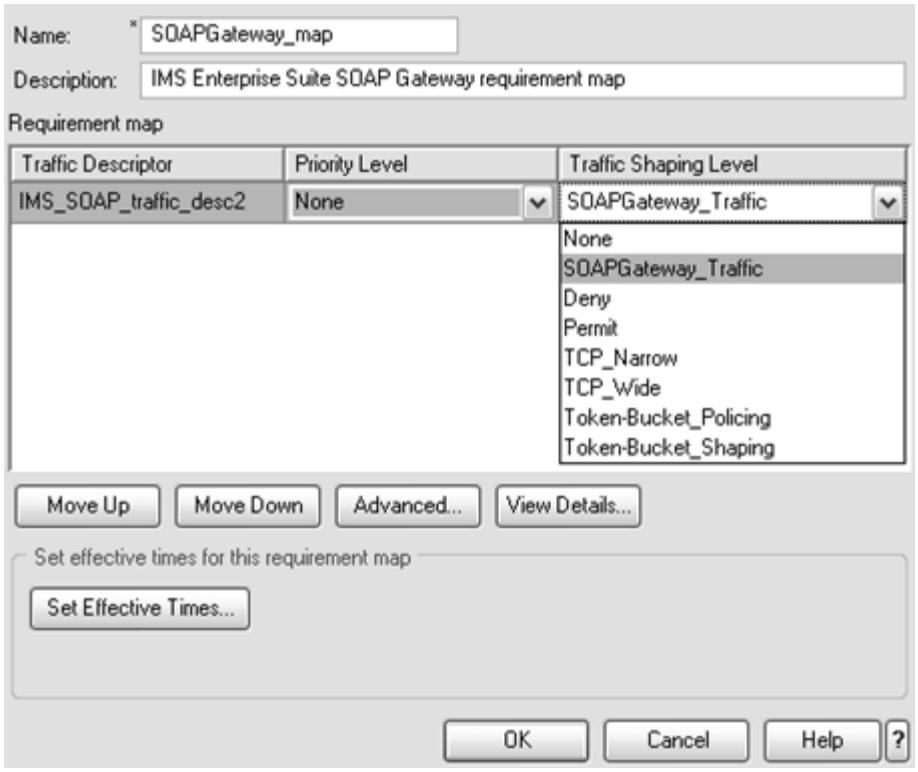


Figure 27. Selecting the traffic shaping level for the traffic descriptor

f. Click **OK**, and then click **Proceed**.

g. Click **Finish**.

5. Create a connection rule that uses the requirement map that you created. If you already have a connection rule, you can follow similar steps to select the appropriate requirement map.

   a. In the navigation tree, click the TCP/IP stack of your image, and click **Add**.

   b. Specify the connectivity rule name, local data endpoint, remote data endpoint, and then click **Next**.

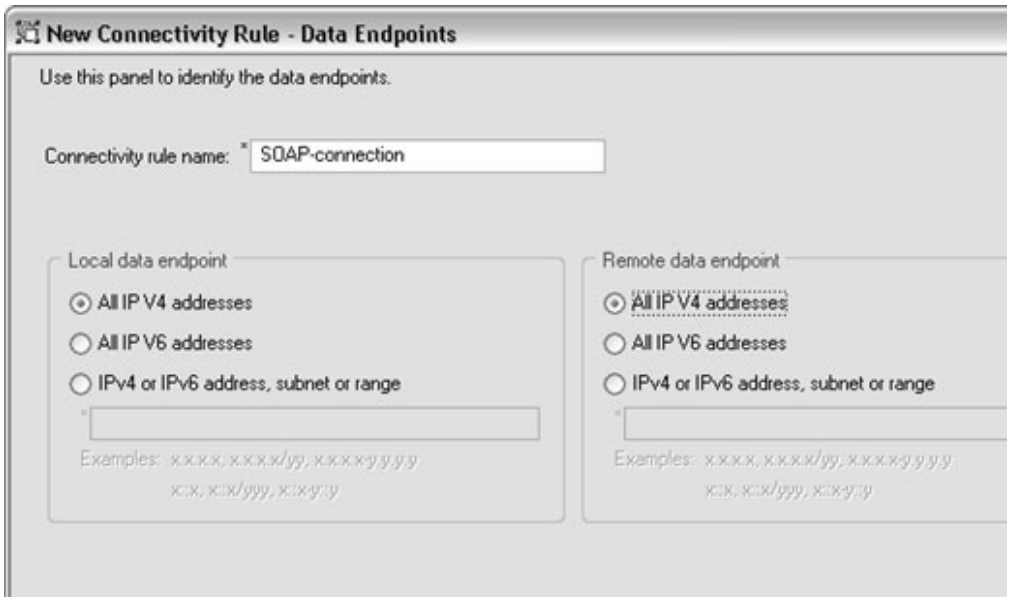


Figure 28. Specifying the connectivity rule name and Identifying the local and remote data endpoints

c. Select the appropriate requirement map, and click **Finish**.

If you view the configuration file (Right-click your image and click **Install Configuration File**, click the stack for your z/OS image, and click **Show Configuration File**), the generated configuration information is as follows:

```
########################################################################
# PolicyRule statements
########################################################################

policyRule SOAP-connection~1
{
   PolicyRulePriority        65000
   SourcePortRange           8080
   DestinationPortRange      0
   ProtocolNumberRange       6
   PolicyActionReference     action~1
}
########################################################################
# PolicyAction Statements
########################################################################

PolicyAction action~1
{
   PolicyScope               DataTraffic
   MaxConnections            6500
}
```

The MaxConnections value reflects the maximum connections value that you specified.

Configuring SSL and HTTPS support with Java keystore (JKS)

To use HTTPS between a SOAP Gateway client and SOAP Gateway, or SSL between SOAP Gateway and its server (IMS Connect), you must create the keystore and truststore, and configure the SOAP Gateway server.

To configure SOAP Gateway for HTTPS communications with its clients, and SSL communications with IMS Connect:

1. If the SOAP Gateway client (for example, a web service or a Java application) does not require authentication of the SOAP Gateway server, go to step 2. If server or client authentication is required:
   a. Create a keystore for SOAP Gateway that contains a private and public key pair and export it as a server certificate.
   b. Create a client truststore, and import the SOAP Gateway server certificate.
   c. If client authentication is required:
      1) Create a client keystore, and export the client certificate.
      2) Create a truststore for SOAP Gateway
      3) Transfer and import the client certificate into the SOAP Gateway truststore.
   d. Configure SOAP Gateway with server authentication, truststore, and keystore information by using the SOAP Gateway management utility iogmgmt -prop -u command to update the SOAP Gateway server properties.
2. If SSL security is required between SOAP Gateway and IMS Connect, use of the Application Transparent Transport Layer Security (AT-TLS) is recommended. IBM z/OS Communications Server provides this AT-TLS feature.

   To set up System SSL between SOAP Gateway and IMS Connect, see the IMS Connect SSL connections topic in IMS Communications and Connections information.

   If NIST SP800-131a is required, you must use System SSL between SOAP Gateway and IMS Connect. You must apply the following fix, depending on the IMS version:
   - IMS V13 APAR PM96825
   - IMS V12 APAR PM98017
   - IMS V11 APAR PM98018

   If you use Java keystore (JKS):
   a. Create a truststore for SOAP Gateway if you have not done so.
   b. Export the IMS Connect certificate.
   c. Import the IMS Connect certificate to the SOAP Gateway truststore.
   d. Decide the IMS Connect SSL port to use. Set up the IMS Connect and SSL configuration members with the appropriate values. For more information about setting up these configuration members, see the IMS Version 13 Communications and Connections information.
3. Set up the connection bundle with the appropriate SSL parameters, including the HTTPS port number from step 2d. Use the SOAP Gateway management utility iogmgmt -conn -u command to update the connection bundle, including the port number, the SSL keystore name and password, and the SSL truststore name and password.

   For FIPS compliance, if you are using System SSL, the SSL encryption type (the -e flag) should be set to STRONG by using the iogmgmt -conn -u command.

   If you are using AT-TLS, do not specify any JKS SSL properties. Specifications of any JKS SSL properties in an AT-TLS environment will result in SSL handshake failure.
4. If IMS Connect requires authentication of the client, use of AT-TLS is recommended. If you use JKS, import the SOAP Gateway public key certificate into the keyring. For more information, see the IMS Version 13 Communications and Connections information.

Related concepts:

"Secure sockets layer (SSL) and Transport Layer Security (TLS)" on page 33
SSL provides security for your interactions by securing the TCP/IP connection between SOAP Gateway and IMS Connect.

Related tasks:

"Creating a connection bundle entry for callout applications" on page 266
Create a connection bundle entry that describes the connection properties for accessing IMS by using the SOAP Gateway management utility. The connection bundle entries are stored in the `connbundle.xml` file.

Related information:

IMS Connect SSL connections in IMS V13 Communications and Connections

Creating the server keystore for SOAP Gateway and exporting the public key as a certificate:

Create a server keystore for SOAP Gateway and export the public key as a server certificate that the SOAP Gateway client can use to verify that the server is trusted.

To create a JKS keystore on the server and export the public key:

1. Create a keystore by using a Key management tool such as Ikeyman or Keytool. In a command console, enter the following command:

   ```
   keytool -genkey -alias server.keystore -dname
   "CN=mycompany.somewhere.com OU=IBM SWG, O=IBM, C=US"
   -keyalg RSA -keypass password -storepass password
   -keystore "/path/to/server.keystore.ks"
   ```

   The CN value must include the hostname. The same hostname must be in the URL that is used by the client to request for the service. For example, http://**mycompany.somewhere.com**:8088/imssoap/services/IMSPHBKService.

   For NIST SP800-131a, specify SHA256withRSA for the signature algorithm and 2048 for the key size.

   ```
   keytool -genkey -alias server.keystore -dname
   "CN=mycompany.somewhere.com OU=IBM SWG, O=IBM, C=US"
   -keyalg RSA -sigalg SHA256withRSA  -keysize 2048
   -keypass password -storepass password
   -keystore "/path/to/server.keystore.ks"
   ```

2. Export the public key from the server keystore (server.keystore.ks in the following example) as a certificate.

   ```
   keytool -export -alias server.keystore -storepass password
   -file "/path/to/server.keystore.cer"
   -keystore "/path/to/server.keystore.ks"
   ```

   Tip: You can have the certificate signed by a Certificate Authority (CA), such as VeriSign, or create your own CA by using software such as OpenSSL to sign your own (self-signed) certificate.

Create a client truststore, and import the SOAP Gateway server certificate.

Creating the client truststore and importing the server certificate:

Create the Java truststore for the client and import the SOAP Gateway server certificate into the client truststore so that the client can authenticate the server.

To create the Java truststore for the client and import the SOAP Gateway server certificate:

1. Create the Java truststore for the SOAP Gateway client (client.truststore.ks).

```
keytool -genkey -alias client.truststore
-dname "CN=IMS Client Truststore, OU=IBM SWG, O=IBM, C=US"
-keyalg RSA -keypass password -storepass password
-keystore "/path/to/client.truststore.ks"
```

2. Import the SOAP Gateway server certificate (server.keystore.cer) into the client truststore (client.truststore.ks). Provide an alias by which the certificate is to be identified.

   Important: When you import a new trusted certificate, the alias must not yet exist in the keystore.

   ```
   keytool -import -v -trustcacerts -alias server
   -file "path/to/server.keystore.cer" -keystore "path/to/client.truststore.ks"
   -keypass password -storepass password
   ```

   The keytool utility prompts you about whether to import the certificate to your keystore:

   ```
   Owner: CN="mycompany.somewhere.com OU=IBM SWG", O=IBM, C=US
   Issuer: CN="mycompany.somewhere.com OU=IBM SWG", O=IBM, C=US
   Serial number: 49f91548
   Valid from: 4/29/09 8:04 PM until: 7/28/09 8:04 PM
   Certificate fingerprints:
            MD5:  72:B3:26:F8:34:94:14:F3:30:4F:AD:6B:8C:D9:EE:FF
            SHA1: E5:AD:B0:D5:EB:B8:E9:57:DB:FA:D2:CD:4D:C5:43:52:58:B2:85:77
   Trust this certificate? [no]:
   ```

3. Enter yes. The following result displays:

   ```
   Certificate was added to keystore
   [Saving path/to/client.truststore.ks ]
   ```

Creating a client keystore and exporting the public key as a certificate:

Create a keystore for the client and export the public key as a client certificate that SOAP Gateway can use to verify that the client is trusted.

Client authentication requires server authentication. You must have completed the following steps first:

- Create a keystore for SOAP Gateway that contains a private and public key pair and export it as a server certificate
- Create a client truststore, and import the SOAP Gateway server certificate.

To create a Java keystore on the client and export the public key:

1. Create a keystore by using a Key management tool such as Ikeyman or Keytool. In a command console, enter the following command:

   ```
   keytool -genkey -alias client.keystore -dname
   "CN=SOAP Gateway Client Keystore OU=IBM SWG, O=IBM, C=US"
   -keyalg RSA -keypass password -storepass password
   -keystore "/path/to/cient.keystore.ks"
   ```

2. Export the public key from the client keystore (client.keystore.ks) as a certificate.

   ```
   keytool -export -alias client.keystore -storepass password
   -file "/path/to/client.keystore.cer"
   -keystore "/path/to/client.keystore.ks"
   ```

   Tip: You can have the certificate signed by a Certificate Authority (CA), such as VeriSign, or create your own CA by using software such as OpenSSL to sign your own (self-signed) certificate.

Creating the server truststore for SOAP Gateway:

Create a truststore for SOAP Gateway to store the HTTPS client certificates, or the SSL server certificate (from IMS Connect).

If the truststore is used to store the server certificate from IMS Connect, the Java keystore must have a valid X.509 certificate from IMS Connect for authentication.

To provide a Java keystore on the client:

Create a truststore:

```
keytool -genkey -alias server.truststore -dname
"CN=SOAP Gateway Keystore OU=IBM SWG, O=IBM, C=US"
-keyalg RSA -keypass password -storepass password
-keystore "/path/to/server.truststore.ks"
```

For NIST SP800-131a, specify SHA256withRSA for the signature algorithm and 2048 for the key size.

```
keytool -genkey -alias server.truststore -dname
"CN=SOAP Gateway Keystore OU=IBM SWG, O=IBM, C=US"
-keyalg RSA -sigalg SHA256withRSA  -keysize 2048
-keypass password -storepass password
-keystore "/path/to/server.truststore.ks"
```

- If client authentication is required for HTTPS connections to SOAP Gateway, transfer and import the HTTPS client certificate into the SOAP Gateway truststore
- If server authentication is required for SSL connections with IMS Connect, export the IMS Connect certificate and import the IMS Connect server certificate to the SOAP Gateway truststore

Importing the client certificate into the SOAP Gateway truststore:

To enable SOAP Gateway to trust the requests from its client, you must import the client certificate into the SOAP Gateway truststore.

You must create a truststore for SOAP Gateway before importing the client certificate.

You must also transfer the client certificate to a location that is accessible to you when you import the certificate.

Import the client certificate (`client.keystore.cer` in this example). The following example shows how to use the keytool utility to import the client certificate into the SOAP Gateway truststore.

```
keytool -import -v -trustcacerts -alias client.keystore.cer
-file "/path/to/client.keystore.cer"
-keystore "/path/to/server.truststore.ks"
-keypass password -storepass password
```

Exporting the certificate from IMS Connect:

Use the RACDCERT command to export the certificate to a data set.

The certificate for IMS Connect is generated by using the RACDCERT GENCERT command. To export the certificate from IMS Connect:

1. Determine the certificate authority to use. You can use the DA option in the System Display and Search Facility (SDSF) to locate the IMS Connect that has been set up for SSL. Use the display output to determine the owner that is associated with the PROC.
2. From TSO, issue the RACDCERT command by using the associated owner of the PROC in the ID value. To issue the RACDCERT command, you must have sufficient authority for the specific RACDCERT function. For more information, see the topic on RACDCERT (Manage RACF digital certificates) in *z/OS V1R13 Cryptographic Services ICSF Administrator's Guide*.

   For NIST 800-131a, the RSA key must be generated with at least 2048 bits by specifying `SIZE(2048)` with the RACDCERT GENCERT command.
3. Create a data set to export the certificate authority CERTAUTH for IMS Connect. if the digital ring information is as follows:

```
Ring:
        >IMSConnKeyring<
   Certificate Label Name              Cert Owner     USAGE      DEFAULT
   ------------------------------      ------------   --------   -------
   IMS Connect Certauth                CERTAUTH       CERTAUTH     NO
   IMS Connect User Cert               ID(IMSCONN)    PERSONAL     YES
```

   The corresponding TSO command to export the Certificate Authority CERTAUTH into the IMSCONN.CERTBIN data set is:

```
RACDCERT CERTAUTH EXPORT(LABEL('IMS Connect Certauth'))
DSN('IMSCONN.CERTBIN') FORMAT(CERTDER)
```

4. From TSO, copy the data set to the HFS system. The following example copies the data set to an imsconn.cer certificate.

```
OPUT 'IMSCONN.CERTBIN' '/u/userID/imsconn.cer' binary convert(no)
```

Related information:

Sample JCL for RACF-Managed SSL in IMS V13 Communications and Connections information
See this sample JCL for how to use the RACF RACDCERT command to set up keyrings and certificates for IMS Connect.

Sample JCL for RACF-Managed SSL in IMS V12 Communications and Connections information
See this sample JCL for how to use the RACF RACDCERT command to set up keyrings and certificates for IMS Connect.

Importing IMS Connect server certificate into the SOAP Gateway truststore:

To enable SOAP Gateway to trust the SSL server, IMS Connect, you must import the IMS Connect certificate into the SOAP Gateway truststore.

Import the IMS Connect certificate (`imsconn.cer` in this example). The following example shows how to use the keytool utility to import an IMS Connect certificate into the SOAP Gateway truststore.

```
keytool -import -v -trustcacerts -alias icon
-file "/path/to/imsconn.cer"
-keystore "/path/to/server.truststore.ks"
-keypass password -storepass password
```

Support for web services security (WS-Security)

Web services security support provides dynamic authentication of users on a per-message basis.

When WS-Security is enabled for a web service, SOAP Gateway extracts the user information from the WS-Security header and propagates the information to IMS.

When WS-Security is not enabled, user ID and password information is provided by the connection bundle on a per-web service basis.

You can enable the WS-Security feature by using the SOAP Gateway management utility. The information that you specify in the SOAP Gateway management utility overrides the WS-Security setting in the WSDL file that is generated by Rational Developer for System z.

Dynamic authentication compared to static authentication

When WS-Security is enabled for a web service, user identity is dynamically determined at run time on a per-message basis. When WS-Security is not enabled for a web service, the user identity information is specified in the web service connection bundle. User information is statically defined at deployment time on a per-web service basis. All clients that access the web service use the same user identity and password to be authenticated by IMS Connect.

WS-Security through secured transport

SOAP Gateway accepts only secured HTTPS requests when WS-Security is enabled for a web service. Because the user information is in clear text, you must configure Secure Sockets Layer (SSL) security on both the web service client and the server to provide data encryption over TCP/IP.

SOAP Gateway supports security tokens in the WS-Security header as part of the SOAP request message. SOAP Gateway extracts the user identity information (User ID) and sends it over to IMS Connect, which propagates the information to IMS. The WS-Security token is also accessible by custom authentication modules, if such a module is enabled.

UsernameToken Profile Version 1.0

For a user name security token, a SOAP Gateway client must provide the user ID and password to SOAP Gateway in the SOAP message security header. SOAP Gateway extracts this user ID and password from the SOAP message and passes them with the payload data to IMS Connect for authentication.

Server authentication or client authentication is recommended. Otherwise the user ID and passsword are transmitted in clear text.

Security Assertion Markup Language (SAML) Version 1.1 and Version 2.0 sender-vouches tokens

SAML is an XML-based OASIS standard for exchanging user identity and security attribute information. The sender-vouches confirmation method is used when a server (SOAP Gateway) needs to further propagate the client identity and attributes on behalf of the client (to IMS Connect and OTMA). An attesting entity uses the sender-vouches confirmation method to assert that it is acting on behalf of the subject of the SAML statements attributed with the sender-vouches SubjectConfirmation element.

SAML support requires an SSL connection with client authentication to enable sender-vouches security tokens. You must configure client authentication to use the SAML sender-vouches confirmation method. The SOAP response message does not carry any security token information.

A SAML assertion is a sender-vouches assertion if it includes the sender-vouches <saml:ConfirmationMethod> element, as defined in the OASIS Web Services Security specifications. A SAML Version 1.1 sender-vouches assertion must contain the following SubjectConfirmation element:

```
<saml:SubjectConfirmation>
    <saml:ConfirmationMethod>
    urn:oasis:names:tc:SAML:1.0:cm:sender-vouches
   </saml:ConfirmationMethod>
</saml:SubjectConfirmation>
```

A SAML Version 2.0 sender-vouches assertion must contain the following SubjectConfirmation element:

```
<saml:SubjectConfirmation
    Method="urn:oasis:names:tc:SAML:2.0:cm:sender-vouches">
</saml:SubjectConfirmation>
```

A SAML token can be signed or unsigned:

- When the token is unsigned, the request contains a minimal sender-vouches SAML assertion with no optional elements included. There are no signatures or certificates required. The response does not contain a security header.
- When the token is signed, the request contains a sender-vouches SAML assertion. The assertion element is signed. A reference to the certificate used to verify the signature is provided in the header. The response does not contain a security header.

Related tasks:

"Enabling WS-Security" on page 158
To enable WS-Security to propagate authentication information with a request to access a deployed web service on the SOAP Gateway, you must have a policy set and a binding file for the server.

Related information:

Samples on enabling WS-Security for a web service
For step-by-step instructions on how to enable an IMS application as a web service with WS-Security enabled, see the IMS Enterprise Suite SOAP Gateway website.

Signed SAML tokens:

A SAML token can be signed by a Security Token Service (STS) or self-issued. SOAP Gateway can be configured to trust the SAML token and the signature (certificate) embedded, or to use the certificates in a specified truststore to verify the signature before trusting the SAML token.

The following figure shows a web service request message that has a signed SAML token. The security information is embedded in the header of the message. When a SAML token is signed, the signature and authentication statement is embedded in the `saml:Assertion` section of the security information.

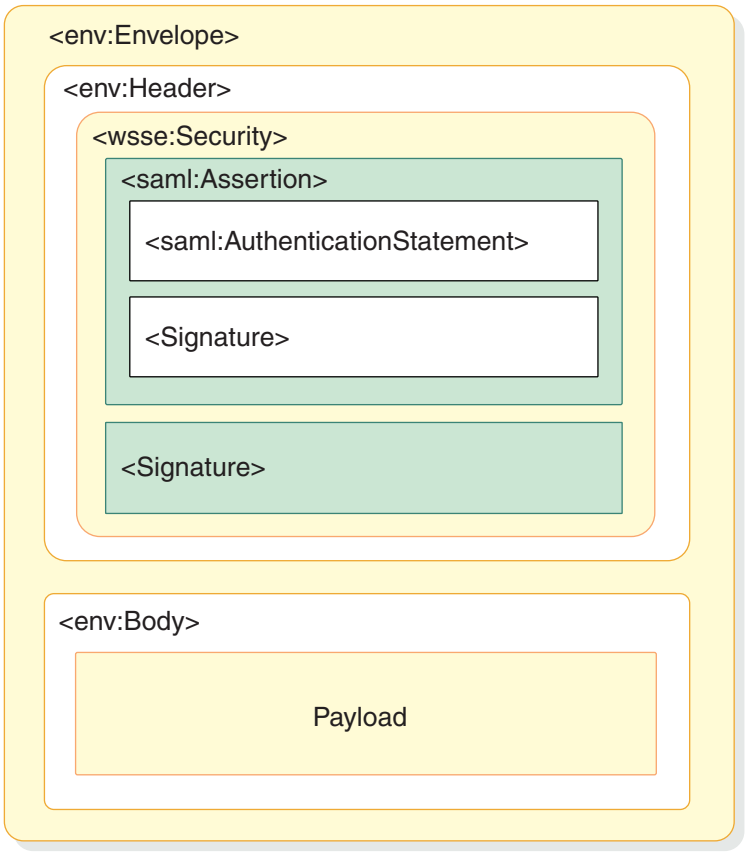


Figure 29. WS-Security header for a signed SAML token in a SOAP message

When a SAML token is signed by the sender, you can configure the SOAP Gateway server to:

- Trust the embedded signature (certificate) within the SOAP header along with the signing SAML token, or
- Trust the certificates in a specified truststore. All certificates in the referenced keystore or truststore are the trusted source for verifying the SAML signature.

After the SAML signature is verified and the token is trusted, SOAP Gateway extracts the SAML ID together with the security attributes from the SOAP header and propagates the SAML ID to IMS Connect for further authorization.

SOAP Gateway includes a WS-Security API for creating self-issued SAML tokens. You can also use any RSA-SHA1 signature method.

Related concepts:

"Signed SAML tokens" on page 155
A SAML token can be signed by a Security Token Service (STS) or self-issued. SOAP Gateway can be configured to trust the SAML token and the signature (certificate) embedded, or to use the certificates in a specified truststore to verify the signature before trusting the SAML token.

Related tasks:

"Creating self-issued signed SAML tokens" on page 160
Use the WS-Security API included in SOAP Gateway to create self-issued signed SAML tokens.

Related reference:

Refer to the IBM WebSphere Application Server Version 7 information center for the SAMLToken API Javadoc information.
For more detail about the SAMLToken API, see the Javadoc information in the IBM WebSphere Application Server Version 7 information center.

Authentication and authorization of the user ID:

When WS-Security is enabled, SOAP Gateway passes user information in the security token to IMS Connect for authentication. IMS Connect then passes the user ID to OTMA, which then authorizes the user access to transactions or OTMA commands.

When WS-Security is disabled, SOAP Gateway passes the user ID and password information in the connection bundle for the web service to IMS Connect.

When WS-Security is enabled, depending on the token type, SOAP Gateway passes the following information to IMS Connect:

- For user name tokens, SOAP Gateway passes the user ID, the password, and the payload data.
- For SAML tokens, SOAP Gateway passes the user ID and the payload data.

When user name tokens are used, if IMS Connect security is enabled (RACF=Y), IMS Connect authenticates the user ID and password. IMS Connect then passes the user ID and data to OTMA. IMS Connect does not pass any password information to OTMA.

If IMS Connect security is disabled (RACF=N), IMS Connect simply passes on the user ID and data to OTMA without authentication.

When SAML tokens are used, IMS Connect security must be turned off because SAML tokens do not contain password information. If IMS Connect security is turned on, the authentication would fail.

If OTMA security is enabled (OTMASE={CHECK | PROFILE | FULL}), OTMA authorizes the user to access transactions or OTMA commands. If OTMA security is set to OTMASE=NONE, then no authorization check is performed.

WS-Security policy sets and binding files:

Policy sets and binding files are attached to a web service to specify how to secure the SOAP message requests from the client to the web service provider and the response to the client.

Policy sets are assertions about how services are defined. Each policy set specifies the protection that will be applied, such as what security tokens to send, what message parts to sign or encrypt, and what token type and algorithms to use for the encryption.

The binding files specify lower-level details of how to secure the SOAP messages that pass between service requesters (SOAP Gateway clients) and SOAP Gateway, and between SOAP Gateway and the target web services (IMS applications). Binding files provide the information that the runtime environment needs to implement the WS-Security configuration.

Each token type has a specific web services security policy set and binding files for the SOAP Gateway server. They are attached to the web service and SOAP Gateway uses the information in these files to handle the web service requests and responses. Do not alter the provided server policy set and binding files.

Enabling WS-Security:

To enable WS-Security to propagate authentication information with a request to access a deployed web service on the SOAP Gateway, you must have a policy set and a binding file for the server.

Server policy set and binding files are provided per web service token type. Do not alter the provided server policy set or binding files.

Deploying a WS-Security enabled web service involves the following tasks and decisions:

1. When you generate the web service artifact files in Rational Developer for System z, in the field where you specify service location for WSDL properties in the form of *protocol*://server:*port/path/to/web_service*, use HTTPS as the protocol, with the correct secure port number.

   If you are using the z/OS Communications Server V1R11 AT-TLS feature with username tokens for authentication, use HTTP as the protocol.

   Tip: In the **SOAP Gateway correlator file** tab, you can specify whether to enable WS-Security. However, SOAP Gateway ignores this value in the correlator. You must use the SOAP Gateway management utility to specify the WS-Security token type. When a token type is specified, WS-Security is enabled.
2. In SOAP Gateway management utility, specify the **-t** parameter in the iogmgmt -deploy command to set the token type. For example, to set the type to user name token:

   ```
   iogmgmt -deploy -w MyWSDL.wsdl -r MyCorrelator.xml -t UserNameToken
   ```

   To set the type to SAML 1.1 unsigned token:

   ```
   iogmgmt -deploy -w MyWSDL.wsdl -r MyCorrelator.xml -t SAML11Token
   ```

   To set the type to SAML 1.1 signed token and to trust the embedded signature (certificate) within the soap header from any signer and the embedded SAML token:

   ```
   iogmgmt -deploy -w MyWSDL.wsdl -r MyCorrelator.xml -t SAML11SignedTokenTrustAny
   ```

   To set the type to SAML 2.0 signed token and to use the certificates inside the specified truststore to verify the SAML 2.0 signature:

   ```
   iogmgmt -deploy -w MyWSDL.wsdl -r MyCorrelator.xml -t SAML20SignedTokenTrustOne
   -y "JCEKS" –p "storepass" –h "c:/cert/recipient.jceks"
   ```

   where:

   - **-y** specifies the truststore type
   - **-p** specifies the truststore password
   - **-h** specifies the full path to the truststore

   The SOAP Gateway management utility will update the server binding file accordingly.
3. If you are using the z/OS Communications Server V1R11 AT-TLS feature, skip this step. Otherwise, you must configure the SOAP Gateway server for HTTPS requests.

For Java keystore (JKS), for username tokens, a minimum of server authentication is required. For SAML tokens, client authentication is required.

First, enable server authentication by exporting the server certificates to the client system.

a. Create a keystore for the SOAP Gateway server that contains a key pair.
b. Export the public key from the server keystore as a certificate.
c. Create a truststore on the client system.
d. Import the server certificate into the truststore of the client.

The detailed steps to create a keystore for the SOAP Gateway server, export it as a server certificate, and import it into the truststore of the client system are described in the "Configuring SSL and HTTPS support with Java keystore (JKS)" on page 148 section.

4. If you are using the z/OS Communications Server V1R11 AT-TLS feature, skip this step. If you are using JKS, for username tokens, client authentication is optional. For SAML tokens, client authentication is required.

   Enable client authentication by exporting the client certificates to the SOAP Gateway server. .

   a. Create a keystore for the client system that contains a key pair.
   b. Export the public key from the client keystore as a certificate.
   c. Create a truststore for the SOAP Gateway server.
   d. Import the client certificate into the server truststore.

   The detailed steps to create a keystore for the client system, export it as a client certificate, and import it into the SOAP Gateway server truststore are described in the "Configuring SSL and HTTPS support with Java keystore (JKS)" on page 148 section.

5. Decide how to collect user ID and password information and pass it to SOAP Gateway. SOAP Gateway sends this information as a user name token to the web service, to be authenticated by the host IMS system.

   SOAP Gateway provides a custom authentication module plug-in mechanism. The custom authentication module passes this information as a user name token to the web service. Based on your business needs, you must decide how to collect the user name token and pass it to SOAP Gateway.

6. If you are using the z/OS Communications Server V1R11 AT-TLS feature, skip this step. Otherwise, specify the HTTPS port number, HTTPS truststore name and password, and HTTPS keystore name and password.

   To specify the HTTPS port number, HTTPS truststore name, HTTPS truststore password, HTTPS keystore name, and HTTPS keystore password, use the SOAP Gateway management utility iogmgmt -prop command.

Related tasks:

Configuring SSL and HTTPS support with Java keystore (JKS)
To use HTTPS between a SOAP Gateway client and SOAP Gateway, or SSL between SOAP Gateway and its server (IMS Connect), you must create the keystore and truststore, and configure the SOAP Gateway server.

Related reference:

WS-Security related samples
See the IMS Exchange website for WS-Security related samples.

OASIS Web Services Security: SAML Token Profile 2.0 specifications
For more information, see the OASIS Web Services Security: SAML Token Profile 2.0 specifications.

OASIS Web Services Security: SAML Token Profile 1.1 specifications
For more information, see the OASIS Web Services Security: SAML Token Profile 1.1 specifications.

OASIS Web Services Security: UsernameToken Profile 1.0 Specifications
For more information, see the OASIS Web Services Security 3 UsernameToken Profile 1.0 Specifications.

"-deploy: Deploy a web service or callout application" on page 444
The -deploy command deploys a web service, callout application, or business event application to the active configuration of the SOAP Gateway server.

Creating self-issued signed SAML tokens:

Use the WS-Security API included in SOAP Gateway to create self-issued signed SAML tokens.

The SAMLTokenFactory class is used for the creation of SAML security tokens.

1. Instantiate a token factory based on the version level of the token. The following example creates a SAML 1.1 token.

```
SAMLTokenFactory samlFactory =
SAMLTokenFactory.getInstance(SAMLTokenFactory.WssSamlV11Token11);
```

2. Create a RequesterConfig object. The default configuration of the AssertionSignatureRequired property (`true`) requires a signed assertion. You can uncomment the second line of the following example to use unsigned tokens.

```
RequesterConfig reqData = samlFactory.newSenderVouchesTokenGenerateConfig();
// reqData.setAssertionSignatureRequired(false);
```

3. Create a CredentialConfig object.

```
CredentialConfig cred = samlFactory.newCredentialConfig();
cred.setRequesterNameID("Alice");
//SAML attributes:
SAMLAttribute sattribute = new SAMLAttribute("Address" /* Name*/, new String[]
 {"123 SAML street, Austin"}
/*Attribute Values*/,null,"IBM WebSphere namespace" /* Namespace*/, null
/* format*/,  null /*Fridendly name */);
   ArrayList<SAMLAttribute> al = new ArrayList<SAMLAttribute>();
   al.add(sattribute);
   sattribute = new SAMLAttribute("Groups", new String[] {"admin users",
"Building ABC", "Reporting to Joe"}, null, null , null, null  );
   al.add(sattribute);
   cred.setSAMLAttributes(al);
```

4. Create a ProviderConfig object which specifies key store for SAML signing and encryption, expiration time, and issuer logical name

```
ProviderConfig samlIssuerCfg = samlFactory.newDefaultProviderConfig(null);
//get or create an instance of ProviderConfig
SAMLToken saml = samlFactory.newSAMLToken(cred, reqData, samlIssuerCfg);
```

Related concepts:

"Signed SAML tokens" on page 155
A SAML token can be signed by a Security Token Service (STS) or self-issued. SOAP Gateway can be configured to trust the SAML token and the signature (certificate) embedded, or to use the certificates in a specified truststore to verify the signature before trusting the SAML token.

Related reference:

Refer to the IBM WebSphere Application Server Version 8 information center for the SAMLToken API Javadoc information.
For more detail about the SAMLToken API, see the Javadoc information in the IBM WebSphere Application Server Version 8 information center.

Direct SOAP messages for passing the user name token:

You can pass a direct SOAP message to pass the user name token with the web service request to SOAP Gateway by coding the message with the SOAP header in your client application.

The following sample demonstrates how the SOAP header is coded in the client application, with the user name and password information.

```
// DISCLAIMER OF WARRANTIES.  The following code is
// sample code created by IBM Corporation.  This sample code is
// not part of any standard or IBM product and is provided to you
// solely for the purpose of assisting you in the development of
// your applications.  The code is provided "AS IS", without
// warranty of any kind.  IBM shall not be liable for any damages
// arising out of your use of the sample code, even if they have
// been advised of the possibility of such damages.

import org.apache.commons.httpclient.HttpClient;
import org.apache.commons.httpclient.methods.PostMethod;
import org.apache.commons.httpclient.params.HttpMethodParams;

import java.io.*;
import java.security.Security;
import javax.net.ssl.HttpsURLConnection.*;
import javax.net.ssl.*;
import java.net.*;

import java.io.BufferedReader;

public class Sample {

  static final String url = "https://localhost:8443/imssoap/services/IMSPHBKService";
  static final String urn = "IMSPHBK";
  static String result = "";


  final static String soapmsg = "<?xml version=\"1.0\" encoding=\"utf-8\"?>" +
     "<soapenv:Envelope xmlns:q0=\"http://www.IMSPHBKI.com/schemas/IMSPHBKIInterface\""+
     " xmlns:soapenv=\"http://schemas.xmlsoap.org/soap/envelope/\"" +
     " xmlns:xsd=\"http://www.w3.org/2001/XMLSchema\""+
     " xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\">" +
     " <soapenv:Body>" +
     " <q0:INPUTMSG>" +
     " <q0:in_ll>0</q0:in_ll>" +
     " <q0:in_zz>22</q0:in_zz>" +
     " <q0:in_trcd>ivtno</q0:in_trcd>" +
     " <q0:in_cmd>display</q0:in_cmd>" +
     " <q0:in_name1>LAST1</q0:in_name1>" +
     " <q0:in_name2/>" +
     " <q0:in_extn/>" +
     " <q0:in_zip/>" +
     " </q0:INPUTMSG>" +
     " </soapenv:Body>" +
     " </soapenv:Envelope>";

   final static String soapmsg_withST = "<?xml version=\"1.0\" encoding=\"utf-8\"?>" +
     "<soapenv:Envelope xmlns:q0=\"http://www.IMSPHBKI.com/schemas/IMSPHBKIInterface\"" +
       " xmlns:soapenv=\"http://schemas.xmlsoap.org/soap/envelope/\"" +
       " xmlns:xsd=\"http://www.w3.org/2001/XMLSchema\" " +
       " xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\">" +
     "<soapenv:Header xmlns:wsa=\"http://www.w3.org/2005/08/addressing\">" +
     "<wsse:Security soapenv:mustUnderstand=\"1\""+
```

```
      " xmlns:wsse=\"http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-
secext-1.0.xsd\">" +
     "<wsse:UsernameToken wsu:Id=\"user1\"" +
       " xmlns:wsu=\"http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-
utility-1.0.xsd\">" +
     "<wsse:Username>user01</wsse:Username><wsse:Password" +
       " Type=\"http://docs.oasis-open.org/wss/2004/01/" +
       "oasis-200401-wss-username-token-profile-1.0#PasswordText\">" +
       " password</wsse:Password></wsse:UsernameToken></wsse:Security><wsa:To>" +
       " http://9.30.20.178:8080/imssoap/services/IMSPHBKService</wsa:To>" +
     "<wsa:MessageID>urn:uuid:89C2332A42CF7FA1671232146960443</wsa:MessageID>" +
     "<wsa:Action>urn:IMSPHBK</wsa:Action></soapenv:Header>" +
     " <soapenv:Body><q0:INPUTMSG>" +
     " <q0:in_ll>0</q0:in_ll>" +
     " <q0:in_zz>22</q0:in_zz>" +
     " <q0:in_trcd>ivtno</q0:in_trcd>" +
     " <q0:in_cmd>display</q0:in_cmd>" +
     " <q0:in_name1>LAST1</q0:in_name1>" +
     " <q0:in_name2/>" +
     " <q0:in_extn/>" +
     " <q0:in_zip/>" +
     " </q0:INPUTMSG>" +
     " </soapenv:Body>" +
     " </soapenv:Envelope>";

   public static void main(String[] args)
  {
     Sample thisSample = new Sample();
     HttpsURLConnection conn = null;
     try{
        System.setProperty("javax.net.ssl.trustStore", +
          "C:\\workImplement\\SSL\\RSA\\client.truststore.ks");
        System.setProperty("javax.net.ssl.trustStorePassword", "imssoap");
        Security.addProvider(new com.ibm.jsse.IBMJSSEProvider());
     }
     catch(Exception e)
     {
        e.printStackTrace();
     }

     HostnameVerifier hv = new HostnameVerifier() {
        public boolean verify(String urlHostName, SSLSession session) {
           return true;
        }
     };
     try
     {
        HttpsURLConnection.setDefaultHostnameVerifier(hv);
        URL thisUrl = new URL (url);
        conn = (HttpsURLConnection)thisUrl.openConnection();
        System.out.println("Opened connection");
  }
  catch(Exception ex)
  {
        ex.printStackTrace();
  }
  try {
        String type = "application/soap+xml; charset=utf-8";
        String soapAction = "\"urn:" + urn + "\"";

        conn.setRequestProperty("Content-Length",
String.valueOf(soapmsg_withST.length()));
        conn.setRequestProperty("Content-Type", "text/xml; charset=utf-8");
        conn.setRequestProperty("SOAPAction", soapAction);
        conn.setRequestMethod("POST");
        conn.setDoOutput(true);
        conn.setDoInput(true);
        OutputStream out = conn.getOutputStream();
        out.write(soapmsg_withST.getBytes());
        out.close();
```

```
        System.out.println(conn.getResponseMessage());
        System.out.println(conn.getResponseCode());

        if(conn.getResponseMessage().equalsIgnoreCase("OK"))
           thisSample.writeResult(conn.getInputStream());
        else
           thisSample.writeResult(conn.getErrorStream());

      } catch(Exception e){
         e.printStackTrace(System.err);

      } finally {
         conn.disconnect();
      }
   }

   public void writeResult(InputStream stream) throws IOException {

     StringBuffer sb = new StringBuffer("");

     InputStreamReader isr = new InputStreamReader(stream);
     BufferedReader in = new BufferedReader(isr);

     String inputLine;

     while ((inputLine = in.readLine()) != null) {
        sb.append(inputLine);
        System.out.println(inputLine);
     }

     in.close();
     isr.close();

   }


}
```

Direct SOAP messages for passing the SAML token:

You can pass a direct SOAP message to pass the SAML token with the web service request to SOAP Gateway by coding the message with the SOAP header in your client application.

The following sample demonstrates how the SOAP header is coded in the client application, with the user information and message timestamp

```
// DISCLAIMER OF WARRANTIES.  The following code is
// sample code created by IBM Corporation.  This sample code is
// not part of any standard or IBM product and is provided to you
// solely for the purpose of assisting you in the development of
// your applications.  The code is provided "AS IS", without
// warranty of any kind.  IBM shall not be liable for any damages
// arising out of your use of the sample code, even if they have
// been advised of the possibility of such damages.

import org.apache.commons.httpclient.HttpClient;
import org.apache.commons.httpclient.methods.PostMethod;
import org.apache.commons.httpclient.params.HttpMethodParams;
import java.io.*;
import java.text.*;
import java.util.Calendar;
import java.security.Security;
import javax.net.ssl.HttpsURLConnection.*;
import javax.net.ssl.*;
import java.net.*;
import java.io.BufferedReader;

public class IMSPHBK_SAML {

  static final String url = "https://localhost:8443/imssoap/services/IMSPHBKService";
  static final String urn = "IMSPHBK";
```

```
  static String result = "";

  final static String soapmsg = "<?xml version=\"1.0\" encoding=\"utf-8\"?>" +
     "<soapenv:Envelope xmlns:q0=\"http://www.IMSPHBKI.com/schemas/IMSPHBKIInterface\""+
     " xmlns:soapenv=\"http://schemas.xmlsoap.org/soap/envelope/\"" +
     " xmlns:xsd=\"http://www.w3.org/2001/XMLSchema\""+
     " xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\">" +
     " <soapenv:Body>" +
     " <q0:INPUTMSG>" +
     " <q0:in_ll>0</q0:in_ll>" +
     " <q0:in_zz>22</q0:in_zz>" +
     " <q0:in_trcd>ivtno</q0:in_trcd>" +
     " <q0:in_cmd>display</q0:in_cmd>" +
     " <q0:in_name1>LAST1</q0:in_name1>" +
     " <q0:in_name2/>" +
     " <q0:in_extn/>" +
     " <q0:in_zip/>" +
     " </q0:INPUTMSG>" +
     " </soapenv:Body>" +
     " </soapenv:Envelope>";


  final static String soapmsg_withSAML = "<?xml version=\"1.0\" encoding=\"utf-8\"?> " +
         "<soapenv:Envelope xmlns:soapenv=\"http://schemas.xmlsoap.org/soap/envelope/\">" +
         "<soapenv:Header xmlns:wsa=\"http://www.w3.org/2005/08/addressing\">" +
         "<wsse:Security
             xmlns:wsse=\"http://docs.oasis-open.org/wss/2004/01/
oasis-200401-wss-wssecurity-secext-1.0.xsd\""+
         " soapenv:mustUnderstand=\"1\">" +
         "<wsu:Timestamp
             xmlns:wsu=\"http://docs.oasis-open.org/wss/2004/01/
oasis-200401-wss-wssecurity-utility-1.0.xsd\">" +
         "<wsu:Created>" + getActualRunDateIso() + "</wsu:Created></wsu:Timestamp>" +
         "<Assertion xmlns=\"urn:oasis:names:tc:SAML:1.0:assertion\"
           xmlns:samlp=\"urn:oasis:names:tc:SAML:1.0:protocol\"" +
         " xmlns:saml=\"urn:oasis:names:tc:SAML:1.0:assertion\" "+
         "AssertionID=\"ff96d003a87aecbed4c130d9e6db9860\"
            IssueInstant=\"2010-05-12T17:23:17.704Z\" " +
         " Issuer=\"http://www.yourcompanyname.com\" MajorVersion=\"1\" MinorVersion=\"1\">"+
         "<Conditions NotBefore=\"2010-05-11T17:23:17.187Z\"
         NotOnOrAfter=\"2011-03-16T20:08:17.187Z\"></Conditions>" +
         "<AuthenticationStatement AuthenticationInstant=\"2010-05-12T17:23:17.187Z\"
         AuthenticationMethod=\"urn:oasis:names:tc:SAML:1.0:am:unspecified\">" +
         "<Subject>" +
         "<NameIdentifier
             Format=\"urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified\">A202545
         </NameIdentifier>"+ "<SubjectConfirmation>" +
         "<ConfirmationMethod>urn:oasis:names:tc:SAML:1.0:cm:sender-vouches
         </ConfirmationMethod>" +
         "</SubjectConfirmation></Subject> </AuthenticationStatement> <AttributeStatement>" +
         "<Subject>"+
         "<NameIdentifier
           Format=\"urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified\">A202545
         </NameIdentifier>" +
         "<SubjectConfirmation>" +
         "<ConfirmationMethod>urn:oasis:names:tc:SAML:1.0:cm:sender-vouches
         </ConfirmationMethod>" +
         "</SubjectConfirmation>" +
         "</Subject>" +
         "<Attribute xmlns:xsd=\"http://www.w3.org/2001/XMLSchema\" " +
         " xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\"
           AttributeName=\"Groups\""+
         " AttributeNamespace=\"urn:bea:security:saml:groups\">" +
         "<AttributeValue>COMPANY_TEST_PROTECTIONDOMAIN</AttributeValue>" +
         "<AttributeValue>COMPANY_GROUP</AttributeValue>" +
         "<AttributeValue>COMPANY_ATN_X509_HTTPS</AttributeValue>" +
         "<AttributeValue>COMPANY_JAP50_WS_SAML11_ConsumerServer_SERVER</AttributeValue>" +
          "</Attribute></AttributeStatement>  </Assertion></wsse:Security>" +
         "<wsa:To>https://localhost:8443/imssoap/services/IMSPHBKService</wsa:To>" +
           "<wsa:MessageID>urn:uuid:FA35D21EB422BCB3701273603365088</wsa:MessageID>" +
         "<wsa:Action>urn:IMSPHBK</wsa:Action></soapenv:Header><soapenv:Body>" +
         "<INPUTMSG xmlns=\"http://www.IMSPHBKI.com/schemas/IMSPHBKIInterface\">" +
         "<in_ll>32</in_ll>" +
         "<in_zz>0</in_zz>" +
         "<in_trcd>ivtno</in_trcd>" +
       "<in_cmd>DISPLAY</in_cmd>" +
       "<in_name1>LAST1</in_name1>" +
         "<in_name2></in_name2>" +
         "<in_extn></in_extn>" +
         "<in_zip></in_zip>" +
```

```
        "</INPUTMSG></soapenv:Body></soapenv:Envelope>";


 public static String getActualRunDateIso() {
  DateFormat dateFormat = new SimpleDateFormat("yyyy-MM-dd'T'HH:mm:ss'Z'");
  Calendar c  = Calendar.getInstance();
  c.add(Calendar.HOUR_OF_DAY, 7);
  c.getTime();

  return dateFormat.format(c.getTime()).toString();
 }
   public static void main(String[] args)
  {
    IMSPHBK_SAML thisSample = new IMSPHBK_SAML();
    HttpsURLConnection conn = null;
    try{
      System.setProperty("javax.net.ssl.trustStore",
"C:\\workimplement\\ssl\\mutual\\client.truststore.ks");
      System.setProperty("javax.net.ssl.trustStorePassword", "imssoap");
      System.setProperty("javax.net.ssl.keyStore",
"C:\\workimplement\\ssl\\mutual\\client.keystore.ks");
      System.setProperty("javax.net.ssl.keyStorePassword", "imssoap");
      System.setProperty("javax.net.ssl.keyStoreType","JKS");
      System.setProperty("javax.net.ssl.trustStoreType","JKS");
      Security.addProvider(new com.ibm.jsse.IBMJSSEProvider());
     }
     catch(Exception e)
     {
        e.printStackTrace();
     }

     HostnameVerifier hv = new HostnameVerifier() {
        public boolean verify(String urlHostName, SSLSession session) {
           return true;
        }
     };
     try
     {
        HttpsURLConnection.setDefaultHostnameVerifier(hv);
        URL thisUrl = new URL (url);
        conn = (HttpsURLConnection)thisUrl.openConnection();
        System.out.println("Opened connection");
  }
  catch(Exception ex)
  {
        ex.printStackTrace();
  }
  try {
        String type = "application/soap+xml; charset=utf-8";
        String soapAction = "\"urn:" + urn + "\"";

        conn.setRequestProperty("Content-Length", String.valueOf(soapmsg_withSAML.length()));
        conn.setRequestProperty("Content-Type", "text/xml; charset=utf-8");
        conn.setRequestProperty("SOAPAction", soapAction);
        conn.setRequestMethod("POST");
        conn.setDoOutput(true);
        conn.setDoInput(true);
        OutputStream out = conn.getOutputStream();
        out.write(soapmsg_withSAML.getBytes());
        out.close();

        System.out.println(conn.getResponseMessage());
        System.out.println(conn.getResponseCode());

        if(conn.getResponseMessage().equalsIgnoreCase("OK"))
           thisSample.writeResult(conn.getInputStream());
        else
           thisSample.writeResult(conn.getErrorStream());

     } catch(Exception e){
        e.printStackTrace(System.err);

     } finally {
        conn.disconnect();
     }
   }

   public void writeResult(InputStream stream) throws IOException {

     StringBuffer sb = new StringBuffer("");
```

```
     InputStreamReader isr = new InputStreamReader(stream);
     BufferedReader in = new BufferedReader(isr);

     String inputLine;

     while ((inputLine = in.readLine()) != null) {
        sb.append(inputLine);
        System.out.println(inputLine);
     }
     in.close();
     isr.close();
   }
}
```

Related tasks:

"Setting the timeout value for WS-Security enabled messages"
You can specify timestamp validation information for WS-Security tokens so that the SOAP Gateway sever can decide if the data has become stale and the message needs to be discarded.

Setting the timeout value for WS-Security enabled messages:

You can specify timestamp validation information for WS-Security tokens so that the SOAP Gateway sever can decide if the data has become stale and the message needs to be discarded.

Timestamp information for both SAML and user name tokens

For both SAML and user name tokens, you can specify timestamp validation information. You can use the WS-Security wsu:Timestamp XML element to define the creation and expiration time for the message. The wsu:Timestamp element has the following helper elements:

- wsu:Created: specifies the message creation time.
- wsu:Expires: specifies the message expiration time.

For example:

```
<wsu:Timestamp
xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd">
   <wsu:Created>2010-12-15T12:03:29Z</wsu:Created>
   <wsu:Expires>2010-12-15T12:04:29Z</wsu:Expires>
</wsu:Timestamp>
```

wsu:Expires contains the time when the message expires. If the current time is after the expiration time, the message is rejected. The creation time cannot be future time. SOAP Gateway would reject the message if it is not created within reasonable time frame. An example of an unreasonable timeframe is the message creation time is older than the current time, minus the timestamp maximum age.

For the wsu:Created element, you can specify the timestamp timeout value and its maximum age inside the `<securityInboundBindingConfig>` element in the server `binding.xml` file. The default timestamp age is 5 minutes (or 300 seconds). The maximum age cannot be greater than the timestamp timeout value. For example:

```
<securityInboundBindingConfig>
  <properties value="150" name="com.ibm.wsspi.wssecurity.core.TimestampMaxAge" />
  <properties value="300" name="com.ibm.wsspi.wssecurity.core.TimestampTimeout" />
<securityInboundBindingConfig>
```

If you want the timestamp maximum age to be larger than 300 seconds, you must first set the timestamp timeout value to a larger number.

In addition, for SAML tokens, you can specify an optional timeout condition for the token to expire. The SAML assertion named Conditions has two attributes, NotBefore and NotOnOrAfter.

- The NotBefore time cannot be a future time (must be before the current time). The assertion is not acceptable if NotBefore is after current time.
- The NotOnOrAfter time cannot be a time in the past (must be after the current time). The assertion is not acceptable if NotOnOrAfter is before the current time.

For example:

```
<Conditions NotBefore="2010-12-15T12:03:24.578Z" NotOnOrAfter="2010-12-15T12:05:24.578Z"/>
```

The checking and validation also takes clock skew into consideration to account for small clock drifts that naturally occur. The default clock skew is 3 minutes. You can specify the custom property clockSkew to adjust the maximum allowed clock skew. The clock skew should be a relatively small value compared with the timestamp values.

Setting the timeout value

1. Open the `bindings.xml` file for your web service, under *install_dir*/imssoap/WS-SECURITY/*token_type*/server.
2. Insert the following lines inside the `<securityInboundBindingConfig>` element:

```
<securityInboundBindingConfig>
  <properties value="number_of_seconds"
   name="com.ibm.wsspi.wssecurity.core.TimestampMaxAge" />
  <properties value="number_of_seconds"
   name="com.ibm.wsspi.wssecurity.core.TimestampTimeout" />
<securityInboundBindingConfig>
```

3. For SAML tokens only, you can set the clock skew value by adding the custom property clockSkew in the callbackHandler to adjust maximum allowed clock skew.

```
<callbackHandler
classname="com.ibm.websphere.wssecurity.callbackhandler.SAMLConsumerCallbackHandler">
  <properties value="false" name="signatureRequired" />
  <properties value="number_of_minutes" name="clockSkew" />
</callbackHandler>
```

The following `binding.xml` example shows a timestamp timeout of one hour (3600 seconds), a timestamp maximum age of 30 minutes (1800 seconds), and a clock skew of 10 minutes.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
  <securityBindings xmlns="http://www.ibm.com/xmlns/prod/websphere/200710/ws-securitybinding">
    <securityBinding name="application">
      <securityInboundBindingConfig>
        <properties value="1800" name="com.ibm.wsspi.wssecurity.core.TimestampMaxAge" />
        <properties value="3600" name="com.ibm.wsspi.wssecurity.core.TimestampTimeout" />
        <tokenConsumer classname="com.ibm.ws.wssecurity.wssapi.token.impl.CommonTokenConsumer"
         name="request:SAMLToken11SenderVouches">
          <valueType uri=""
           localName="http://docs.oasis-open.org/wss/oasis-wss-saml-token-profile-1.1#SAMLV1.1" />
          <securityTokenReference reference="request:SAMLToken11SenderVouches" />
          <jAASConfig configName="system.wss.consume.saml11" />
          <callbackHandler
           classname="com.ibm.websphere.wssecurity.callbackhandler.SAMLConsumerCallbackHandler">
            <properties value="false" name="signatureRequired" />
            <properties value="10" name="clockSkew" />
          </callbackHandler>
        </tokenConsumer>
      </securityInboundBindingConfig>
    </securityBinding>
  </securityBindings>
```

Custom authentication modules:

You can plug in your own authentication module to intercept an inbound message for the provider scenario to perform additional checks by using a Java Authentication and Authorization Service (JAAS) module.

Restriction: A web service must have WS-Security enabled before you can plug in a custom authentication module for that service.

JAAS authentication is performed in a pluggable fashion. New or updated technologies can be plugged in without requiring modifications to the client application itself. SOAP Gateway client applications can remain independent from underlying authentication technologies. An implementation for a particular authentication technology to be used is determined at run time. The implementation is specified in a login configuration file called the JAAS configuration file.

Custom JAAS security module

You can add a custom JAAS module by using a user name token or a SAML token. The custom JAAS security module could access the following objects or information when necessary:

- WS-Security token from the web service client
- Peer certificate in X509 format, if client authentication is used for SSL handshake
- SAML signed assertion certificate

The Distinguished Name (DN) and X509 certificate information is included in the WS-Security token. DN contents are defined as:

- CN = common name
- OU = organizational unit
- O = organization name
- L = locality name
- S = state name
- C = country name

This custom module is used to identify the user ID in the token to be authorized with the DN contents.

If the module is coded to reject a request by throwing an exception, SOAP Gateway generates an Axis fault with the thrown exception message. Otherwise, the request is assumed accepted, and SOAP Gateway continues to process the SOAP message. If for some reason the WS-Security token is null or missing in the SOAP request, SOAP Gateway would reject the request.

The JAAS configuration file is the `wsjaas.conf file` under the *`install_dir`*`/imssoap/WEB-INF` directory.

The following entries show how a user name token module and a SAML token module are configured in SOAP Gateway.

```
system.ims.soap.soapunt  {
    com.ibm.ims.soap.server.module.UNTConsumeLoginModule required;
};
...
```

```
system.wss.consume.saml11 {
    com.ibm.ims.soap.server.module.SAMLConsumeLoginModule  required;
};
...
system.wss.consume.saml20 {
    com.ibm.ims.soap.server.module.SAMLConsumeLoginModule  required;
};
```

These modules define which SOAP Gateway class intercepts and consumes a request with the user name or SAML token.

- The system.ims.soap.soapunt class is part of the server bindings entry for UsernameToken Profile requests. This mechanism lets you plug in custom authentication modules for UsernameToken Profile WS-Security requests. The UNTConsumeLoginModule module is a key module that processes and validates the UsernameToken Profile requests. It is required and cannot be removed.
- The system.wss.consume.saml11 class is part of the server bindings entry for SAML requests. This mechanism lets you plug in custom authentication modules for SAML 1.1 WS-Security requests. The SAMLConsumeLoginModule module is a key module that processes and validates the SAML V1.1 requests. It is required and cannot be removed.
- The system.wss.consume.saml20 class is part of the server bindings entry for SAML requests. This mechanism lets you plug in custom authentication modules for SAML 2.0 WS-Security requests. The SAMLConsumeLoginModule module is a key module that processes and validates the SAML V2.0 requests. It is required and cannot be removed.

To plug in your custom authentication module, add your module in the system.ims.soap.soapunt or system.wss.consume.saml20 entry, after the default module:

```
system.wss.consume.saml20 {
com.ibm.ims.soap.server.module.SAMLConsumeLoginModule required;
com.yourcompany.security.server.MySAMLLoginModule required;
};
```

The MySAMLLoginModule module is called after the standard SAMLConsumeLoginModule module.

Related reference:

Refer to the IBM WebSphere Application Server Version 8 information center for the SecurityToken API Javadoc information.
For more detail about the SecurityToken API, see the Javadoc information in the IBM WebSphere Application Server Version 8 information center.

Refer to the IBM WebSphere Application Server Version 8 information center for the SAMLToken API Javadoc information.
For more detail about the SAMLToken API, see the Javadoc information in the IBM WebSphere Application Server Version 8 information center.

Refer to the IBM WebSphere Application Server Version 8 information center for the UsernameToken API Javadoc information.
For more detail about the UsernameToken API, see the Javadoc information in the IBM WebSphere Application Server Version 8 information center.

Related information:

Custom authentication module samples on the IMS Exchange website
Download the SOAP Gateway custom authentication module samples from the

IMS Exchange website.

Plugging in a custom authentication module:

Append your custom authentication module to the default authentication module in the corresponding UsernameToken or SAML entries in the `wsjaas.conf` file in the SOAP Gateway installation.

To add a custom authentication module entry in the JAAS configuration file to intercept UsernameToken Profile or SAML requests, your custom entry must be added after the corresponding SOAP Gateway login module.

```
system.ims.soap.soapunt  {
    com.ibm.ims.soap.server.module.UNTConsumeLoginModule required;
};
...
system.wss.consume.saml11 {
com.ibm.ims.soap.server.module.SAMLConsumeLoginModule  required;
};
...
system.wss.consume.saml20 {
com.ibm.ims.soap.server.module.SAMLConsumeLoginModule  required;
};
```

The UNTConsumeLoginModule module is for user name tokens, and the SAMLConsumeLoginModule module is for SAML tokens.

1. Open the `wsjaas.conf` file in the *install_dir*`/imssoap/WEB-INF` directory.
2. For custom user name token module, add your module in the system.ims.soap.soapunt entry, after the default UNTConsumeLoginModule module:

   ```
   system.ims.soap.soapunt  {
       com.ibm.ims.soap.server.module.UNTConsumeLoginModule required;
       com.yourcompany.security.server.yourUNTPLoginModule required;
   };
   ```

   The com.yourcompany.security.server.yourUNTPLoginModule entry is your custom authentication module. The underlying security module calls the modules in the specified order when a WS-Security request with the user name token is encountered. The default UNTConsumeLoginModule module is invoked first followed by the yourUNTPLoginModule module.
3. For custom SAML modules, add your module in the system.wss.consume.saml11 or system.wss.consume.saml20 entry, after the default SAMLConsumeLoginModule module:

   ```
   system.wss.consume.saml11 {
       com.ibm.ims.soap.server.module.SAMLConsumeLoginModule required;
       com.yourcompany.security.server.yourSAMLLoginModule required;
   };
   ```

   The com.yourcompany.security.server.yourSAMLLoginModule entry is your custom authentication module. The underlying security module calls the modules in the specified order when a WS-Security request with the SAML token is encountered. The default SAMLConsumeLoginModule module is invoked first followed by the yourSAMLLoginModule module.
4. Store your compiled custom module .class file in the *install_dir*`/imssoap/WEB-INF/classes` directory. For JAR files that contain classes files, store them in the *install_dir*`/imssoap/WEB-INF/lib` directory.

Code examples that you can customize for your own use are available from the IMS Enterprise Suite website.

Related reference:

Refer to the IBM WebSphere Application Server Version 8 information center for the SecurityToken API Javadoc information.
For more detail about the SecurityToken API, see the Javadoc information in the IBM WebSphere Application Server Version 8 information center.

Refer to the IBM WebSphere Application Server Version 8 information center for the SAMLToken API Javadoc information.
For more detail about the SAMLToken API, see the Javadoc information in the IBM WebSphere Application Server Version 8 information center.

Refer to the IBM WebSphere Application Server Version 8 information center for the UsernameToken API Javadoc information.
For more detail about the UsernameToken API, see the Javadoc information in the IBM WebSphere Application Server Version 8 information center.

Related information:

Custom authentication module samples on the IMS Exchange website
Download the SOAP Gateway custom authentication module samples from the IMS Exchange website.

Web service consumer (callout) scenario

SOAP Gateway enables IMS applications to make synchronous or asynchronous callout requests to external web services and, optionally, to receive responses back.

A *synchronous callout request* is a request from an IMS application that expects the response message from the external web service to return in the same transaction.

An *asynchronous callout request* is a request from an IMS application that either does not expect a response from the web service, or expects the response to return in a separate transaction. The response might be returned to the same or a different IMS application. The key benefit of having the response return asynchronously is not to hold up the dependent regions.

A synchronous callout request from an IMS application is issued by making an ICAL call to put the message in the hold queue, or transaction pipe (also known as tpipe). An asynchronous callout request from an IMS application is issued by making an ISRT ALTPCB call to insert a callout message into the alternate program communication block (ALTPCB) and to an OTMA destination descriptor. The OTMA destination descriptor contains the name of the destination tpipe where the synchronous and asynchronous callout request messages are queued.

For synchronous callout requests, SOAP Gateway communicates with IMS Connect to pull the messages from the hold queue and to send the response messages from the external web services back to IMS Connect and to the appropriate callout requester.

The following figure shows the SOAP Gateway runtime environment when IMS applications are enabled as web service consumers that issue asynchronous callout requests to an external web services. It demonstrates how SOAP Gateway supports both the synchronous and asynchronous callout function. The numbers in the figure correspond to the description that follows.

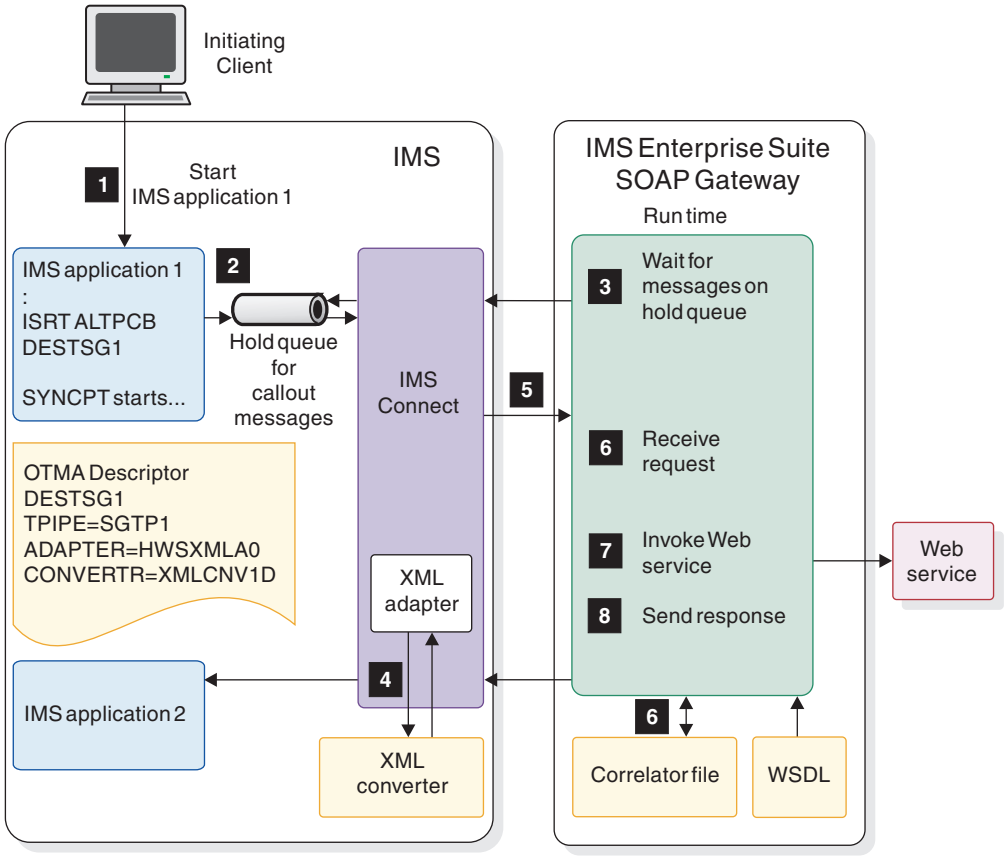


Figure 30. SOAP Gateway runtime environment for the IMS applications as web service consumers scenario (asynchronous callout)

1. A client starts IMS application 1.
2. IMS application 1 issues an ICAL call for a synchronous callout request, or inserts an asynchronous callout message into the alternate program control block (ALTPCB).
3. SOAP Gateway establishes a TCP/IP connection with IMS Connect and sends a request from the hold queue (the tpipe) to IMS Connect. The tpipe name is obtained from the connection bundle file. When SOAP Gateway starts, it scans through the correlator files for callout connection bundle information, and identifies the tpipes that are defined in those callout connection bundles.
4. IMS Connect retrieves the callout message as part of the request processing for SOAP Gateway. IMS Connect calls the adapter and converter for XML processing if necessary.
5. After the converter converts the callout message successfully, the IMS Connect XML adapter sends the callout request message back to SOAP Gateway.
6. SOAP Gateway receives the callout request message in XML, and parses the message to retrieve the service and payload data. Based on the service name and operation name values, SOAP Gateway obtains from its cache the corresponding correlator file and the web service information for invoking the web service. The request message for the web service is built. SOAP Gateway sends an acknowledgment to IMS Connect, and the message is removed from the corresponding tpipe.
7. SOAP Gateway sends the callout request message to the web service by using the SOAP/HTTP protocol.
8. The response is sent back to IMS (optional for asynchronous callout requests).

a. For synchronous callout requests, the response is sent back to the original IMS application that is waiting in the IMS dependent region.
b. For asynchronous callout requests, if a response is expected, the response is sent back as a separate IMS transaction. The response might be handled by the original, or a different, application.

Related concepts:

"Security for the consumer (callout) scenario" on page 182
Security support for the callout scenario is provided for messages from IMS to SOAP Gateway through SSL, and from SOAP Gateway to the web service through HTTPS.

One-way versus request-response web service invocation

When an IMS application is enabled as a web service consumer, it can invoke either a one-way web service operation or a request-response operation.

One-way invocation corresponds to the one-way port type, and the request-response invocation corresponds to the request-response port type of the web service operation.

In a one-way invocation (asynchronous notification), no response is expected from the web service. The invocation sends the request message by taking the XML message as the payload data but does not wait for the response to come back. For one-way invocations, issue an asynchronous callout request from your IMS application. SOAP Gateway would indicate in its log that no response is received from the external web service.

In a request-response invocation, a response message from the web service is returned to SOAP Gateway from the same SOAP/HTTP connection. SOAP Gateway then sends the response to IMS Connect either in the same transaction to the IMS application that has been waiting for the response (in the case of a synchronous callout request) or by invoking a new IMS transaction (in the case of an asynchronous callout request).

Send-only with acknowledgement protocol for web service consumer applications

The send-only with acknowledgement protocol allows your application to receive a final confirmation that the response message was delivered to the original IMS application that issued the callout request.

Restriction: Only synchronous callout applications can use the send-only with acknowledgement protocol. For IMS V12, the PTF for APAR PM91443 is required.

By default, SOAP Gateway uses the send-only protocol without acknowledgement to send callout response messages to IMS. The send-only with acknowledgement protocol provides an alternative approach that offers an extra level of confirmation that the response message was received. After the callout response message is sent to IMS, IMS sends either an ACK (positive acknowledgement) or NACK (negative acknowledgement) response to SOAP Gateway. The ACK or NACK is not sent to the external callout target application, but SOAP Gateway logs the response.

For NACK responses, SOAP Gateway logs an IOGS0082E message if the trace level is set to error or higher. For ACK responses, SOAP Gateway logs an IOGS0081I message if the trace level is set to informational or higher. If no ACK or NACK is received, SOAP Gateway logs an IOGS0083E message.

The main advantage of the send-only with ACK protocol is that you can easily identify if a callout response message was sent to IMS, whether IMS received the message, and how long it took the message to reach IMS. The main disadvantage is that the ACK or NACK reply adds an extra network communication event for each callout response message.

You can enable the send-only with acknowledgement protocol for a web service consumer application with the SOAP Gateway management utility when you create or update a correlator. Set the value for the `-k` property to `true`. The send-only with acknowledgement protocol is configured at the operation level and can be enabled or disabled without restarting the SOAP Gateway server.

Correlator files from previous versions of IMS Enterprise Suite must be migrated to correlator schema version 3.0 by using the SOAP Gateway management utility iogmgmt -migrate command to take advantage of this feature.

Related tasks:

"Migrating correlator files to schema version 3.0" on page 302
IMS Enterprise Suite Version 3.1 SOAP Gateway requires correlator schema version 3.0. To migrate an existing correlator file from older versions to version 3.0, use the SOAP Gateway management utility iogmgmt -migrate correlator command.

Related reference:

"-corr: Create or update a correlator entry" on page 439
Use the -corr command to create or update the transaction and runtime properties of a correlator entry.

Related information:

Send-only with acknowledgment protocol
Get more information about the send-only with acknowledgement protocol in *IMS Version 13 Communications and Connections*.

Thread management for callout messages retrieval

SOAP Gateway supports two options to determine how to manage the callout threads to send the requests to poll the hold queue for callout request messages: one thread per tpipe, or one thread per connection bundle.

When SOAP Gateway starts, it scans through the callout correlator files for connection bundle information, and identifies the tpipes that are defined in those callout connection bundles. SOAP Gateway then creates either one thread per tpipe, or one thread per connection bundle, based on the thread policy.

One thread per tpipe

With the one thread per tpipe configuration, SOAP Gateway creates one thread for each tpipe that is defined in the connection bundle or bundles. Identical tpipe names within a connection bundle are ignored and treated as the same tpipe. Identical tpipe names across connection bundles are valid.

All messages that are queued to a tpipe are retrieved until no more messages are available on the queue. The appropriate web service is invoked for each message that is retrieved.

This flow is repeated in all threads for each tpipe in all connection bundles.

One thread per connection bundle

With the one thread per connection bundle configuration, SOAP Gateway creates one thread for each connection bundle. Identical tpipe names within a connection bundle are ignored and treated as the same tpipe. Identical tpipe names across connection bundles are valid.

One single message is retrieved at a time, and the appropriate web service is invoked. After a message is retrieved (or if there is no message) the thread moves to the next tpipe, in the same connection bundle, and repeats the message retrieval steps.

This retrieval process is repeated for all the tpipes in the connection bundle in a round-robin fashion. After the end of the last tpipe in the connection bundle, the entire process is repeated all over again, starting with the first tpipe in the same connection bundle.

Configuration and design considerations

- Do not use the same tpipe for callout requests and for regular asynchronous output messages.
- Keep synchronous and asynchronous callout messages on different tpipes unless the configuration of your environment or the design of your applications requires that they share the same tpipe.
- You must use the OTMA destination descriptor for callout requests from IMS.
- A separate OTMA destination descriptor definition is required for each web service operation destination (one for each converter) if you are using the IMS Connect XML adapter function.
- To achieve high availability, multiple instances of SOAP Gateway servers can be configured to pull messages out of the same tpipes.
- After the callout threads are stopped and then restarted (or the server is stopped and restarted without using the graceful shutdown option), the first synchronous callout request on each tpipe to SOAP Gateway is returned to OTMA with a NAK response. Subsequent synchronous callout requests are processed normally.

Related tasks:

"Starting and stopping all callout threads" on page 330
You can start and stop the callout threads by using the SOAP Gateway management utility.

"Starting the callout thread for a specific application" on page 268
You must start a callout thread after a callout application is deployed to SOAP Gateway before the application can begin processing callout messages.

"Stopping the thread pool" on page 332
Stopping the thread pool stops all worker threads. When the worker threads are stopped, no messages in the callout work queue are processed.

Related reference:

"Callout properties for thread management" on page 179
SOAP Gateway provides several properties for configuring and managing the threads.

Poll interval for callout messages

You can configure how long each thread sleeps or waits before it retrieves the callout requests by setting a poll interval.

A poll interval is the time between the end of callout request retrieval processing (that does not include the web service invocation), and the start of the invocation of the next callout request retrieval.

If one-thread-per-tpipe thread policy is used, the poll interval is the wait time of the thread between the end of the callout request retrieval processing and the invocation of the next callout request retrieval on the same tpipe that the thread is dedicated to.

With the one-thread-per-tpipe policy, the user-specified poll interval setting applies only when an error occurs. The error could be caused by web service timeout, communication errors with IMS Connect, or other factors.

If the one-thread-per-connection-bundle thread policy is used, the poll interval indicates the wait time of the thread between the end of a callout request retrieval processing and the invocation of the next callout request retrieval on the next tpipe in the connection bundle.

Use the SOAP Gateway management utility to specify the poll interval.

Related reference:

"-callout -updateprop: Update SOAP Gateway callout properties" on page 434
The -callout –updateprop command updates the SOAP Gateway callout properties.

"Callout properties for thread management" on page 179
SOAP Gateway provides several properties for configuring and managing the threads.

Thread pool for maximum concurrency

To achieve maximum concurrency, SOAP Gateway uses different threads to process requests and to dispatch their responses. The callout threads pull callout messages, whereas the invocation of the external web service and the dispatching of the response is handled by the worker threads in the thread pool.

Callout messages from a tpipe are retrieved by a callout thread and sent to the worker threads for processing. After a message is retrieved, the callout thread validates the message, sends the validated callout message to a free worker thread for processing, and then continues to retrieve the next message.

For a one-way, asynchronous callout message that does not expect a response from the web service, the worker thread completes processing the request (because there is no response) and returns to the free thread pool for future callout requests.

For a request-response callout messages:

- If the callout message is asynchronous, SOAP Gateway sends a response message back to IMS as a new send-only transaction.
- If the callout request is synchronous, SOAP Gateway sends the response message back to IMS as a new send-only transaction to the requesting IMS application that remains in the dependent region. SOAP Gateway handles the correlation to ensure that the response goes back to the IMS application that initiated the callout request.

In either case (one-way or request-response), the response to the callout request is sent on a different IMS Connect shareable persistent socket connection.

Thread types, in-flight messages, and the work queue

- The *callout threads* pull callout messages, validate them, and assign callout messages to a work queue. The callout threads are also referred to as the main threads.
- The *worker threads* sleep when there are no callout messages in the work queue to process. When a callout message appears on the queue, a free worker thread is assigned to process it. The worker threads issue the calls to the external web services and return the responses. The number of worker threads is specified in the numberOfWorkerThreadsInPool setting.
- Messages that are being processed by worker threads are called *in-flight messages*. The processing of a message by a worker thread is also known as executing a job.
- The *daemon thread* maintains the number of worker threads at the configured level. The daemon thread wakes up at an interval that is specified in the checkWorkerHealthInterval callout property to ensure that the required number of worker threads are up. If not, the daemon thread starts additional worker threads to maintain the level that is specified in the numberOfWorkerThreadsInPool callout property. A daemon thread is started when the SOAP Gateway server starts up, to monitor the thread pool.

The numberOfWorkerThreadsInPool property specifies the system resources that are used by SOAP Gateway and must be configured with care. You can configure up to 32 worker threads. The SOAP Gateway management utility can be used to administer the callout threads (for stopping and starting) and to configure this numberOfWorkerThreadsInPool setting.

A worker thread is considered non-existent if the thread exits due to an error.

The thread pool accesses the work queue where callout messages (jobs) are placed for processing. The length of this worker queue, or the number of maximum in-flight messages that are allowed at a given time, is specified in the queueThrottleLength setting.

Callout message processing flow

The following diagram describes the basic flow of a successful callout message processing scenario.

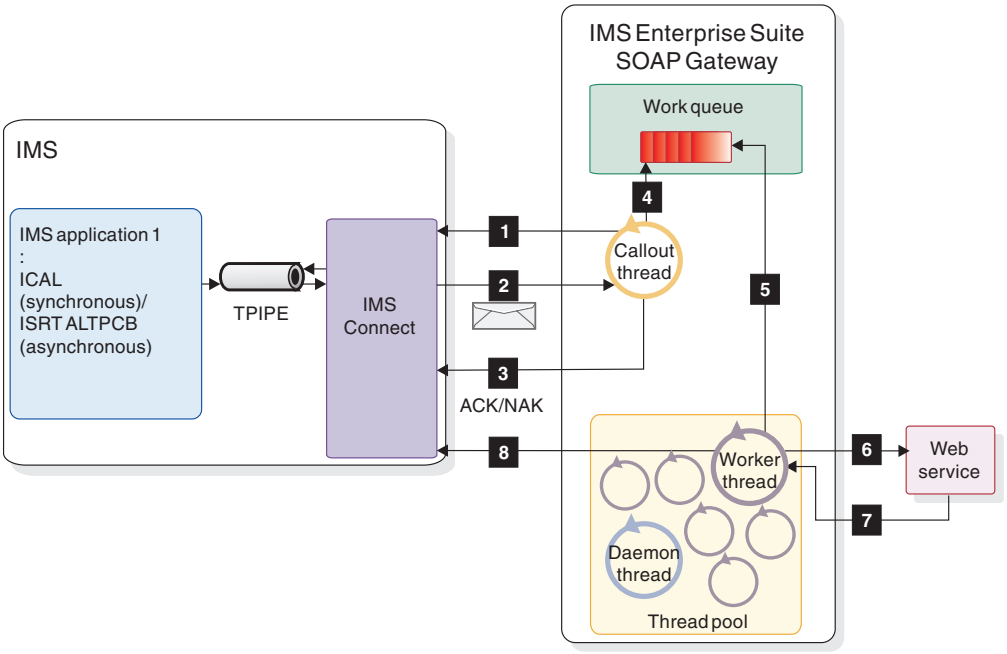


Figure 31. Callout message processing

1. The callout thread pulls the callout messages that are placed on the hold queue.
2. A callout message is retrieved.
3. The callout thread sends an ACK (or NAK if SOAP Gateway cannot process this message).
4. The message is placed into the work queue.
5. A free worker thread is assigned to process the message. The message now becomes an in-flight message. Processing of an in-flight message is referred to as executing a job.
6. The worker thread sends the callout request to the web service.
7. The web service returns a response.
8. The worker thread forwards the response to the initiating client.

Related reference:

"Callout properties for thread management" on page 179
SOAP Gateway provides several properties for configuring and managing the threads.

Error policy for managing threads

You can specify an error policy to control how SOAP Gateway continues to poll callout request messages if an error occurs.

Errors might occur when the main thread is polling callout messages from the IMS OTMA tpipe or when it waits for the response message from the web service. Errors might also occur when there are still messages being processed while the SOAP Gateway server is shut down.

Use the SOAP Gateway management utility to configure these error handling settings.

Errors during callout message polling

The shouldStopRTOnError property setting determines if the main thread ignores the error and continues to poll callout messages. If the shouldStopRTOnError property is set to true, the thread that encounters the error stops and no longer continues to poll callout request messages from its tpipe or all the tpipes in the connection bundle.

If the shouldStopRTOnError property is set to false:

- For the one-thread-per-tpipe configuration, the thread skips the error, discards the message, and continues to poll callout request messages from its tpipe.
- For the one -thread-per-connection bundle configuration, the error is ignored, the message is discarded, and the thread continues to poll messages from the next tpipe in the connection bundle.

Evaluate the two thread management options based on your needs, available resources, and performance considerations.

In-flight message handling when the thread pool is stopped

When the thread pool is stopped by using the iogmgmt -callout -stoppool command, all in-flight messages are processed before the thread pool is stopped. To force the thread pool to stop immediately and discard all pending messages, add the -force option: iogmgmt -callout -stoppool -force.

Related reference:

"Callout properties for thread management"
SOAP Gateway provides several properties for configuring and managing the threads.

Callout properties for thread management

SOAP Gateway provides several properties for configuring and managing the threads.

Table 25. Callout properties

Property	Description
Thread poll interval	The time interval in milliseconds that the SOAP Gateway callout thread waits before polling for messages on a particular tpipe. The time interval value must be greater than -1. When the oneThreadPerTpipe property is set to `true`, the poll interval applies only when an error occurs. The default is 3000.
Stop on thread error	Set this property to `true` if you want the SOAP Gateway to terminate a callout thread when an error is encountered while it is retrieving or processing a callout message on that thread. Set this property to `false` if you want the processing to continue even if there is an error. The default is `false`.
One thread per tpipe	Set this property to `true` if you want one thread dedicated for every tpipe that must be pulled for messages. Set this property to `false` if you want to have one thread per connection bundle, which could contain several tpipes to be pulled for messages. The default is `true`.

Table 25. Callout properties (continued)

Property	Description
Number of worker threads in pool	The number of live worker threads in the thread pool. Each of these worker threads executes a job and then returns to the pool for reuse by the SOAP Gateway runtime engine. The maximum number of worker threads is 32. The default is 5.
Queue throttle length	The number of maximum in-flight messages in the queue that are allowed at run time. The default is 64.
Check worker health interval	The time interval in milliseconds that the daemon thread wakes up to monitor the execution time of the worker thread. The default is 600000, which is 10 minutes.
Thread pool cache capacity	The number of messages to be logged in the cache for debugging purposes. Each message is 9 lines. The default is 2000.
Thread pool cache directory	Set the directory (the full path) for the thread pool log cache dump.

Related reference:

"-callout -updateprop: Update SOAP Gateway callout properties" on page 434
The -callout –updateprop command updates the SOAP Gateway callout properties.

Thread management and configuration considerations

Depending on the work load in your environment, you can tune SOAP Gateway to maximize performance and throughput of the callout processing by configuring the SOAP Gateway callout properties.

Important: You must fully understand your requirements and fine-tune these properties appropriately to suit your needs in order to achieve maximum performance benefit out of these features. Inadvertent usage of these properties without fully understanding their implications could lead to poor performance.

If the thread pool is stopped or the worker threads become nonexistent, when callout threads are pulling messages, the messages will not be processed and eventually the callout threads stop or continue, depending on the stopOnThreadError property setting. If you have too few worker threads in the thread pool and many callout requests, there might be delay in servicing the callout requests. You might need to increase the number of worker threads in the thread pool. If there are fewer callout requests and too many worker threads, some worker threads might be idle in the thread pool. In this case, you might want to reduce the number of worker threads in the thread pool appropriately.

When the queue throttle length (specified in the queueThrottleLength property) is reached, the callout threads would temporarily pause so that the worker threads could catch up. This situation occurs if the worker threads are too slow, or if there are too few worker threads to process the jobs while there are too many callout threads or messages.

For the one thread per connection bundle policy, you can also slow down the process of callout requests retrieval by increasing the poll interval, so the worker threads would have the chance to empty the work queue. The ideal configuration for your environment is that the volume of the callout messages that are processed by the callout threads is balanced with the volume of the request processing by the worker threads. In other words, the worker threads need to consume or drain the worker queue as fast as the callout threads place callout requests in the queue. If the workers cannot catch up with the speed of the callout threads because the workers are too slow or the number of workers is too few, the work queue will gradually reach its maximum capacity. In such a situation, the callout threads will pause temporarily. As the worker threads keep draining the work queue, the callout threads will resume operation normally and will be able to add jobs to the work queue.

Tips:

- Monitor the work queue to ensure that the callout work queue is drained well. If the queue reaches its full capacity quickly or often, one option is to configure SOAP Gateway to have more worker threads, or set a slightly larger rtPollInterval property value so the callout messages are not pulled too quickly.
- The messages in the work queue (in-flight messages) are not persistent, and therefore are not recoverable when the server restarts. The draining of the in-flight messages is handled based on whether the thread pool is stopped gracefully or forced (with the `-force` option). Follow the following general steps when changing the SOAP Gateway callout properties or when stopping the SOAP Gateway server.

General steps to update a callout property

When you need to change any SOAP Gateway callout properties, it is better to stop the thread pool to ensure that no in-flight messages are being processed, and to avoid any undesirable results.

1. Stop the callout threads.
2. Stop the thread pool.
3. Update the callout properties.
4. Start the thread pool.
5. Start the callout threads.

Related reference:

"-callout -stopall: Stop all callout threads" on page 432
The -callout -stopall command stops all callout threads.

"-callout -stoppool: Stop the thread pool" on page 433
The -callout –stoppool command stops the thread pool.

"-callout -updateprop: Update SOAP Gateway callout properties" on page 434
The -callout –updateprop command updates the SOAP Gateway callout properties.

"-callout -startpool: Start the thread pool" on page 432
The -callout -startpool command starts the thread pool.

"-callout -startall: Start all callout threads" on page 431
The -callout -startall command starts all callout threads.

Callout messages correlation to web services

SOAP Gateway supports correlating a callout request message to a web service, and the response, if any, back to IMS.

A callout request message consists of a service data prefix and the payload data. The information in the service data prefix is used by SOAP Gateway to correlate the callout request message to the external web service.

The following table describes the elements in the service data prefix.

Table 26. Elements in the service data prefix

Elements in the prefix	Description
web service identifier (WSID)	Not used for the web service consumer scenario.
Namespace	The target namespace of the web services description language WSDL file.
Service name	The service name of the port of operation to be invoked.
Port name	The port name of the operation to be invoked.
Operation name	The operation name of the web service to be invoked.

After SOAP Gateway receives the callout request message from IMS Connect, it retrieves the service data prefix from the callout request message. By using the information in the service prefix, SOAP Gateway loads the corresponding correlator file and retrieves the following outbound web service information:

- The WSDL file name for the web service to be invoked
- The timeout value for waiting for a response from the web service
- The IMS transaction code (for asynchronous callout requests only) and the connection bundle name for returning the callout response

SOAP Gateway uses the information from the correlator file to correlate to the appropriate web service. The correlator file can be generated by using IBM Rational Developer for System z or the SOAP Gateway management utility.

Security for the consumer (callout) scenario

Security support for the callout scenario is provided for messages from IMS to SOAP Gateway through SSL, and from SOAP Gateway to the web service through HTTPS.

HTTPS encapsulates the SOAP messages from one point to another to prevent alteration to the exchanged messages. Although this level of security secures your messages and verifies your endpoints, it does not prevent execution of a web service by an unauthorized user.

To ensure that only authorized users can execute a web service, in addition to HTTPS support for invoking web services, SOAP Gateway also support passing of user information on a per web service or per message basis.

- Basic authentication is supported by passing user name and password information in the connection bundle to the web service server. SOAP Gateway supports both the IbmX509 and SunX509 algorithms when they are configured with either the IBM Java Runtime Environment (JRE) or Sun JRE that runs on the server that hosts the web service.
- WS-Security SAML confirmation method is supported for synchronous callout applications by extracting the user ID (the user that initiates the synchronous callout application) from the correlation token and passing it to the external web service.

Restriction: Security certificates for all external web service servers that IMS applications call out to must be stored in the same SOAP Gateway truststore because SOAP Gateway supports one truststore and one keystore per server instance.

Use different tpipes for secure and non-secure callout web services. Sharing the same tpipe for both secure and non-secure callout web services would result in problems with non-secure callout web services.

Important: You must use System SSL for communication with IMS Connect. You must apply the following fix, depending on the IMS version.

- IMS V13 APAR PM96825
- IMS V12 APAR PM98017
- IMS V11 APAR PM98018

SSL security, server authentication, and client authentication

You can use SSL security between IMS Connect and SOAP Gateway, and between SOAP Gateway and the external web services to provide security for the consumer scenario. Use of SSL on either side is optional and independent of the security setting on the other side.

Regardless of the scenarios (IMS applications as web service providers, consumers, or business event emitters), SOAP Gateway is always a client to IMS Connect. To enable the security between IMS Connect and SOAP Gateway for the web service consumer scenario, the steps are the same as the web service provider scenario:

1. Create a truststore for SOAP Gateway to store the SSL server certificate from IMS Connect.
2. Export the certificate from IMS Connect.
3. Import the IMS Connect server certificate into the SOAP Gateway truststore.

When SSL security is used between SOAP Gateway and the web service, SOAP Gateway establishes security when it sends the IMS callout request to the external web service by using HTTPS. The server that hosts the web service sends back a certificate. After the transmission is secured and SOAP Gateway determines that the server certificate can be trusted, it executes the web service.

The following diagram shows the process flow of the server authentication process, where the web service server sends the client (SOAP Gateway) a certificate, and the client looks up the certification in its truststore and verifies that it knows the server before engaging communication.

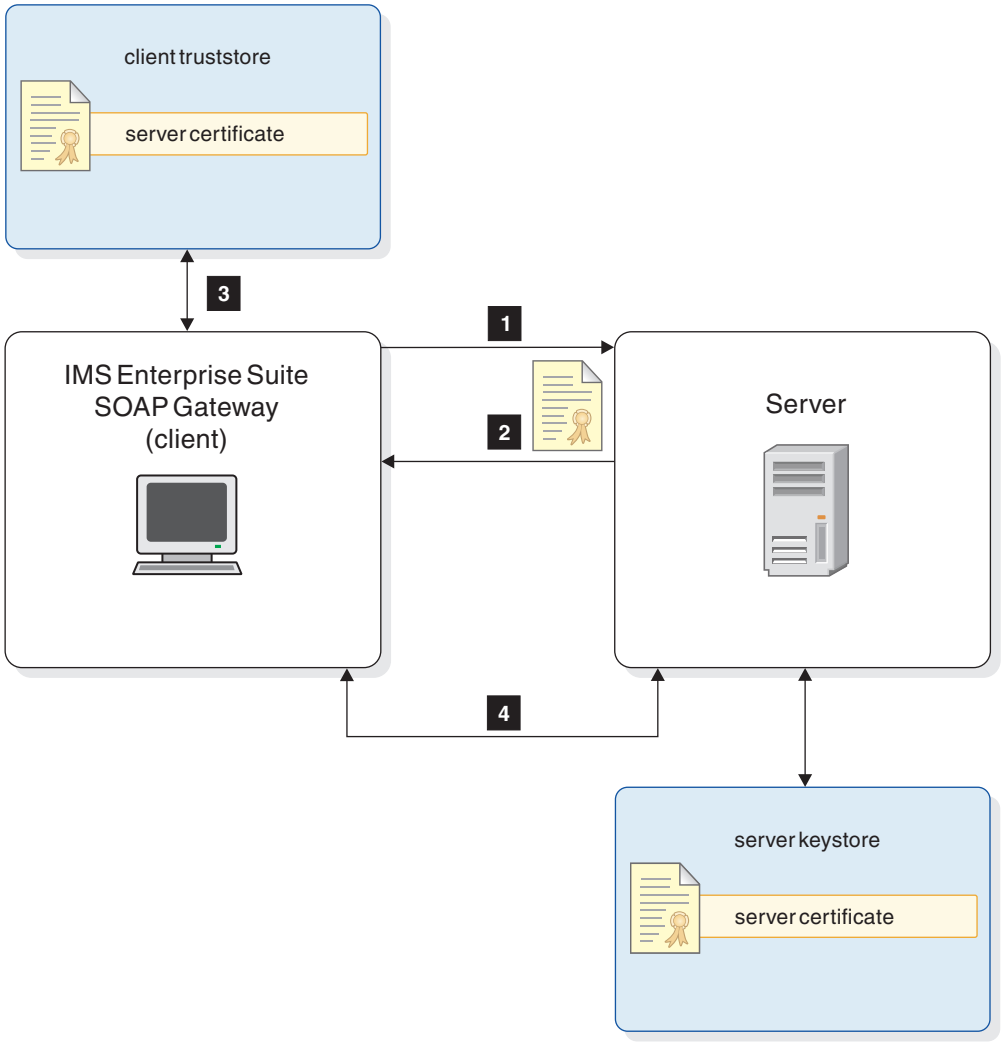


Figure 32. Server authentication for the IMS applications as web service consumers scenario

1. The client (SOAP Gateway) initiates an HTTPS call.
2. The web service server sends back a certificate.
3. The client verifies the certificate with the server certificate stored in the truststore.
4. After the transmission is secure, the client is allowed to execute the services.

The following diagram shows the process flow of the client authentication security scheme, where both the server that hosts the web service and the client (SOAP Gateway) that requests the service require authentication from the other before a connection is established.

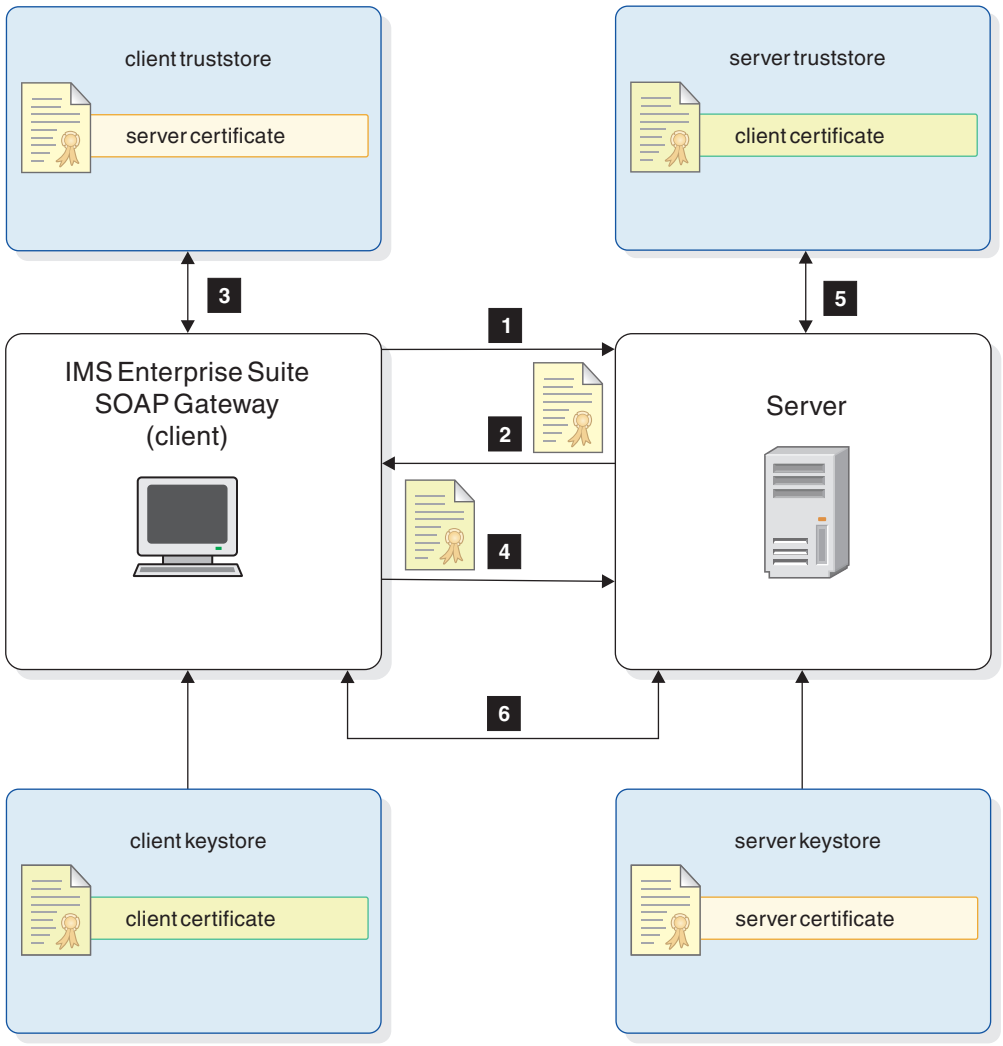


Figure 33. Client authentication for the IMS applications as web service consumers scenario

1. The client initiates an HTTPS call.
2. The server sends back a certificate.
3. The client verifies the certificate with the server certificate that is stored in the truststore.
4. The client sends the server a certificate.
5. The server verifies the client certificate with the certificate that is stored in the truststore.
6. After the transmission is secured, the client is authenticated and allowed to access protected services.

Important: If the SOAP Gateway client and the web service are hosted on the same system, do not use the same JRE for both SOAP Gateway and your server. The runtime environments must be different instances of a JRE. Otherwise, the JVM cannot distinguish whether the keystore belongs to the client or the server.

Basic authentication security scheme

Basic authentication means that the server that hosts the web service requires the client (SOAP Gateway) to have proper basic authentication credentials in order to invoke a service. The client transmits the credentials during the invocation of a web service.

The user ID and password for basic authentication are stored in the connection bundle. The user ID and password are specified when you generate the connection bundle for a callout web service by using the SOAP Gateway management utility. SOAP Gateway retrieves the pre-defined basic authentication security token from the connection bundle and passed it to external web service.

Basic authentication can be used with or without HTTPS. Without HTTPS, the user name and password information is transmitted in clear text.

Recommendation: Use basic authentication with HTTPS support to ensure that user name and password information is not transmitted in clear text.

The following diagram shows the basic authentication security scheme:

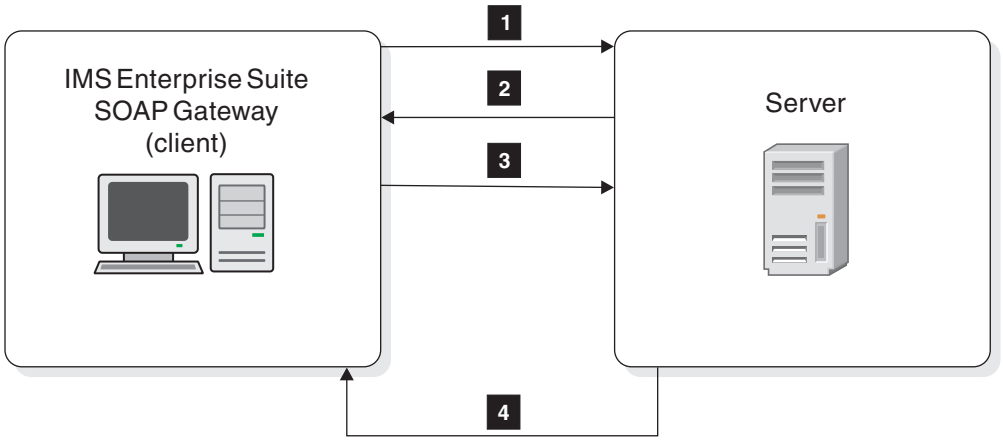


Figure 34. Basic authentication for the IMS applications as web service consumers scenario

1. The client (SOAP Gateway) initiates a request for the protected service.
2. SOAP Gateway retrieves the basic authentication user ID and password information that is stored in the connection bundle.
3. The web service server requests basic authentication information (user ID and password).
4. The client sends the authentication information.
5. After the transmission is secure, the server returns the requested service.

You can use server authentication and basic authentication together. SOAP Gateway supports the following combinations of web service callout security scenarios:

- Server and basic authentication
- Client authentication
- Client authentication and basic authentication

Related concepts:

"Security support in SOAP Gateway" on page 30
SOAP Gateway supports HTTPS communication with its clients, and SSL communications with its host, IMS Connect.

Security features supported for the web service consumer scenario

SOAP Gateway supports server authentication, client authentication, and basic authentication to secure callout requests to external web services. Web services security (WS-Security) is also supported for synchronous callout requests so the user ID is sent with the SAML token to the external web service.

SOAP Gateway can be configured for server authentication and client authentication through Java keystore (JKS).

For message-level web services security (WS-Security), you can use the following token types:

- SAML 1.1 sender-vouches unsigned tokens
- SAML 2.0 sender-vouches unsigned tokens

The following table lists the supported security features for the consumer scenario.

Table 27. Supported security features for the web service provider scenario

Security feature	Description	Key type
Server authentication	The server hosting the web services provides server authentication information (certificate) to the client (SOAP Gateway) that binds the server identify to subsequent communications.	Java keystore (JKS)
Client authentication	Also known as *mutual authentication* because in addition to server authentication, the client (SOAP Gateway) must send certification information to the server.	JKS
Basic authentication	The server hosting the web service requires the client (SOAP Gateway) to have proper basic authentication credentials in order to invoke a service.	JKS
WS-Security	SAML 1.1 and SAML 2.0 unsigned tokens are supported for synchronous callout requests. Client authentication is required to pass the security token.	JKS
Custom authentication module	You can plug in your own custom Java Authentication and Authorization Service (JAAS) authentication module when WS-Security is enabled for the deployed callout web service, and client authentication is configured.	JKS

Combined server authentication and basic authentication security scheme

With combined server authentication and basic authentication, the client (SOAP Gateway) that requests the web service is required to establish HTTPS security before it sends the basic authentication credentials to be authenticated.

In this scheme, the client, SOAP Gateway, initiates an HTTPS call, the sever sends back a certificate, and the client verifies the authenticity of the certificate.

After it secures the transmission, the client sends the basic authentication credentials to be authenticated by the server.

The following diagram shows the process flow when server authentication is used with the basic authentication security scheme:

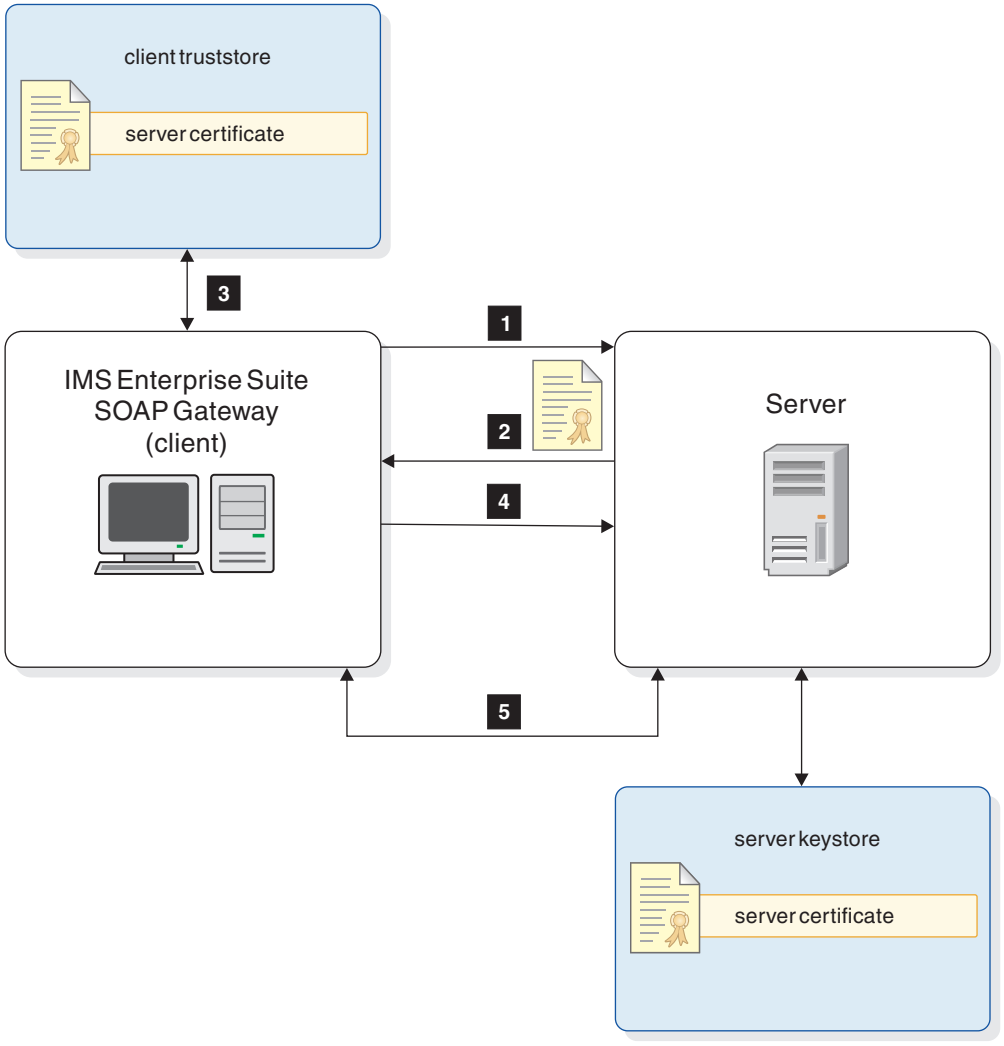


Figure 35. Server authentication and basic authentication for the IMS applications as web service consumers scenario

1. The client (SOAP Gateway) initiates an HTTPS call.
2. The server sends back a certificate.
3. The client verifies the certificate with the server certificate that is stored in the truststore.

4. The client sends the basic authentication credentials (user name and password) to the server.
5. After securing the transmission, the client is allowed to access protected services.

Client authentication and basic authentication security scheme

With client authentication and basic authentication, the server that hosts the web service and the client (SOAP Gateway) require the other to be authenticated in order to establish trust before it establishes a connection.

The client, SOAP Gateway, initiates an HTTPS call, the sever sends back a certificate, and then the client sends the server a certificate.

Because client authentication is used with basic authentication, in addition to the HTTPS call and the certificates that the server and the client send to each other, basic authentication credentials are also sent to the server with the request in order to execute a web service.

The following diagram shows the process flow when client authentication is used with the basic authentication security scheme:

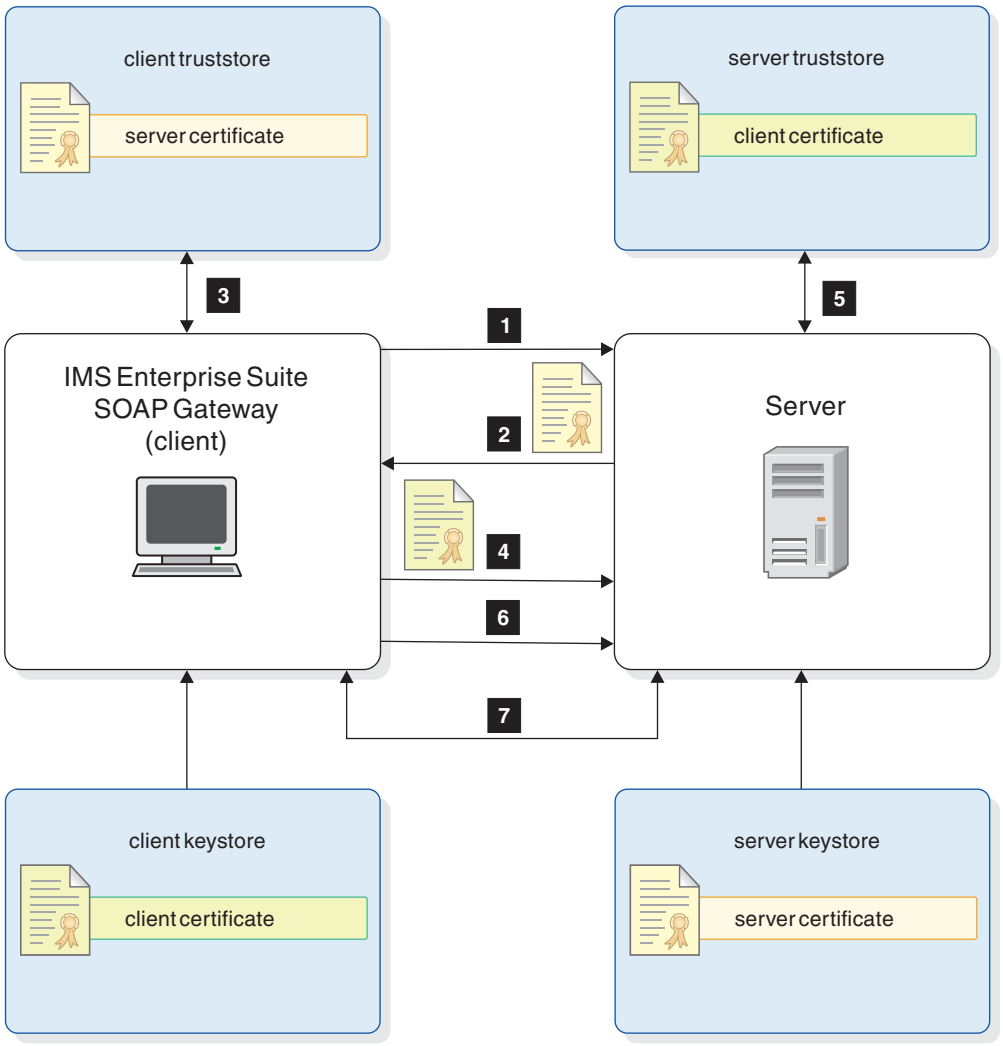


Figure 36. Client authentication and basic authentication for the IMS applications as web service consumers scenario

1. The client initiates an HTTPS call.
2. The server sends back a certificate.
3. The client verifies the certificate with the server certificate that is stored in the truststore.
4. The client sends the server a client certificate.
5. The server verifies with the client certificate that is stored in the truststore.
6. Basic authentication credentials (user name and password) are sent to the server because the server is set up to require this information.
7. The client is allowed to execute a web service.

Security process flow with SAML tokens for the synchronous callout scenario

For the synchronous callout scenario, you can deploy a callout web service and specify either a SAML 1.1 or SAML 2.0 unsigned token to send user ID information to the external web service for further authentication and authorization.

When you deploy a synchronous callout web service, if a SAML token type is specified, SOAP Gateway generates the SAML token for the callout web service.

SOAP Gateway extracts the user ID from the correlation token that comes with the IMS synchronous callout request. This user ID is passed in the SOAP header to the external web service for further authentication and authorization.

The following figure shows the processing between SOAP Gateway and the external web services when WS-Security is enabled for the callout application. The diagram does not include details on the communications between IMS and SOAP Gateway for callout request processing.

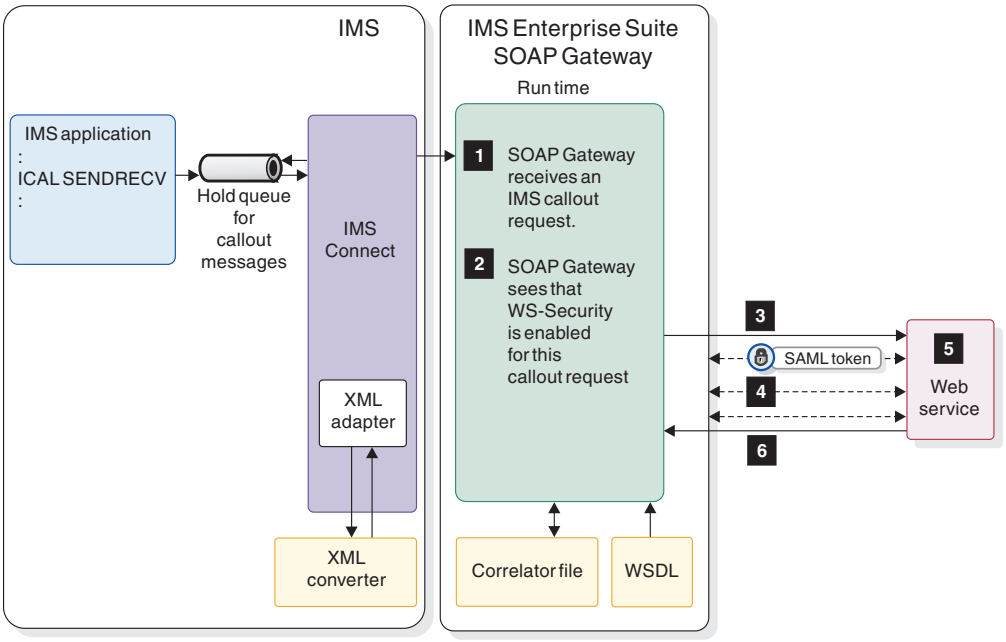


Figure 37. Security process flow with SAML token support for synchronous callout

1. A deployed callout web service on SOAP Gateway receives the callout request message in XML, and parses the message to retrieve the service and payload data.
2. Based on the service name and operation name values, SOAP Gateway obtains from its runtime configuration the corresponding correlator file and the web service information for invoking the web service.
   - The correlator file indicates that WS-Security is enabled for this callout web service.
   - The callout web service was deployed with either a SAML 1.1 token or SAML 2.0 token.
3. SOAP Gateway sends the request to the external web service with the SOAP Gateway generated SAML token through SSL over HTTP.
4. The external web service server and SOAP Gateway exchanges security credentials:
   a. The server sends back a certificate.
   b. SOAP Gateway verifies the certificate with the server certificate that is stored in the truststore.
   c. SOAP Gateway sends the server a client certificate.
   d. The server verifies with the client certificate that is stored in the truststore.
   e. SOAP Gateway is allowed to execute a web service.

5. The external web service extracts the user ID from the SAML token and does additional authentication and authorization checking.
6. The response from the external web service is sent back to the original IMS application that is waiting in the IMS dependent region

Example: Configuring the client authentication and basic authentication security scheme

This example demonstrates how to create self-signed certificates to configure client authentication and basic authentication when the web service is hosted on an Apache Tomcat server on Windows. The actual location of the key management utility might be different based on your server environment.

In this example, the Apache Tomcat server that hosts the web service is referred to as the server. SOAP Gateway is referred to as the client that is requesting a web service.

The instructions assume that both the server and the client are installed on Windows systems.

To configure client authentication and basic authentication for callout requests:

1. On the web service server, create the server keystore and truststore, and export the server certificate.
   a. In a command prompt, change the directory to where the keytool program is located on the Apache server (for example, *server_dir*\java\jre\bin).
   b. Create a keystore (server.keystore.ks):

      ```
      keytool -genkey -alias server.keystore -dname "CN=hostname.company.com
      OU=IBM SWG, O=IBM, C=US" -keyalg RSA -keypass password
      -storepass password -keystore server.keystore.ks
      ```

      The CN value must be a valid hostname. The exact hostname must be specified in the callout web service WSDL endpoint. For example,

      ```
      <wsdl:service name="IMSSOAPCalloutService">
        <wsdl:port binding="tns:IMSSOAPCalloutBinding" name="IMSSOAPCalloutPort">
          <soap:address location=
           "https://hostname.company.com:8443/imssoap/services/IMSSOAPCalloutService"/>
        </wsdl:port>
      ```

      If the hostname in the server keystore and the hostname in the callout WSDL file do not match, you would get an IOGS0077E error:

      ```
      IOGS0077E: An error occurred during the invocation of the external web
      service: hostname in certificate didn't match:
      <hostname1.company.com> != <hostname2.company.com>
      ```

      For NIST SP800-131a, specify SHA256withRSA for the signature algorithm and 2048 for the key size.

      ```
      keytool -genkey -alias server.keystore -dname
      "CN=hostname.company.com OU=IBM SWG, O=IBM, C=US"
      -keyalg RSA -sigalg SHA256withRSA  -keysize 2048
      -keypass password -storepass password
      -keystore "/path/to/server.keystore.ks"
      ```

   c. Create a truststore (server.truststore.ks):

      ```
      keytool -genkey -alias server.truststore -dname "CN=Server Truststore,
      OU=IBM SWG, O=IBM, C=US" -keyalg RSA -keypass password
      -storepass password -keystore server.truststore.ks
      ```

      For NIST SP800-131a, specify SHA256withRSA for the signature algorithm and 2048 for the key size.

```
keytool -genkey -alias server.truststore -dname
"CN=Server Truststore OU=IBM SWG, O=IBM, C=US"
-keyalg RSA -sigalg SHA256withRSA  -keysize 2048
-keypass password -storepass password
-keystore "/path/to/server.truststore.ks"
```

d. Export the server certificate (server.keystore.cer) from the server keystore:

```
keytool -export -alias server.keystore -storepass password
-file server.keystore.cer -keystore server.keystore.ks
```

2. On the client (SOAP Gateway), create the client keystore and truststore, and export the client certificate.
   a. In a command prompt, change the directory to *SOAP_Gateway_install_directory*\java\jre\bin, where the keytool program is located. For example:

```
cd C:\Program Files\IBM\IMS Enterprise Suite Vx.x\SOAP Gateway\java\jre\bin
```

   b. Create the keystore (callout.client.keystore.ks):

```
keytool -genkey -alias callout.client.keystore -dname
"CN=IMS Callout Client Keystore, OU=IBM SWG, O=IBM, C=US" -keyalg RSA
-keypass password -storepass password
-keystore callout.client.keystore.ks
```

   c. Create a truststore (callout.client.truststore.ks):

```
keytool -genkey -alias callout.client.truststore -dname
"CN=IMS Callout Client Truststore, OU=IBM SWG, O=IBM, C=US" -keyalg RSA
-keypass password -storepass password
-keystore callout.client.truststore.ks
```

   d. Export the client certificate (callout.client.keystore.cer) from the client keystore:

```
keytool -export -alias callout.client.keystore -storepass password
-file callout.client.keystore.cer -keystore callout.client.keystore.ks
```

3. Transfer the web service server certificate to the location of the client (SOAP Gateway) truststore (or a location that the client can access).
4. Transfer the client (SOAP Gateway) certificate to the location of the server truststore (or a location that the server can access).
5. On the server, import the client certificate and configure the server to point to the keystore and truststore.
   a. Go to the *server_dir*\java\jre\bin directory on the server.
   b. Import the client certificate (callout.client.keystore.cer):

```
keytool -import -v -trustcacerts -alias callout.client
-file callout.client.keystore.cer -keystore server.truststore.ks
-keypass password -storepass password
```

   c. Modify the file server.xml to point to the keystore and truststore with the appropriate passwords.
      - Set **clientAuth** to true.
      - Set **sslProtocol** to SSL.
      - For NIST SP800-130a:
        – Add the following two attributes if they do not yet exist:

```
SSLEnabled="true"
sslEnabledProtocols="TLSv1.2"
```

        – Remove the sslProtocol attribute if it is present.
      - Specify the file location for keystore and truststore.
      - Specify the password for keystore and truststore

      The following example demonstrates the settings when the IbmX509 algorithm is used with an IBM Java Runtime Environment (JRE). For the

SUN JRE, set the algorithm attribute to SunX509. This example has client authenticated specified, but is not FIPS-enabled and does not support NIST SP800-130a.

```
<!-- Define a SSL HTTP/1.1 Connector on your port -->
<Connector SSLEnabled="true"
acceptCount="100" algorithm="IbmX509" clientAuth="true"
disableUploadTimeout="true" enableLookups="false"
keystoreFile="c:\keys\server.keystore.ks"
keystorePass="password"
maxThreads="150" minSpareThreads="25" port="8663"
scheme="https" secure="true" sslProtocol="SSL"
truststoreFile="c:\keys\server.truststore.ks"
truststorePass="password"/>
```

To support NIST SP800-130a, remove the **sslProtocol** attribute, and add **sslEnabledProtocols** attributes.

```
<!-- Define a SSL HTTP/1.1 Connector on your port -->
<Connector SSLEnabled="true"
acceptCount="100" algorithm="IbmX509" clientAuth="true"
disableUploadTimeout="true" enableLookups="false"
sslEnabledProtocols="TLSv1.2"
keystoreFile="c:\keys\server.keystore.ks"
keystorePass="password"
maxThreads="150" minSpareThreads="25" port="8663"
scheme="https" secure="true"
truststoreFile="c:\keys\server.truststore.ks"
truststorePass="password"/>
```

6. On the server, configure basic authentication.
   a. Add a role for the user credentials and for the client certificate by modifying the file tomcat-users.xml in *server_dir*\server\conf\. The following example demonstrates the specification of a callout user *imsuser* with the *imspwd* as the password.

```
<role rolename="imsuser"/>
<user username="imsuser" password="imspwd" roles="imsuser"/>

<!-- Add the client certificate as a user to tomcat -->
<user username="CN=IMS Callout Client Keystore, OU=IBM SWG, O=IBM, C=US"
password="null" roles="admin"/>
```

   b. Add the security constraints and login configurations to a specific web service by modifying the file web.xml in *server_dir*\server\webapps\imssoap\WEB-INF\. For example:

```
<security-constraint>
   <web-resource-collection>
       <web-resource-name>Protected</web-resource-name>
       <!-- Specify the directory for restricted web Services application -->
      <url-pattern>/servlet/YourServlet</url-pattern>
   </web-resource-collection>
   <auth-constraint>
       <!-- Specify the role name of the new user -->
       <role-name>imsuser</role-name>
   </auth-constraint>
</security-constraint>

<!-- Define the Login Configuration for this Application -->
<login-config>
   <auth-method>BASIC</auth-method>
   <realm-name>Protected Web Services</realm-name>
</login-config>
```

7. On the client (SOAP Gateway), import the server certificate (server.keystore.cer):
   a. Go to the directory *SOAP_Gateway_install_directory*\java\jre\bin, where the keytool program is located.
   b. Import the server certificate (server.keystore.cer).

```
keytool -import -v -trustcacerts -alias server
-file <path_to>/server.keystore.cer
-keystore <path_to>/callout.client.truststore.ks -keypass password
-storepass password
```

8. On the client, configure the callout security by using the SOAP Gateway management utility. Use the iogmgmt -conn command to specify the callout security information. For example:

```
iogmgmt -conn -c -n MyCalloutConnBundle -h ICONHOST -p 9998
-d IMSSTOR1 -i tpipe1
-l callout_target_ks_name -y callout_target_ks_pwd
-v callout_target_ts_name -q callout_target_ts_pwd
-m callout_target_basic_auth_id -b callout_target_basic_auth_pwd
```

The configuration of client authentication and basic authentication is completed.

Enabling WS-Security for synchronous callout

To enable WS-Security to propagate user ID information with a request to an external web service, you must specify the keystore and truststore to the callout connection bundle. Then deploy the callout web service with the token type specified.

Prerequisite: Client authentication must be configured. For more information, see "Example: Configuring the client authentication and basic authentication security scheme" on page 192.

1. Create a connection bundle and provide callout keystore and truststore information. The following example creates a callout connection bundle:

```
iogmgmt -conn -c -n myCalloutConnBundleName
-l /usr/lpp/some/where/myclient_keystore.ks -y keystore_password
-v /usr/lpp/some/where/myclient_truststore.ks -q truststore_password
```

2. Deploy the callout web service that would process and send the IMS callout request to an external web service. The following sample deploys a callout web service that sends a SAML 2.0 unsigned token to the external web service:

```
iogmgmt -deploy -w myCalloutService.wsdl -r myCalloutService.xml -t SAML20Token
```

Custom authentication modules for callout security:

You can use a Java Authentication and Authorization Service (JAAS) module to intercept an outgoing message for the synchronous callout scenario to access the security header to perform additional validation before the SOAP request messages are sent to the external web service server.

Custom JAAS security module

JAAS authentication is performed in a pluggable fashion. The custom JAAS security module could access the following objects:

- WS-Security SAML token from the security header of an outbound message
- Peer certificate in X509 format

The Distinguished Name (DN) and X509 certificate information is included in the WS-Security token. DN contents are defined as:

- CN = common name
- OU = organizational unit
- O = organization name
- L = locality name
- S = state name

- C = country name

This custom module is used to identify the user ID in the token to be authorized with the DN contents.

If the module is coded to reject a request by throwing an exception, SOAP Gateway generates an Axis fault with the thrown exception message. Otherwise, the request is assumed accepted, and SOAP Gateway continues to send out the SOAP message to the external web service server. If for some reason the WS-Security token is null or missing in the SOAP request, SOAP Gateway would reject the request.

The JAAS configuration file is the `wsjaas.conf file` under the *`install_dir`*`/imssoap/WEB-INF` directory.

The following entries show how a SAML token module are configured in SOAP Gateway.

```
system.wss.generate.saml11 {
    com.ibm.ws.wssecurity.wssapi.token.impl.SAMLGenerateLoginModule  required;
};
...
system.wss.generate.saml20 {
    com.ibm.ws.wssecurity.wssapi.token.impl.SAMLGenerateLoginModule  required;
};
```

These modules define which SOAP Gateway class should be used to generate the SAML token for the callout request.

- The system.wss.generate.saml11 class specifies the module to use to generate the SAML 1.1 security token.
- The system.wss.generate.saml20 class specifies the module to use to generate the SAML 2.0 security token.
- The SAMLGenerateLoginModule module is a default module that is required and cannot be removed.

To plug in your custom authentication module, add your module in either the system.wss.generate.saml11 or system.wss.generate.saml20 entry, after the default module:

```
system.wss.generate.saml20 {
    com.ibm.ws.wssecurity.wssapi.token.impl.SAMLGenerateLoginModule  required;
    com.yourcompany.security.server.GenerateSAMLLoginModule required;
};
```

The GenerateSAMLLoginModule module is called after the default SAMLGenerateLoginModule module.

Related reference:

Refer to the IBM WebSphere Application Server Version 8 information center for the SecurityToken API Javadoc information.
For more detail about the SecurityToken API, see the Javadoc information in the IBM WebSphere Application Server Version 8 information center.

Refer to the IBM WebSphere Application Server Version 8 information center for the SAMLToken API Javadoc information.
For more detail about the SAMLToken API, see the Javadoc information in the IBM WebSphere Application Server Version 8 information center.

Plugging in a custom authentication module for callout security:

Define your custom authentication module in the corresponding SAML token generation entries in the `wsjaas.conf` file in the SOAP Gateway installation.

To add a custom authentication module entry in the JAAS configuration file to intercept outgoing requests, your custom entry must be added after the default login module in the corresponding SAML token generation module.

The system.wss.generate.saml11 module is for SAML 1.1 tokens, and the system.wss.generate.saml20 module is for SAML 2.0 tokens.

1. Open the `wsjaas.conf` file in the *install_dir*`/imssoap/WEB-INF` directory.
2. Add your module in the system.wss.generate.saml11 or system.wss.generate.saml20 entry, after the default SAMLGenerateLoginModule module:

   ```
   system.wss.generate.saml11 {
       com.ibm.ws.wssecurity.wssapi.token.impl.SAMLGenerateLoginModule  required;
       com.yourcompany.security.server.yourGenerateSAMLLoginModule required;
   };
   ```

   The com.yourcompany.security.server.yourGenerateSAMLLoginModule entry is your custom authentication module. The underlying security module calls the modules in the specified order when a WS-Security-enabled callout web service receives a synchronous callout request. The default SAMLGenerateLoginModule module is invoked first followed by the yourGenerateSAMLLoginModule module.
3. Store your compiled custom module .class file in the *install_dir*`/imssoap/WEB-INF/classes` directory. For JAR files that contain classes files, store them in the *install_dir*`/imssoap/WEB-INF/lib` directory.

Related reference:

Refer to the IBM WebSphere Application Server Version 8 information center for the SecurityToken API Javadoc information.
For more detail about the SecurityToken API, see the Javadoc information in the IBM WebSphere Application Server Version 8 information center.

Refer to the IBM WebSphere Application Server Version 8 information center for the SAMLToken API Javadoc information.
For more detail about the SAMLToken API, see the Javadoc information in the IBM WebSphere Application Server Version 8 information center.

Business event scenario

IMS applications can emit business events to IBM business event engines, such as IBM WebSphere Business Events and IBM WebSphere Business Monitor, for business activities processing and monitoring through SOAP Gateway.

A business event is the representation of a business activity that occurs inside or outside the enterprise or business. An event is a notification to signal a problem, an opportunity, a threshold, or a deviation. The occurrence of the event can trigger activities in another business application, or it can simply trigger data movement.

Event data is data that is included with the emitted event.

WebSphere Business Events

WebSphere Business Events is a business event processing engine that supports advanced event processing features for detecting, evaluating, correlating, and responding to events and complex event patterns. Often times, businesses need complex event processing and management capability to detect missing events, late events, or specific event correlation for identification and prevention of risks or to comply with regulations. Business owners also need a high-level view of their line of business or their companies through graphical user interfaces to better manage their environment and event processes.

WebSphere Business Monitor

WebSphere Business Monitor is a comprehensive business activity monitoring solution that provides the capability to measure business performance, monitor inflight and completed processes, and report on business operations. WebSphere Business Monitor provides visibility into the performance of business activities by processing events, calculating business metrics, and presenting key performance indicators through business dashboards. It enables customers to identify business problems, correct exceptions, and change processes to achieve a more efficient and competitive business. It helps when something goes wrong. Alerts can be delivered to make an organization aware of potential problems and proactively take directed action. It can monitor business events from any application via a variety of protocols.

Processing and monitoring IMS business activities and business logic

The support for IMS applications to emit event data to business events processing engines enables IMS assets to participate in business event processing solutions. With the use of the graphical tooling interface and runtime engines in WebSphere Business Events and WebSphere Business Monitor, business users can define and manage business events, proactively monitor IMS business activities, extend the use of IMS business logic, identify new business opportunities, and mitigate risks.

Related tasks:

Chapter 7, “Enabling an IMS application to emit a business event,” on page 271
To enable an application to emit a business event, you must modify your IMS application, define an OTMA destination descriptor, generate the correlator file, the XML converter, and the data mapping XSD file, and configure SOAP Gateway for the business event server.

Business events processing flow

IMS applications can emit business events placing the event data on an IMS OTMA hold queue for SOAP Gateway to retrieve and emit to the business event processing engine by using either the SOAP or REST protocol.

The business events support in SOAP Gateway works in a similar fashion as the asynchronous callout function. Business event data is placed in an OTMA hold queue by using the ISRT ALTPCB call in the IMS application to be retrieved by SOAP Gateway. Depending on how the event processing engine expects the event data to be sent, the correlation file that provides information about transaction and runtime properties and about how to match the event data between IMS and the external business event processing engine.

The following figure describes the business event process flow.

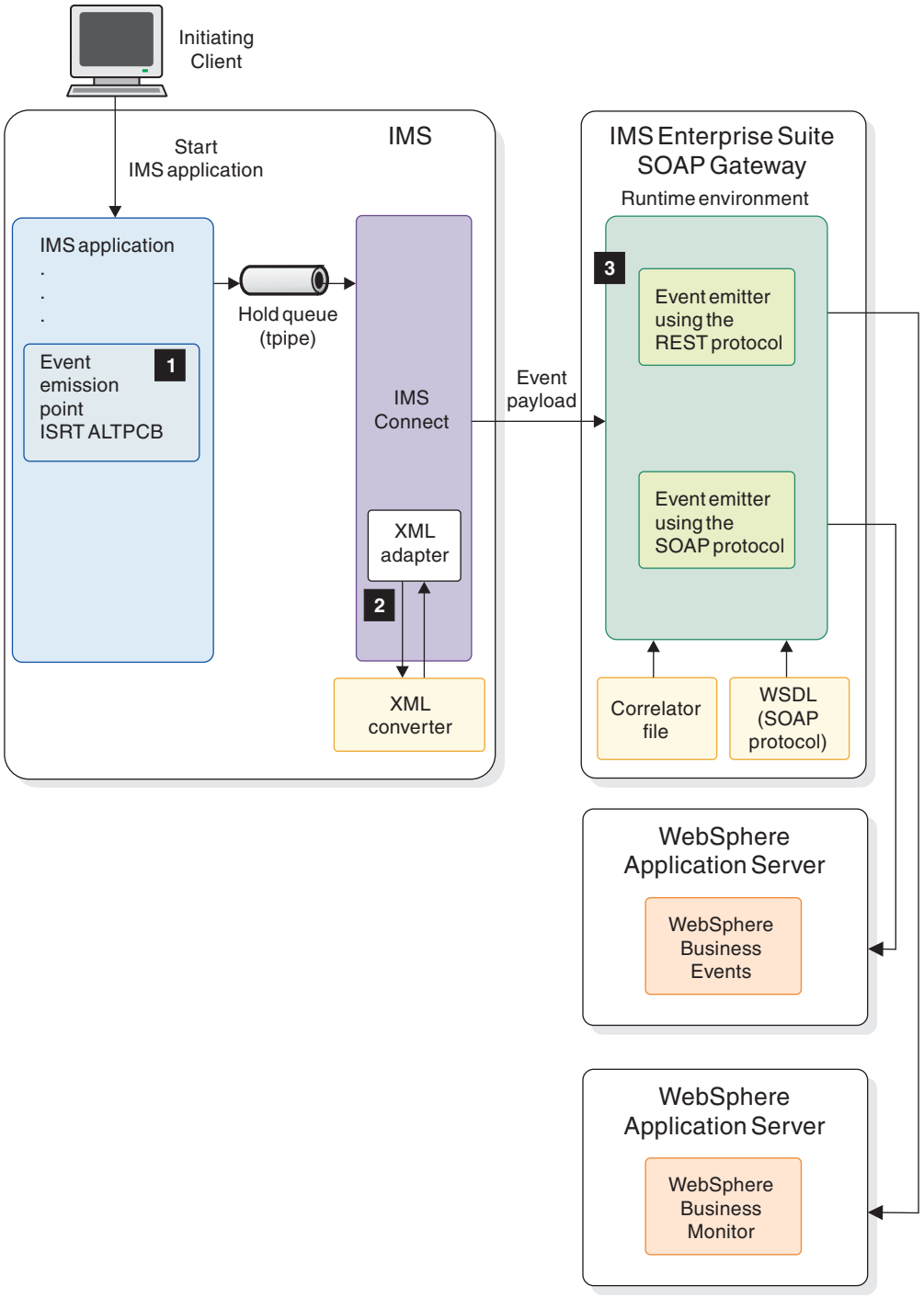


Figure 38. SOAP Gateway process flow when an IMS application is enabled to emit business event data to a business event monitoring engine

The process flow is as follows:

1. An IMS application emits business event data by issuing an ISRT ALTPCB call to an OTMA destination descriptor, which contains the destination tpipe name. The message that contains the business event data is queued in this tpipe.

2. The IMS Connect XML adapter function converts the business event data from bytes to XML by using the XML converter that is generated by Rational Developer for System z from the IMS application source file.
3. The SOAP Gateway callout thread retrieves the business event message from the tpipe. Based on the correlator file that is generated by Rational Developer for System z, SOAP Gateway emits the business event by using either the SOAP protocol or the REST protocol.

How SOAP Gateway emits business events to WebSphere Business Events

To emit business events to WebSphere Business Events, SOAP Gateway uses the one-way asynchronous callout function. SOAP Gateway consumes the WSDL file that is generated by WebSphere Business Events at deployment time. At run time, SOAP Gateway takes the WSID value, or the service name and operation name values, from the callout request message data, loads the correlator file, and then loads the WSDL file. The URL address for WebSphere Business Events is taken from the WSDL file and SOAP Gateway emits the business event payload XML data by using the SOAP protocol.

How SOAP Gateway emits business events to WebSphere Business Monitor

To emit business events to WebSphere Business Monitor, SOAP Gateway asynchronously sends the business event by using the REST protocol. REST provides a lightweight XML API that communicates through HTTP. At run time, SOAP Gateway takes the WSID value from the callout request message data and loads the correlator file. The CalloutURI property in the correlator file specifies the URL address for WebSphere Business Monitor. SOAP Gateway uses this URL address to communicate and emits the event data to WebSphere Business Monitor.

Related information:

WebSphere Business Events information center
WebSphere Business Events information center

WebSphere Business Monitor information center
WebSphere Business Monitor information center

Design guidelines for emitting business events

You might want to add new fields to your data structure or generate the ALTPCB value in the PSB, depending on how the events are being processed and whether the IMS application has access to an ALTPCB.

OTMA guidelines

- You can use an OTMA destination descriptor or code the OTMA routing exits to set the routing destination information. The OTMA destination descriptor is recommended because the descriptor has a simple and straightforward format, and does not require the knowledge of routing exits.
- Whenever possible, use different tpipe hold queues for synchronous callout, asynchronous callout, and business event emission messages.

Application design guidelines

- The IMS application that emits business event needs access to an ALTPCB. If the application currently does not have access to an ALTPCB, you must generate the program specification block, application control block, and complete an online

change. In general, use a modifiable ALTPCB for emitting the events. The application would first issue a CHNG call to set the destination and then issue an ISRT call to emit the events. If the ALTPCB is not modifiable, the destination can be set by having the ALTPCB value generated in the PSB before the ISRT call. The application simply issues the ISRT call to emit the event.

- The event is sent out after the synchronization point completes. If the event needs to be sent out before synchronization point, an express ALTPCB is needed to emit the event.
- When creating the data structure for emitting the business event, you might reuse an existing structure, modify an existing one, or create a new one. Some fields, such as a timestamp to indicate the time the event is emitted, and an ID for your business event, are common for business event processing. Consider these common fields and add them to the data structure with your business data.

Security for business event requests

Security support is provided for business event messages from IMS to SOAP Gateway through Secure Sockets Layer (SSL), and from SOAP Gateway to the business event servers through HTTPS.

Security of event messages can be set up in similar fashion as in the asynchronous callout scenario where IMS applications issue asynchronous callout request messages.

- You can set up resume tpipe security with IMS Connect and OTMA to ensure that only authorized users can retrieve event messages from the tpipe. You can set the user ID and password for resume tpipe security by using the SOAP Gateway connection bundle.
- You can set up Secure Sockets Layer (SSL) between IMS Connect and SOAP Gateway to secure the event message that is sent from IMS. Set the SSL properties by using the SOAP Gateway connection bundle.
- You can set up HTTPS between SOAP Gateway and a business event processing engine such as WebSphere Business Monitor and WebSphere Business Events to secure the event message sent from SOAP Gateway.

  Tip: For WebSphere Business Monitor, when HTTPS is set up for server authentication, it must be configured with basic authentication.

Because event data is emitted by using the same approach as for an asynchronous callout request, the same considerations and steps apply to business events.

Related concepts:

"Security for the consumer (callout) scenario" on page 182
Security support for the callout scenario is provided for messages from IMS to SOAP Gateway through SSL, and from SOAP Gateway to the web service through HTTPS.

"Combined server authentication and basic authentication security scheme" on page 188
With combined server authentication and basic authentication, the client (SOAP Gateway) that requests the web service is required to establish HTTPS security before it sends the basic authentication credentials to be authenticated.

"Security support in SOAP Gateway" on page 30
SOAP Gateway supports HTTPS communication with its clients, and SSL communications with its host, IMS Connect.

Chapter 5. Enabling an IMS application as a web service provider

To enable an IMS application as a web service provider, you must have a Web Services Description Language (WSDL) file, determine how to handle message conversion between XML and bytes, and deploy the web service by providing connection and data correlation information.

For an existing IMS application in COBOL or PL/I, you can use IBM Rational Developer for System z to generate the required WSDL, XML converter, and correlator. This approach is known as the *bottom-up* development scenario in Rational Developer for System z.

Rational Developer for System z also supports the *top-down* development scenario, with which you can generate an Enterprise PL/I-based application from a WSDL file. You then modify the application to add your business logic and enable it as a web service on SOAP Gateway.

You can also use the *meet-in-middle* development scenario, where you define the mappings between high level language data structures from your application and the web service WSDL file.

Table 28. Source and generated files for different development scenarios in Rational Developer for System z

File to start with	Development scenario to use	Files generated
• A web service WSDL file • Generation properties files that describe the web service – `Container.xml` – `ServiceSpecification.xml` – `PlatformProperties.xml`	Top-down	• A PL/I application template. • One or more files containing the drivers and converters. One converter driver is generated for each operation in the WSDL file. • The correlator file (.xml)
COBOL copybook or PL/I source file	Bottom-up	• The web service description file (.wsdl) • The correlator file (.xml) • The runtime XML converter driver (.cbl)
• Web service WSDL file to generate the request mapping session file. • COBOL copybook or PL/I source file to generate the response mapping session file.	Meet-in-middle	• The correlator file (.xml) • The file that contains the web service driver and runtime XML converter (.cbl)

To enable an IMS application as a web service:

1. Prepare the web service WSDL file, message converters (XML converter drivers), and data correlation files based on your development approach. Detailed steps for the top-down and the bottom-up scenarios are provided.

- "Top-down: Creating an IMS PL/I application from a WSDL file"
- "Bottom-up: Creating a web service from an IMS COBOL or PL/I application" on page 216

2. For security-related support, design, setup, and considerations, see the section on "Security for the web service provider scenario" on page 123.
3. Deploy the XML converter in IMS Connect. For more information about compiling and binding the XML converter, see "Configuring the IMS Connect XML adapter function" on page 223.

   If you are not using the XML conversion function in IMS Connect, you must specify in your correlator file that no XML adapter function is used. By default the adapter type is set to IBM XML Adapter. You must use the SOAP Gateway management utility iogmgmt -corr command to set the adapter type to `No_Adapter`.
4. Deploy a web service.
5. Write a client application to access IMS applications.

Related concepts:

"Security for the web service provider scenario" on page 123
SOAP Gateway provides support for both server authentication and client authentication and web-services security (WS-Security) for the web service provider scenario regardless of the platform that SOAP Gateway runs on.

Related information:

A sample on how to enable an IMS applications as a web service provider from the IMS Enterprise Suite SOAP Gateway web page.
Download the sample from the IMS Exchange web site.

Top-down: Creating an IMS PL/I application from a WSDL file

Use the batch processor in Rational Developer for System z to generate the IMS web service provider application in Enterprise PL/I.

For IMS Version 11, the generated PL/I top-down XML converter drivers require IMS V11 APAR PM16945.

The batch processor in Rational Developer for System z V9.0.1 or later generates PL/I application templates that are based on the segmentation APIs (DFSPWSIO) in IMS V12 (APAR PM97469) and IMS V13 (APAR PI17898). If you have IMS V11, use Rational Developer for System z V9.0. This version uses the segmentation APIs that are included in the IRZPWSH module in the SFEKSAMP data set in Rational Developer for System z.

Related information:

DFSPWSIO segmentation APIs in IMS
This document describes the DFSPWSIO APIs added to IMS V12 and IMS V13.

Batch processor and WSDL to PL/I mapping

The batch processor in Rational Developer for System z generates the data structure, XML converters, and PL/I application template file from the WSDL file that describes the web service.

The batch processor is a command-line utility and requires three generation properties files as the input:

Table 29. Generation properties files for the top-down PL/I application generation approach

File	Description
`Container.xml`	The top-level generation properties file is used by the batch processor to create web services implementation artifacts and message converters.
`PlatformProperties.xml`	This file specifies the default options properties that reflect your target runtime environment. The options affect the processing of the language types that are used in producing XML schema descriptions of web service messages that are based on that language type.
`ServiceSpecification.xml`	This file specifies the options required to generate new web service interfaces or web service implementations. You can also override certain options that you specify in the `PlatformProperties.xml` file.

WSDL to PL/I mapping

The WSDL2PLI component in Rational Developer for System z generates metadata to record the high-level relationships between the WSDL file that you supply and the WSDL2PLI-generated artifacts. Language structures are written to a single include file that begins with an operation-to-language-structure dictionary comment. The metadata file is in XML format and is used by the batch processor to generate XML converters, deployment metadata, and template programs.

Annotations are added to the generated source code to describe the relationships between the generated language structures and the XML schemas from which they are derived. The annotations appear as language comments immediately preceding the definitions of the language structures or language structure members to which they apply.

In Rational Developer for System z V9.0, the WSDL2PLI component uses a set of segmentation APIs in the IRZPWSH include file that is required during compilation of the PL/I program. Starting Rational Developer for System z V9.0.1, the WSDL2PLI component uses the segmentation APIs in the DFSPWSH include file that is part of IMS V12 (APAR PM97469) and IMS V13 (APAR PI17898). These APIs define how to consume and produce IMS messages.

The IRZPWSH module (in the SFEKSAMP data set in Rational Developer for System z V9.0 or earlier version) or DFSPWSH module (in the SDFSSMPL data set in IMS), provides the PL/I binding and offers pointers to the data structures.

For each operation on the specified service and port, WSDL2PLI generates:

- The PL/I structure(s) for operation input message
- The XSD to PL/I mapping session for operation input message
- The PL/I structure(s) for operation output message
- The PL/I to XSD mapping session for operation output message

WSDL2PLI generates the following output:

- The WSDL2ELS metadata that is used to generate the IMS application template, IMS correlator file, and the XML converters.
- A log file

Finally, the batch processor generates the following files:

- The XML converter driver for each operation in the WSDL2ELS metadata for the input and output messages.
- A multi-operation IMS correlator file
- An IMS service provider application template

Rational Developer for System z offers flexibility in the converter file naming convention to avoid possible name collisions for multiple conversion source packages. A conversion source package is generated for each language data structure. By default, the WSDL name is used as the converter file name prefix. If the name of the input WSDL file is longer than 6 characters, the first 6 characters are used to form the default name of the converter, with an added letter D at the end. If the WSDL file name ends with a number, the number increments for each additional conversion source package and is encoded in hexadecimal values to maximize the number of unique file names that can be formed within the 8 character limit.

You can also specify SOAP faults messages for each operation. The operation-level fault messages can be used to report business logic related issues, such as invalid account number. If a custom SOAP fault message is specified in the WSDL file, a converter is generated for the fault message.

Rational Developer for System z supports additional generic PL/I binding for SOAP header XML pass-through. A structure is created in the generated application template for the client application to access and parse the SOAP header content. The client application is responsible for parsing and processing the header information, if there is any.

The following compatibility table describes the version of Rational Developer for System z that you should use depending on the IMS version, and the differences between the WSDL-to-PL/I segmentation APIs in the two versions.

Table 30. WSDL-to-PL/I segmentation APIs location and compatibility

IMS version	Rational Developer for System z version	WSDL-to-PL/I segmentation API module name	Data set that contains the module
IMS V11	Rational Developer for System z V9.0	IRZPWSIO in Rational Developer for System z, with the IRZPWSH include file	SFEKSAMP in Rational Developer for System z
IMS V12 and IMS V13	Rational Developer for System z V9.0.1 or later	DFSPWSIO in IMS, with the DFSPWSH include file	SDFSSMPL in IMS

For more information about the DFSPWSIO segmentation APIs in IMS, see the Information changes for APAR PM97469 and APAR PI17898 document.

For details about how XML schema data types are mapped to Enterprise PL/I data declarations, limitations and restrictions, the IRZPWSH segmentation APIs, and troubleshooting information, see the WSDL2PLI reference information in the Rational Developer for System z online help, or the Rational Developer for System z information center.

Preparing the generation properties files for the top-down PL/I development approach

You must first create three generation properties files to describe the web service and then run the Rational Developer for System z batch processor to generate Enterprise PL/I top-down artifacts, such as an PL/I application template, language structures, XML converter driver(s), and the correlator file.

The best way to create these XML files is to modify the sample generation properties that are provided. The sample generation properties files are located at: *plugins_directory**ui_directory*\BatchProcessorSamples\EISServiceImplementation where:

- *plugins_directory* is the complete path to the named plugin subdirectory within the product installation directory, for example, C:\Program Files\IBM\SDPShared\plugins
- *ui_directory* is the directory com.ibm.etools.est.ui_*rrrrrr.vyyyymmdd_hhmm*
  - *rrrrrr* is a one-to-eight-character release number.
  - *vyyyymmdd_hhmm* is a time stamp that indicates the year, month, day, hour, and minute.

For example, the complete file path for the sample directories in Windows might be: C:\Program Files\IBM\SDPShared\plugins\com.ibm.etools.est.ui_*x.x.x.vyyyymmdd_xxxx*\BatchProcessorSamples\EISServiceImplementation

See the Rational Developer for System z online help or information for more information about preparing the generation properties files and the batch processor.

Running the command-line batch processor

Run the xsebatch.bat command to generate the web service artifacts such as the PL/I application template, XML converter driver(s), and the correlator file.

The batch processor is located in the *RDz_install_directory*/bin directory.

1. Run the xsebatch.bat command and specify where the Container.xml file that describes the web service is located.

```
xsebatch.bat -f "Fully_Qualified_PathTo/Container.xml" -c -d "Fully_Qualified_PathTo/workspace" -verbose
```

   - -c *containerFile* indicates to generate the set of language converters, the drivers, and XML schemas based on the provided container file. You can override this option by using the generateConverters and the generateSeparateXSD options in the Container.xml file and in the ServiceSpecification.xml file.
   - -d *workspace* indicates the path to the workspace to be used for the import. In Rational Developer for System z, specify either a relative path or a fully qualified absolute path to the workspace.
   - -verbose causes the diagnostic messages to be printed to the console.

   Refer to the Rational Developer for System z online help for more information about the batch processor command syntax and usage.
2. After the xsebatch.bat program finishes running, restart Rational Developer for System z to view the generated files in the workspace.

You are ready to modify the generated PL/I application template and add your business logic.

Adding business logic to the generated PL/I template

For each operation in the WSDL, an *operationName*Handler procedure and an *operationName*Impl procedure are created in the generated template. The *operationName*Handler procedure contains protocol logic while the *operationName*Impl procedure is ready to be filled out and customized with your business logic.

For example, a service WSDL file has an import statement that references operation definitions from an XSD file.

```
<wsdl:types>
   <xs:schema targetNamespace="http://www.fastbank.com/FAST247/">
     <xs:import namespace="http://www.interface.fastbank.com/FAST247/"
          schemaLocation="Operations.xsd">
      </xs:import>
  </xs:schema>
</wsdl:types>
```

In the XSD file, the checkBalaneRequest and checkBalanceresponse operations are defined:

```
<xs:complexType name="balance">
  <xs:sequence>
   <xs:element name="status" type="xs:boolean" />
   <xs:element name="amount" type="xs:decimal" />

  </xs:sequence>
 </xs:complexType>

 <xs:complexType name="checkBalance">
  <xs:sequence>
   <xs:element name="accountno" type="xs:long" />
  </xs:sequence>
 </xs:complexType>

 <xs:element name="checkBalanceRequest" type="account:checkBalance" />
 <xs:element name="checkBalanceResponse" type="account:balance" />
```

For each operation in the wsdl:binding section, there is an input message, an output message, and a fault message:

```
<wsdl:operation name="CheckBalanceOperation">
  <soap:operation soapAction="CheckBalanceOperation" style="document" />
  <wsdl:input name="CheckBalanceRequest">
   <soap:body parts="CheckBalancePart" use="literal" />
  </wsdl:input>
  <wsdl:output name="CheckBalanceResponse">
   <soap:body parts="CheckBalancePart" use="literal" />
  </wsdl:output>
  <wsdl:fault name="ServiceExceptionFault">
   <soap:fault use="literal" name="ServiceExceptionFault" />
  </wsdl:fault>
 </wsdl:operation>
```

This operation is specified in the correlator entry:

```
<correlatorEntry operationName="CheckBalanceOperation" portName="FAST247Port"
serviceName="FAST247Service">
```

These input and output elements are directly reflected in the generated PL/I application template. Add necessary business logic in the implementation procedure for each operation. The generated PL/I application template uses the DFSQGETS and DFSQSETS APIs to retrieve and set the SOAP body language structure from the IMS message queue.

The *operationName*Handler procedure checks for and retrieves the @dfs_soap_header structure before retrieving the request body structure. If the dfs_soap_header_ptr pointer is null, it means that no SOAP header element is present in the request SOAP message. You can implement application-specific processing of the information that is contained in the SOAP header in the *operationName*Impl procedures.

The following sample shows a generated *operationName*Impl procedure. The dfs_soap_header_ptr pointer information is available to the PutOperationImpl procedure.

```
CheckBalanceOperationImpl: procedure(iopcb_mask_ptr, dfs_soap_header_ptr,
 checkBalanceRequest_ptr, checkBalanceResponse_ptr, ServiceException_ptr)
 internal;

   dcl iopcb_mask_ptr pointer byvalue;
   dcl dfs_soap_header_ptr pointer byvalue;
   dcl checkBalanceRequest_ptr pointer byvalue;
   dcl checkBalanceResponse_ptr pointer byaddr;
   dcl ServiceException_ptr pointer byaddr;
   ...
   return;

 end CheckBalanceOperationImpl;
```

You add your back-end business logic to process the SOAP header. If the dfs_soap_header_ptr pointer is not null, parse the @dfs_soap_header structure. The dfs_soap_header_ptr providers the pointer to the header data structure.

```
CheckBalanceOperationImpl: procedure(iopcb_mask_ptr, dfs_soap_header_ptr,
 checkBalanceRequest_ptr, checkBalanceResponse_ptr, ServiceException_ptr)
 internal;

   dcl iopcb_mask_ptr pointer byvalue;
   dcl dfs_soap_header_ptr pointer byvalue;
   dcl checkBalanceRequest_ptr pointer byvalue;
   dcl checkBalanceResponse_ptr pointer byaddr;
   dcl ServiceException_ptr pointer byaddr;

   /* Add your custom business logic and SOAP header processing code.
    * The PARSEHDR procedure is a user-supplied procedure that parses
    * the SOAP header by using a parser such as PLISAXA. It must be
    * declared before it is called.
    */
 if (dfs_soap_header_ptr ^= sysnull()) then do;
     call PARSEHDR(addr(@dfs_soap_header.header_xml(1)),
          @irz_soap_header.header_xml_cnt);
   end;

   /* Add your business logic for generating a Fault */
   if (checkBalanceRequest.accountno < 9167889 |
     checkBalanceRequest.accountno > 9167889) then do;

     allocate ServiceException set(ServiceException_ptr);
     ServiceException.faultcode = 'env:Client';
     ServiceException.faultstring = 'The specified account no '
       || trim(checkBalanceRequest.accountno)
       || ' is invalid or undefined.';
     ServiceException.faultactor = trim(packagename())
       || '#' || trim(procedurename());
     ServiceException.exCode = '10999';
     ServiceException.exDescription = 'The specified account no '
       || trim(checkBalanceRequest.accountno)
```

```
        || ' is invalid or undefined.';

    return;
  end;
```

After processing the SOAP header, handle the request message and the response message in the SOAP body. The following example shows the allocating of the structure that handles the response from the CheckBalance operation.

```
/* Allocate the response data structure for returning a response
allocate checkBalanceResponse
 set (checkBalanceResponse_ptr);

 checkBalanceResponse.amount = 987.50;
 checkBalanceResponse.status = '1'b;

 return;

end CheckBalanceOperationImpl;
```

Related information:

DFSPWSIO segmentation APIs in IMS
This document describes the DFSPWSIO APIs added to IMS V12 and IMS V13.

Custom SOAP headers with XML passthrough

Rational Developer for System z supports generic PL/I binding for SOAP header XML passthrough. In the generated IMS PL/I message processing program (MPP) template, you can implement application- or system-specific processing of the information that is contained in the SOAP header.

The SOAP header element of a SOAP message might contain zero or more blocks. Each block consists of a global XML element that is either of a simple or complex type. A SOAP message might contain undefined SOAP header blocks, and contents of the header blocks are not required to be defined within the WSDL file. Rational Developer for System z provides a generic passthrough binding of the XML that is contained within the SOAP header element, and therefore offers maximum flexibility in the implementation of passing and processing of information contained within the SOAP header.

The IMS PL/I top-down function detects and extracts the SOAP header content in request SOAP messages. The complete SOAP header element in UTF-8 XML markup is stored in an instance of the @dfs_soap_header structure that is defined in the DFSPWSH include file in IMS. A pointer to the @dfs_soap_header instance is passed by value in parameter dfs_soap_header_ptr to the implementation procedure for each operation. You can add your XML parsing code and business logic in each implementation procedure to process the elements in the SOAP header.

Restrictions

SOAP header XML passthrough is supported for incoming requests to IMS MPPs that are enabled as web services. Outbound passthrough of the SOAP header element is not supported.

Any namespace declarations outside of the SOAP header element is not preserved. Namespace declarations set on the SOAP header element, other than the SOAP namespace itself, is preserved.

For example, consider namespace declarations as follows:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:q0="http://services.my.company.com/inquiry/ping_1_0">
  <soapenv:Header>
    <p1:test1 xmlns:p1="http://www.example1.com" >
    <p2:test2 xmlns:p2="http://www.example2.com" >
  </soapenv:Header>
  <soapenv:Body>
    <q0:ping_1_0 />
  </soapenv:Body>
</soapenv:Envelope>
```

The soapenv prefix is bound to the same namespace that the original element was bound to. Therefore, for this example, the following information is passed through to the MPP:

```
<SOAPENV:Header xmlns:SOAPENV="http://schemas.xmlsoap.org/soap/envelope/">
   <p1:test1 xmlns:p1="http://www.example1.com">
   <p2:test2 xmlns:p2="http://www.example2.com">
</SOAPENV:Header>
```

Related information:

WSDL2PLI reference information in Rational Developer for System z V9 information center.

Custom SOAP Fault messages

With Rational Developer for System z, you can specify SOAP Fault messages for problems that occur during the execution of business logic.

Important: IMS Version 12 APAR PM43645, or IMS Version 11 APAR PM39865 is required for the IMS Connect XML Adapter function to dynamically select the correct converter package for the varying operation output.

SOAP Fault messages in the SOAP body

A WSDL author can specify an XSD that defines the layout for the details of a problem for custom SOAP fault messages. Rational Developer for System z supports the generation of the PL/I structures, the mapping sessions, and metadata for each fault message defined for an operation. The custom SOAP Fault messages are carried in the SOAP Body, different from the SOAP Header Faults.

The SOAP Fault element has four sub-elements:

1. `faultcode`: A code for identifying the fault
2. `faultstring`: A human-readable explanation of why the Fault occurred
3. `faultactor`: The URI associated with the actor that caused the Fault on the message path
4. `detail`: Application-specific information about why the error occurs

```
<env:Body xmlns:env="http://schemas.xmlsoap.org/soap/envelope/">
   <env:Fault>
      <env:faultcode>
         env:VersionMismatch | env:MustUnderstand | env:Client | env:Server
      </env:faultcode>
      <env:faultstring>
         // Human readable explanation of the fault
      </env:faultstring>
      <env:faultactor>
         // Who or what caused the fault
      </env:faultactor>
      <env:detail>
         // The contents of the detail element can be described using an XSD
         // Application-specific error information (such as wrong account number,
```

```
or invalid request).
      </env:detail>
   </env:Fault>
</env:Body>
```

The following is a simplified sample WSDL file.

1. The operation MyOperation defines two possible SOAP fault messages, MyOperationException and MySystemException, that can be issued when the operation is invoked.
2. The MyOperationException fault message is intended to be used for reporting problems at the business-logic level, such as invalid account number or invalid request.
3. The MySystemException fault message is intended for reporting issues with the provision of the service itself, such as internal application error, if the corresponding fault data structure has been allocated in the message processing program (MPP).

```
<definitions xmlns="http://schemas.xmlsoap.org/wsdl/" ...>
    <types>
        <schema ...>
            <xs:element name="MyOperationRequest" type="p0:OperationRequestData" />
            <xs:complexType name="OperationRequestData">
                < ... />
            </xs:complexType>
            <xs:element name="MyOperationResponse" type="p0:OperationResponseData" />
            <xs:complexType name="OperationResponseData">
                < ... />
            </xs:complexType>
            <xs:element name="MyOperationException" type="p0:OperationException" />
            <xs:complexType name="OperationException">
                <xs:sequence>
                    <xs:element name="op_error_code" type="xs:string" />
                    <xs:element name="op_trace_entry" type="xs:string"
                           minOccurs="0" maxOccurs="unbounded" />
                    </xs:sequence>
            </xs:complexType>
            <xs:element name="MySystemException" type="p0:SystemException" />
            <xs:complexType name="SystemException">
                <xs:sequence>
                    <xs:element name="sys_status_code" type="xs:string" />
                    <xs:element name="sys_status_message" type="xs:string" />
                    <xs:element name="sys_admin_email" type="xs:string" />
                    <xs:element name="sys_log_entry" type="xs:string"
                           minOccurs="0" maxOccurs="unbounded" />
                </xs:sequence>
            </xs:complexType>
        </schema>
    </types>
    <message name="MyOperationRequest">
        <part name="parameters" element="p0:MyOperationRequest" />
    </message>
    <message name="MyOperationResponse">
        <part name="parameters" element="p0:MyOperationResponse" />
    </message>
    <message name="MyOperationException">
        <part name="parameters" element="p0:MyOperationException" />
    </message>
    <message name="MySystemException">
        <part name="parameters" element="p0:MySystemException" />
    </message>
    <portType name="MyServicePortType">
1       <operation name="MyOperation">
           <input message="p0:MyOperationRequest" />
           <output message="p0:MyOperationResponse" />
2            <fault message="p0:MyOperationException" name="MyOperationException" />
3            <fault message="p0:MySystemException" name="MyOperationException" />
        </operation>
```

```
    </portType>
    <binding name="MyServiceBinding" type="p0:MyServicePortType">
        <soap:binding style="document"
            transport="http://schemas.xmlsoap.org/soap/http" />
            <operation name="MyOperation">
                <soap:operation
                    soapAction="http://www.example.org/MyService/MyOperation" />
                <input>
                    <soap:body use="literal" />
                </input>
                <output>
                    <soap:body use="literal" />
                </output>
                <fault name="MyOperationException">
                    <soap:fault use="literal" name="MyOperationException" />
                </fault>
                <fault name="MySystemException">
                    <soap:fault use="literal" name="MySystemException" />
                </fault>
            </operation>
    </binding>
    <service name="MyService">
        <port binding="p0:MyServiceBinding" name="MyServicePort">
            <soap:address location="http://server:port/imssoap/services/MyService" />
        </port>
    </service>
</definitions>
```

For this WSDL file, the WSDL to PL/I mapping (WSDL2PLI) component would generate the PL/I structures, mapping sessions, and metadata for each fault message defined in the MyOperation operation. Multiple operations can share the same fault message. The IMS Connect XML adapter function would dynamically choose the converter package to use for the output message and include the appropriate Fault details for each operation.

The generated PL/I template includes the PL/I structure for each of the varying output:

```
/*********************************************************************
 * REFER objects language structure "MyOperationException_ref" for r
 * esponse SOAP Fault language structure "MyOperationException" of b
 * inding operation(s) "MyOperation, MySecondOperation".
 *********************************************************************/
DCL 01 MyOperationException_ref UNALIGNED,
 /* @LIMIT MyOperationException.op_trace_entry
  */
 02 op_trace_entry_lim SIGNED FIXED BINARY(31);

/*********************************************************************
 * POINTER language structure "MyOperationException_ptr" for respons
 * e SOAP Fault language structure "MyOperationException" of binding
 *  operation(s) "MyOperation, MySecondOperation".
 *********************************************************************/
/* @POINTER MyOperationException
 */
DCL 01 MyOperationException_ptr POINTER;

/*********************************************************************
 * Response SOAP Fault language structure "MyOperationException" for
 *  binding operation(s) "MyOperation, MySecondOperation".
 *********************************************************************/
/* @XPATH Fault/Detail/MyOperationException
 */
DCL 01 MyOperationException UNALIGNED BASED(MyOperationException_ptr),
 /* @XPATH Fault/faultcode
  */
 02 faultcode CHAR(64) VARYING,
 /* @XPATH Fault/faultstring
  */
```

```
 02 faultstring CHAR(64) VARYING,
 /* @XPATH Fault/faultactor
  */
 02 faultactor CHAR(64) VARYING,
 /* @LIMIT MyOperationException.op_trace_entry
  */
 02 op_trace_entry_lim SIGNED FIXED BINARY(31),
 /* @XPATH Fault/Detail/MyOperationException/op_error_code
  */
 02 op_error_code CHAR(64) VARYING,
 /* @COUNT MyOperationException.op_trace_entry
  */
 02 op_trace_entry_cnt SIGNED FIXED BINARY(31),
 /* @XPATH Fault/Detail/MyOperationException/op_trace_entry
  */
 02 op_trace_entry (MyOperationException_ref.op_trace_entry_lim REFER (
op_trace_entry_lim)) CHAR(64) VARYING;

/*********************************************************************
 * REFER objects language structure "MySystemException_ref" for resp
 * onse SOAP Fault language structure "MySystemException" of binding
 *  operation(s) "MyOperation, MySecondOperation".
 *********************************************************************/
DCL 01 MySystemException_ref UNALIGNED,
 /* @LIMIT MySystemException.sys_log_entry
  */
 02 sys_log_entry_lim SIGNED FIXED BINARY(31);

/*********************************************************************
 * POINTER language structure "MySystemException_ptr" for response S
 * OAP Fault language structure "MySystemException" of binding opera
 * tion(s) "MyOperation, MySecondOperation".
 *********************************************************************/
/* @POINTER MySystemException
 */
DCL 01 MySystemException_ptr POINTER;

/*********************************************************************
 * Response SOAP Fault language structure "MySystemException" for bi
 * nding operation(s) "MyOperation, MySecondOperation".
 *********************************************************************/
/* @XPATH Fault/Detail/MySystemException
 */
DCL 01 MySystemException UNALIGNED BASED(MySystemException_ptr),
 /* @XPATH Fault/faultcode
  */
 02 faultcode CHAR(64) VARYING,
 /* @XPATH Fault/faultstring
  */
 02 faultstring CHAR(64) VARYING,
 /* @XPATH Fault/faultactor
  */
 02 faultactor CHAR(64) VARYING,
 /* @LIMIT MySystemException.sys_log_entry
  */
 02 sys_log_entry_lim SIGNED FIXED BINARY(31),
 /* @XPATH Fault/Detail/MySystemException/sys_status_code
  */
 02 sys_status_code CHAR(64) VARYING,
 /* @XPATH Fault/Detail/MySystemException/sys_status_message
  */
 02 sys_status_message CHAR(64) VARYING,
 /* @XPATH Fault/Detail/MySystemException/sys_admin_email
  */
 02 sys_admin_email CHAR(64) VARYING,
 /* @COUNT MySystemException.sys_log_entry
  */
 02 sys_log_entry_cnt SIGNED FIXED BINARY(31),
```

```
 /* @XPATH Fault/Detail/MySystemException/sys_log_entry
  */
 02 sys_log_entry (MySystemException_ref.sys_log_entry_lim REFER (sys_l
og_entry_lim)) CHAR(64) VARYING;
```

For more information about the support for SOAP Fault messages and related restrictions, see the Rational Developer for System z information center.

Related information:

WSDL2PLI reference information in Rational Developer for System z V9 information center.

Compiling the PL/I application

Use a JCL to compile and link the PL/I application.

Modify the data set name in the JCL to a pre-allocated data set. The following is a sample JCL.

Important: For Rational Developer for System z Version 9.0, replace IMS.SDFSSMPL with FEK.SFEKSAMP, and the IMS.SDFSRESL module name with FEK.SFEKLMOD. Version 9.0 generates the template based on the segmentation APIs in the IRZPWSH module that is included in Rational Developer for System z.

```
//MYMPP JOB 'Z PROGRAMMER',MSGCLASS=H,REGION=0M,TIME=1444,
// MSGLEVEL=(1,1),NOTIFY=&SYSUID
//*************************************************************
//* COMPILE AND LINK-EDIT PL/I TOP-DOWN PL/I MPP
//*************************************************************
//RDZXML EXEC PROC=IBMZCB,LIBPRFX='SYS1'
//PLI.SYSIN   DD DSN=ZPROG.IMS.PLI(WSPOC1),DISP=SHR
//PLI.SYSLIB  DD DSN=ZPROG.IMS.PLI.INCLUDE,DISP=SHR
//            DD DSN=IMS.SDFSSMPL,DISP=SHR
//            DD DSN=&LIBPRFX..SCEESAMP,DISP=SHR
//PLI.SYSLIN  DD DSN=ZPROG.IMS.PLI.OBJECT(WSPOC1),DISP=SHR
//BIND.OBJECT DD DSN=ZPROG.IMS.PLI.OBJECT,DISP=SHR
//BIND.RESLIB DD DSN=IMS.SDFSRESL,DISP=SHR
//BIND.SYSLIN DD *
     INCLUDE OBJECT(WSPOC1)
     INCLUDE RESLIB(DFSPWSIO)
/*
//BIND.SYSLMOD DD DSN=IMS.PGMLIB(WSPOC1),DISP=SHR
/*
```

- The following artifacts must be available to the compiler:
  - PL/I top-down include file generated by Rational Developer for System z
  - PL/I top-down MPP generated by Rational Developer for System z and completed by a programmer
  - PL/I top-down API include file provided in IMS in IMS.DFSSMPL(DFSPWSH).
  - Language Environment® macro library in SYS1.SCEESAMP.
- The following artifacts must be available to the binder/linker:
  - PL/I top-down MPP object code.
  - PL/I top-down API object code provided in IMS.SDFSRESL(DFSPWSIO).

Compiling the PL/I top-down converter

Modify the generated JCL files to specify the data set names and run the JCL to compile and link the PL/I application.

The Rational Developer for System z also generates JCL files that you can modify to compile and link edit your PL/I application and the XML converter programs (which are also PL/I programs). SYSIN, SYSLIN and SYSLIB DD cards are provided in the generated JCL.

For IMS Version 11, the generated PL/I top-down XML converter drivers require IMS V11 APAR PM16945.

Modify the data set name in the JCL files to a pre-allocated data set. The following is a sample JCL for Rational Developer for System z Version 8.5 or later.

```
//WSPOC1D JOB 'Z PROGRAMMER',MSGCLASS=H,REGION=0M,TIME=1444,
// MSGLEVEL=(1,1),NOTIFY=&SYSUID
//*************************************************************
//* COMPILE AND LINK-EDIT PL/I TOP-DOWN XML CONVERTER
//*************************************************************
//RDZXML EXEC PROC=IBMZCB,LIBPRFX='SYS1',
//   LNGPRFX='ENPLI.V4R10'
//PLI.SYSIN   DD DSN=ZPROG.IMS.PLI(WSPOC1D),DISP=SHR
//PLI.SYSLIB  DD DSN=ZPROG.IMS.PLI.INCLUDE,DISP=SHR
//        DD DSN=FEK.SFEKSAMP,DISP=SHR
//           DD DSN=&LIBPRFX..SCEESAMP,DISP=SHR
//PLI.SYSLIN  DD DSN=ZPROG.IMS.PLI.OBJECT(WSPOC1D),DISP=SHR
//BIND.OBJECT DD DSN=ZPROG.EST.IMS.PLI.OBJECT,DISP=SHR
//BIND.SYSLIB DD
//           DD DSN=FEK.SFEKLMOD,DISP=SHR
//BIND.RESLIB DD DSN=FEK.SFEKLMOD,DISP=SHR
//BIND.SYSLIN DD *
     INCLUDE OBJECT(WSPOC1D)
     INCLUDE RDZLIB(IRZPWSIO)
     ENTRY WSPOC1D
     ALIAS WSPOC1X
     NAME WSPOC1D(R)
//*
//BIND.SYSLMOD DD DSN=IMS.HWS.XMLXLIB(WSPOC1D),DISP=SHR
```

- The following artifacts must be available to the compiler:
  - PL/I top-down API include file that is provided in FEK.SFEKSAMP(IRZPWSH)
  - Language Environment macro library in SYS1.SCEESAMP
- The PL/I top-down API object code that is provided in FEK.SFEKLMOD(IRZPWSIO).must be available to the binder or linker

After you compiled and link-edited the PL/I top-down XML converter driver(s):

1. Examine "Security for the web service provider scenario" on page 123, and set up for server authentication, client authentication, and web services security (WS-Security).
2. Configure the XML converter driver(s) in IMS Connect.
3. Deploy the web service.

Bottom-up: Creating a web service from an IMS COBOL or PL/I application

Use the Enterprise Service Tools wizard in IBM Rational Developer for System z to generate the WSDL file that is needed to enable your IMS application to run as a web service provider.

WSDL files are used by the client that invokes the service to discover the service and to understand how to invoke the service. The WSDL file specifies the location of the service and the operations that the service exposes.

Generating the WSDL file from an application in IBM Rational Developer for System z

Use the Enterprise Service Tools wizard in IBM Rational Developer for System z to generate the WSDL file, the correlator file, and the XML converter driver that are needed to enable your IMS application to run as a web service.

You must have a COBOL or PL/I copybook that describes the format of the input and output messages for your IMS application.

To generate the WSDL file:

1. Start Rational Developer for System z.
2. Open the Enterprise Service Tools perspective by clicking **Window** > **Open Perspective** > **Other**. The Open Perspective window opens.
3. Select **Enterprise Service Tools** and click **OK**. The EST Project Explorer tab displays.
4. Right-click in the EST Project Explorer window and click **New** > **IMS Enterprise Suite SOAP Gateway Project**.

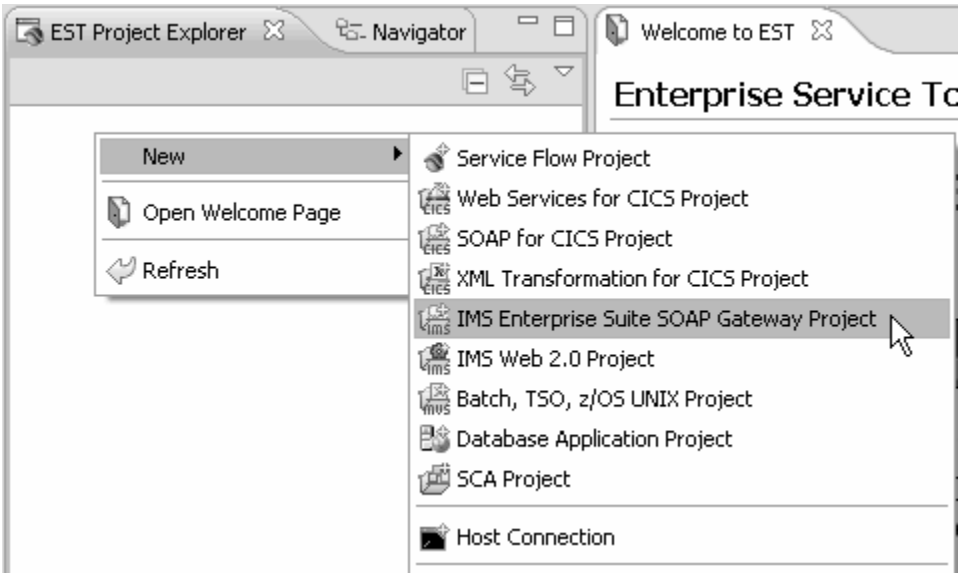


Figure 39. Creating a new IMS Enterprise Suite SOAP Gateway project

5. On the New IMS Enterprise Suite SOAP Gateway Project page, type the project name and select a development scenario.
   a. In the **Project name** field, type the name of your project.
   b. Select the following options:
      - Development scenario: **Create New Service Interface (bottom-up)** (default)
      - Application mode: **Service Provider** (default)
      - Conversion type: **Compiled XML Conversion**
   c. Click **Next**.

6. On the Import source files page, depending on where the COBOL copybook or the PL/I source file that describes the format of the input and output messages of your IMS application is located, click **File System**, **Workspace**, or **Remote**.
7. Navigate to your COBOL copybook or PL/I source file and click **Open**. Your source file name displays in the Source files to import box.
8. Click **Finish**. A new project is created and displayed in the EST Project Explorer. The Enterprise Service Tools wizard launchpad opens and the Language structures page displays.
9. On the Language structures page of the wizard, specify the request, response, or both language structures.
   a. In the **Request Language Structure** tab, select the structures that comprise the input to your IMS application. The following figure shows a single-segment application where the input consists of only one language structure.

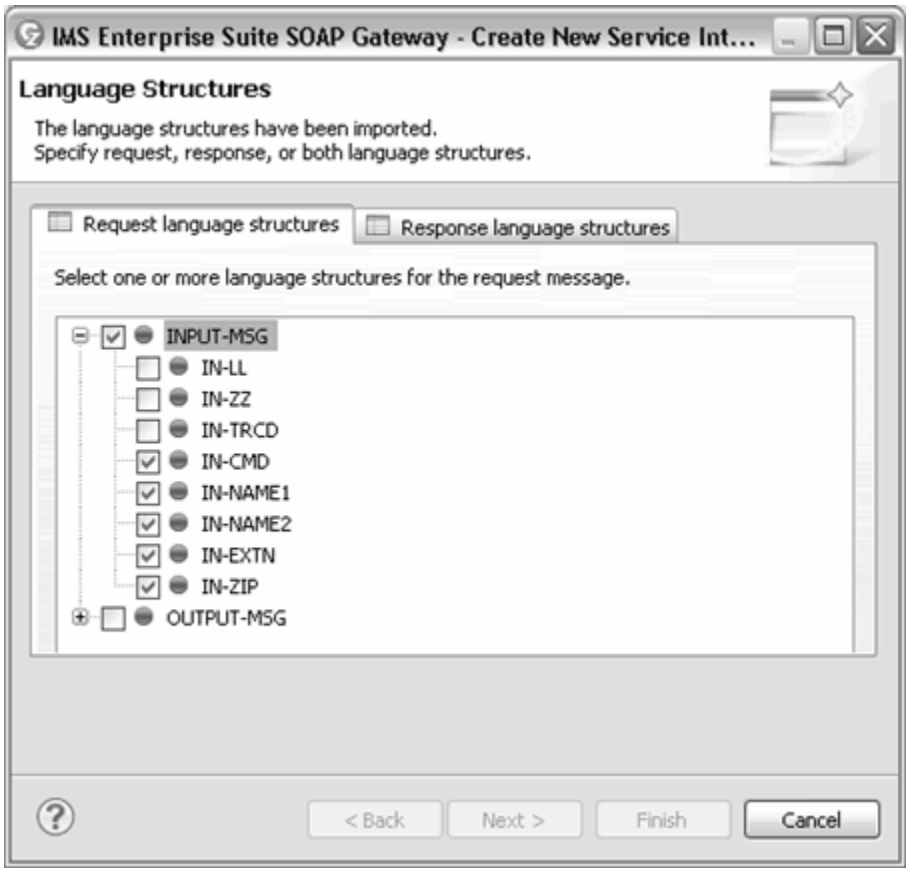


Figure 40. The language structures panel of the Enterprise Service Tools Wizard

The following figure shows the selection of multiple language structures for the request message to a multi-segment IMS COBOL application.

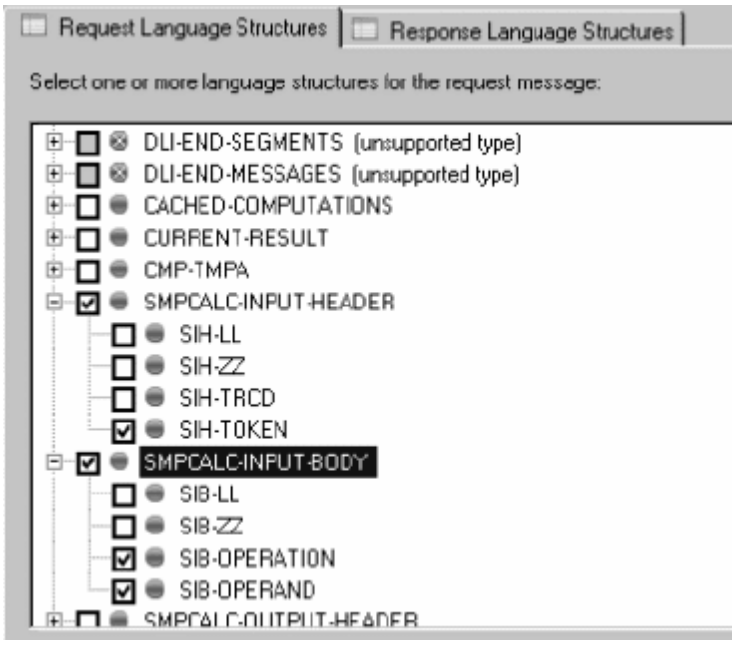


Figure 41. A sample of multiple language structures selection

Important: Each language structure describes an IMS message segment and, therefore, begins with LL and ZZ. Do not select the LL, ZZ, and trancode because doing so would expose implementation details in the web service interface. These fields are populated automatically by SOAP Gateway, IMS Connect, and the compiled XML converter.

b. In the **Response Language Structure** tab, select the structures that comprise the output to your IMS application.

Restrictions:

- Each IMS language structure cannot exceed 32 KB, the maximum length of an IMS message segment.
- For COBOL applications, at most 256 language structures can be selected for each of the requests and responses.
- For COBOL applications, inter-language structure dependencies are not supported. For example, OCCURS DEPENDING ON (ODO) subjects in selected language structures cannot specify an ODO object declared outside of the structure.

c. Click **Next**.

The IMS Message Layouts page opens. The language structures that you selected are added to the layout tables in the order in which they are selected.

10. Specify the layout of the request and response messages in detail. The layouts that are specified in these tables indicate how the XML and binary messages are structured. You must specify the order and repetition of the request and response language structures.

a. Indicate the order of the structure by selecting a structure and clicking **Move Up** and **Move Down** to change the order of the language structures in the XML and IMS runtime representations of the message.

b. For COBOL applications, specify the minimum and maximum numbers of times the pattern can repeat in the message. The following figure shows a multi-segment message where the language structure SMPCALC-INPUT-

BODY occurs 1 to 512 times after a single instance of the language structure SMPCALC-INPUT-HEADER in the request message.

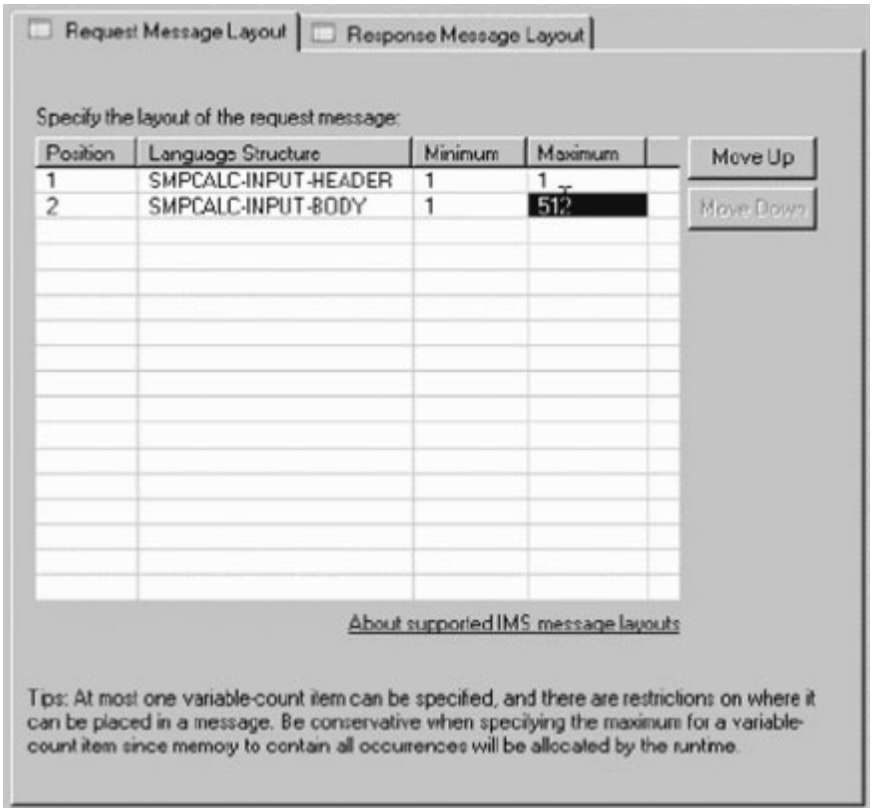
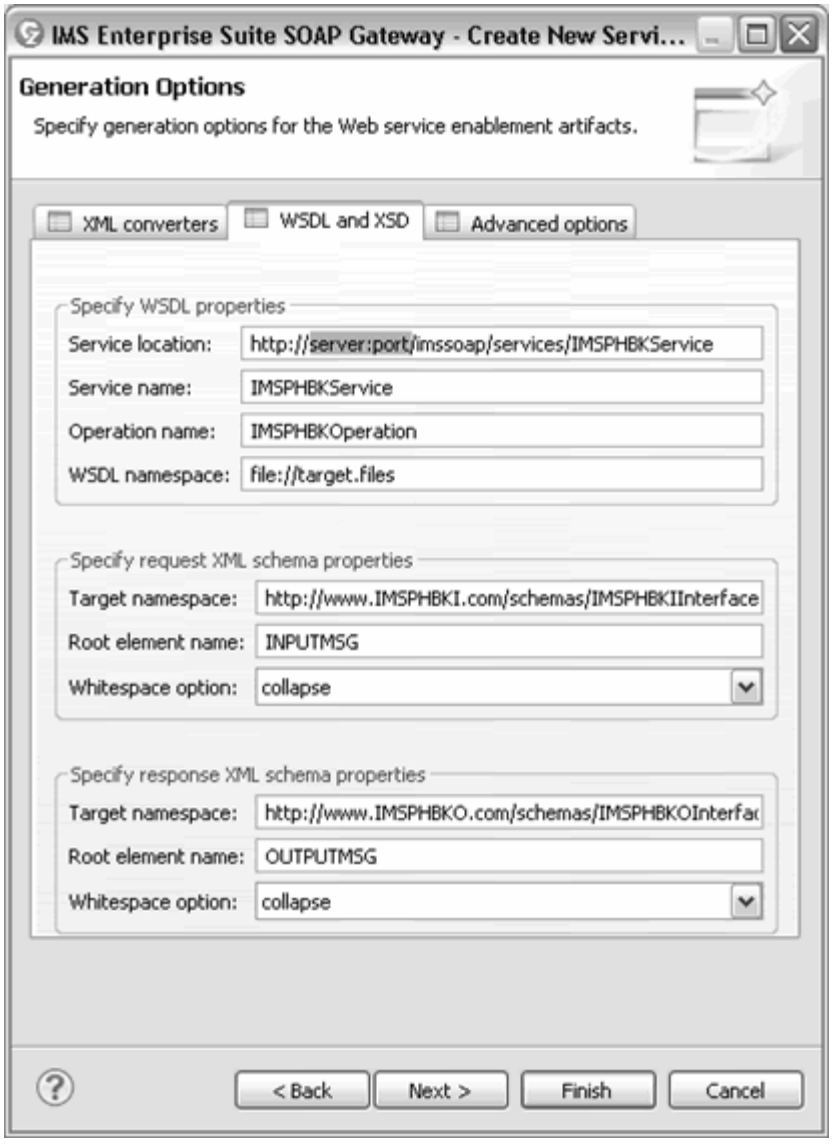


Figure 42. Specifying the order and count of each language structure in request and response messages

c. When both the request and response message layouts are specified, click **Next**.

11. On the Generation options page, specify the generation options for the web service artifacts.
    a. In the **XML Converters** tab, select or specify the following settings:
        1) Converter program name prefix: Specify the prefix that is used to generate the XML converter program.
        2) Host code page: Select the code page that the host uses.
        3) Request code page and response code page: Leave these fields at the default value, **1208 Unicode, UTF-8**. SOAP Gateway supports only UTF-8.
    b. In the **WSDL and XSD** tab, specify the properties for the WSDL file, and the request and response XML schema properties.



1) Service location: Change the server name and port number to the location of the SOAP Gateway server.
2) Leave all other fields unchanged.

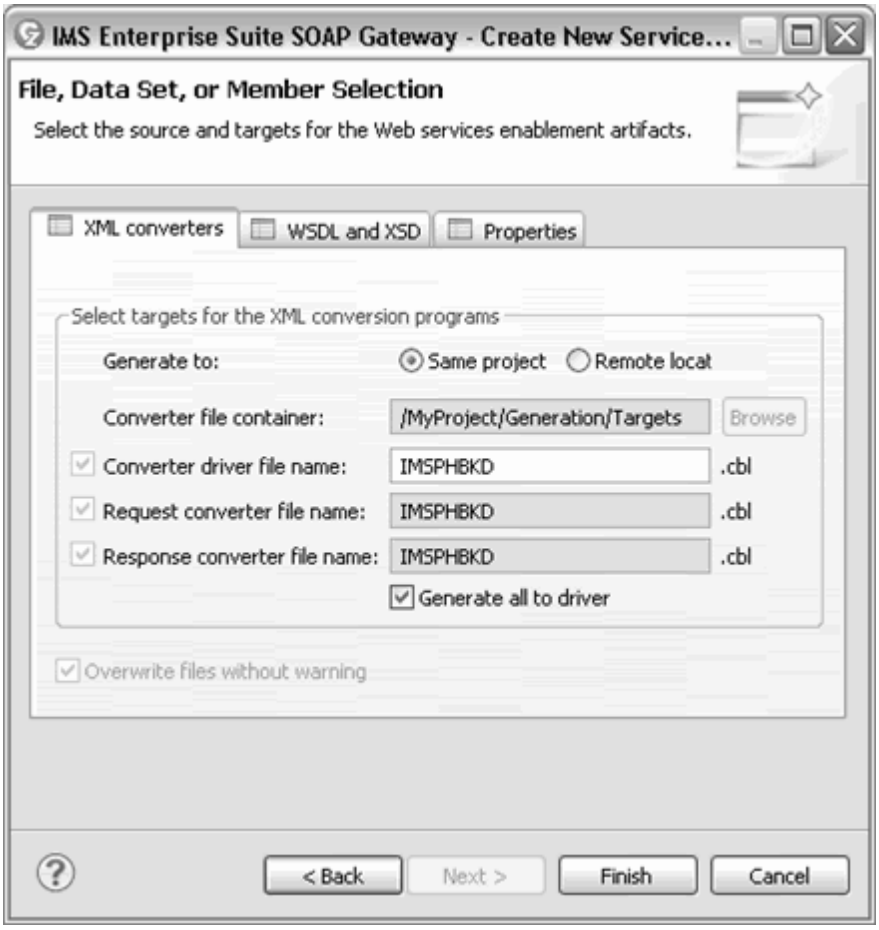
c. Click **Next**.

12. On the IMS Enterprise Suite SOAP Gateway Web Service Provider page, specify properties for defining the web service to SOAP Gateway.
    a. Specify any correlator properties and IMS system interaction properties for your SOAP Gateway environment. In particular, specify the transaction code, inbound connection bundle name, socket timeout value, and execution timeout value.

       Optionally, select **Enabled** for **WS-Security** if you want to enable web services security. When web services security is enabled, the client application passes user identity information with the request message. SOAP Gateway propagates the information to the IMS host system for user authentication.

    b. Click **Next**.

13. On the File, data set, or member selection page, select the source and targets for the web service artifacts.



a. In the **XML Converters** tab, specify the following options:
   1) Converter driver file name: Specify the name that you want the converter programs to be generated with.
   2) Click the **Generate all to driver** check box. This selection causes all the generated web service programs (driver, inbound converter, and outbound converter) to be placed in the same file.

b. In the **WSDL and XSD** tab, specify the following options:
   1) Ensure that the check box for **Input field WSDL file name** is selected.
   2) Type the WSDL file name.
   3) Clear the **Request XSD file name** and **Response XSD file name** check boxes.

      Selection of these check boxes generates the request and response data mapping XSD files, which are not used by SOAP Gateway.

c. Click **Finish**.

The following files are generated in the directories and file names that you specified:

- WSDL file (.wsdl)
- Correlator file (.xml)
- File that contains the web service driver and runtime XML converter (.cbl)

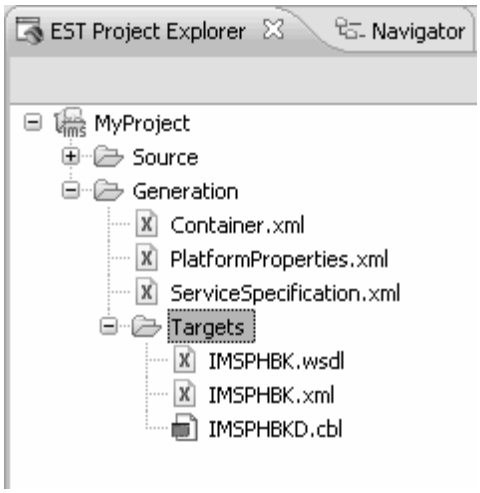


Figure 43. The generated WSDL file, correlator file, and XML converter

Related tasks:

"Deploying a web service" on page 229
Use the SOAP Gateway management utility to create a connection bundle and to deploy an IMS application as a web service.

Related information:

Creation of XML schemas from multiple language structures
For more information on how to create XML schemas from multiple language structures, see the Rational Developer for System z information center.

Configuring the IMS Connect XML adapter function

In order to handle the XML data from the client, you can either modify the IMS application to accept the XML input message and to return an XML output message, or use the IBM Rational Developer for System z XML converter drivers to transform the XML data in IMS Connect.

One of the features of the SOAP Gateway is the ability to allow clients to send and receive IMS transaction input and output messages in XML format without the need of changing the backend IMS application. This feature is made possible by the XML adapter function in IMS Connect.

The IMS Connect XML adapter function allows the XML adapter to run inside IMS Connect to perform the XML transformation in the IMS Connect address space. To handle IMS transaction input and output messages in XML format from the SOAP client, the XML adapter converts the XML-tagged data to the appropriate IMS application format that the IMS application accepts.

If you are not using the IMS Connect XML adapter function, you must modify your IMS application to handle XML formats.

If the IMS application is a multi-segment message processing program (MPP), you must use Rational Developer for System z to generate the XML converter.

The following steps provide a high-level description of how to configure the IMS Connect XML adapter function with SOAP Gateway. For detailed instructions on

setting up the IMS Connect XML adapter function, see the SOAP Gateway Phone Book sample, which can be downloaded from the IMS Enterprise Suite support site or the IMS Exchange website.

1. Configure the XML adapter function in IMS Connect. For more information, see IMS Version 13 System Definition information.
2. Use Rational Developer for System z to generate a WSDL file, a correlator file, and the XML converter driver program for your IMS application. The Rational Developer for System z SFEKLMOD module must be concatenated to the STEPLIB of the IMS Connect startup JCL.

   To compile and bind the converters into a data set concatenated with the IMS Connect STEPLIB:

   - The converter must have an alias that is linked with the converter code, using the same name as the converter, with an X suffix. If the converter name is eight-character long, the last character for the alias must be changed to an X.
   - The converter and the load module name must end with the letter D.
   - The converter should be linked with the option REUS=SERIAL so that the converter is loaded into the memory only once.

   For example, if your converter file name is IMSSGWS1D. Your JCL would contain ENTRY, ALIAS, and NAME statements as follows:

   ```
   //LINK    EXEC PGM=HEWL,COND=(4,LT),
   //        PARM='XREF,COMPAT=MIN,REUS=SERIAL'
   ...
     ENTRY IMSSGWS1D
     ALIAS IMSSGWS1X
     NAME IMSSGWS1D(R)
   ```
3. Upload the XML converter driver that was generated by Rational Developer for System z as part of the WSDL generation to your host IMS machine. Compile and link edit the program file such that it can be accessed by IMS Connect. The data set with the compiled converters needs to be concatenated to the STEPLIB with the IMS RESLIB data set. Because RESLIB requires APF authorization, all data sets concatenated with it must also be APF-authorized.
4. Deploy the WSDL and correlator file to SOAP Gateway by using the SOAP Gateway management utility.

Related tasks:

"Configuring IMS Connect for SOAP Gateway" on page 101
You must configure IMS Connect to allow SOAP Gateway to access IMS transactions.

Configuring XML conversion support for IMS Connect clients (IMS Version 13)
For information about configuring XML conversion support for IMS Connect clients, see IMS V13 System Definition information.

Related information:

Rational Developer for System z Host Configuration Guide
For host configuration instructions, see the *Host Configuration Guide* for the version of Rational Developer for System z that you on the Rational Developer for System z library page.

Modifying the IMS application for XML messages

If you are *not* using the IMS Connect XML adapter function for data transformation, you must modify the IMS application by using the provided guidelines and samples.

Message prefix

The client and IMS application must send and receive messages in XML that matches the XML schema in the WSDL file. The message data does not need to include LL, ZZ, or transaction code information.

IMS requires:

- All transaction messages to be prefixed by a two-byte LL field and a two-byte ZZ field
- That the length of the transaction code field must be eight bytes or less for the first input message of the IMS application
- That the LL, ZZ and transaction code fields are not wrapped in XML tags

To allow your XML-formatted IMS message data to reach your IMS application, SOAP Gateway allows you to specify the transaction code value as a correlator property. At run time, SOAP Gateway adds the simple EBCDIC byte values of the transaction code and LL and ZZ fields to XML-formatted IMS data input message. In addition, SOAP Gateway adds the XML declaration, `<?xml version="1.0" encoding="utf-8"?>`, at the beginning of the XML-formatted IMS data input message.

SOAP Gateway supports XML messages encoded in UTF-8 only. The input XML transaction data inside the SOAP message must be encoded in UTF-8 and the output XML transaction data from the IMS application must be encoded in UTF-8 as well. In addition, the IMS application must be able to handle the XML transaction data from SOAP Gateway in UTF-8.

Each input message includes the XML input data, as well as the prefix LL, ZZ, transaction code, and the XML declaration added by SOAP Gateway. Each output message includes LL, ZZ, and the output XML data.

Tip: The XML schema can contain the LL, ZZ, and transaction code definitions, but the values are not used by IMS if you specify the transaction code value by using the SOAP Gateway correlator property.

Sample

The IMS application has the following COBOL copybook:

```
    01  INPUT-MSG.
    02  IN-LL          PICTURE S9(3) COMP.
    02  IN-ZZ          PICTURE S9(3) COMP.
    02  IN-TRCD        PICTURE X(10).
    02  IN-CMD         PICTURE X(8).
    02  IN-NAME1       PICTURE X(10).
    02  IN-NAME2       PICTURE X(10).
    02  IN-EXTN        PICTURE X(10).
    02  IN-ZIP         PICTURE X(7).

01  OUTPUT-MSG.
    02  OUT-LL       PICTURE S9(3) COMP VALUE +0.
    02  OUT-ZZ       PICTURE S9(3) COMP VALUE +0.
    02  OUT-MSG      PICTURE X(40) VALUE SPACES.
    02  OUT-CMD      PICTURE X(8) VALUE SPACES.
    02  OUT-NAME1    PICTURE X(10) VALUE SPACES.
    02  OUT-NAME2    PICTURE X(10) VALUE SPACES.
    02  OUT-EXTN     PICTURE X(10) VALUE SPACES.
    02  OUT-ZIP      PICTURE X(7) VALUE SPACES.
    02  OUT-SEGNO    PICTURE X(4) VALUE SPACES.
```

The following schema, which maps to the COBOL copybook, is in the WSDL file:

```
<schema attributeFormDefault="qualified"
   elementFormDefault="unqualified"
   targetNamespace="http://ims.sample/"
   xmlns="http://www.w3.org/2001/XMLSchema" xmlns:xsd1="http://ims.sample/">
   <complexType name="INPUTMSG">
       <sequence>
           <element name="in__ll">
               <simpleType>
                   <restriction base="short">
                       <minInclusive value="-999"/>
                       <maxInclusive value="999"/>
                   </restriction>
               </simpleType>
           </element>
           <element name="in__zz">
               <simpleType>
                   <restriction base="short">
                       <minInclusive value="-999"/>
                       <maxInclusive value="999"/>
                   </restriction>
               </simpleType>
           </element>
           <element name="in__trcd">
               <annotation>
                   <appinfo source="http://www.wsadie.com/appinfo">
                       <initialValue kind="SPACE"/>
                   </appinfo>
               </annotation>
               <simpleType>
                   <restriction base="string">
                       <maxLength value="10"/>
                   </restriction>
               </simpleType>
           </element>
           <element name="in__cmd">
               <annotation>
                   <appinfo source="http://www.wsadie.com/appinfo">
                       <initialValue kind="SPACE"/>
                   </appinfo>
               </annotation>
               <simpleType>
                   <restriction base="string">
                       <maxLength value="8"/>
                   </restriction>
               </simpleType>
           </element>
           <element name="in__name1">
               <annotation>
                   <appinfo source="http://www.wsadie.com/appinfo">
                       <initialValue kind="SPACE"/>
                   </appinfo>
               </annotation>
               <simpleType>
                   <restriction base="string">
                       <maxLength value="10"/>
                   </restriction>
               </simpleType>
           </element>
           <element name="in__name2">
               <annotation>
                   <appinfo source="http://www.wsadie.com/appinfo">
                       <initialValue kind="SPACE"/>
                   </appinfo>
               </annotation>
               <simpleType>
                   <restriction base="string">
                       <maxLength value="10"/>
                   </restriction>
               </simpleType>
           </element>
           <element name="in__extn">
               <annotation>
                   <appinfo source="http://www.wsadie.com/appinfo">
                       <initialValue kind="SPACE"/>
                   </appinfo>
               </annotation>
               <simpleType>
                   <restriction base="string">
                       <maxLength value="10"/>
                   </restriction>
```

```
            </simpleType>
        </element>
        <element name="in__zip">
            <annotation>
                <appinfo source="http://www.wsadie.com/appinfo">
                    <initialValue kind="SPACE"/>
                </appinfo>
            </annotation>
            <simpleType>
                <restriction base="string">
                    <maxLength value="7"/>
                </restriction>
            </simpleType>
        </element>
    </sequence>
</complexType>
<complexType name="OUTPUTMSG">
    <sequence>
        <element name="out__ll">
            <annotation>
                <appinfo source="http://www.wsadie.com/appinfo">
                    <initialValue kind="string_value" value="+0"/>
                </appinfo>
            </annotation>
            <simpleType>
                <restriction base="short">
                    <minInclusive value="-999"/>
                    <maxInclusive value="999"/>
                </restriction>
            </simpleType>
        </element>
        <element name="out__zz">
            <annotation>
                <appinfo source="http://www.wsadie.com/appinfo">
                    <initialValue kind="string_value" value="+0"/>
                </appinfo>
            </annotation>
            <simpleType>
                <restriction base="short">
                    <minInclusive value="-999"/>
                    <maxInclusive value="999"/>
                </restriction>
            </simpleType>
        </element>
        <element name="out__msg">
            <annotation>
                <appinfo source="http://www.wsadie.com/appinfo">
                    <initialValue kind="SPACE"/>
                </appinfo>
            </annotation>
            <simpleType>
                <restriction base="string">
                    <maxLength value="40"/>
                </restriction>
            </simpleType>
        </element>
        <element name="out__cmd">
            <annotation>
                <appinfo source="http://www.wsadie.com/appinfo">
                    <initialValue kind="SPACE"/>
                </appinfo>
            </annotation>
            <simpleType>
                <restriction base="string">
                    <maxLength value="8"/>
                </restriction>
            </simpleType>
        </element>
        <element name="out__name1">
            <annotation>
                <appinfo source="http://www.wsadie.com/appinfo">
                    <initialValue kind="SPACE"/>
                </appinfo>
            </annotation>
            <simpleType>
                <restriction base="string">
                    <maxLength value="10"/>
                </restriction>
            </simpleType>
        </element>
```

```
                <element name="out__name2">
                    <annotation>
                        <appinfo source="http://www.wsadie.com/appinfo">
                            <initialValue kind="SPACE"/>
                        </appinfo>
                    </annotation>
                    <simpleType>
                        <restriction base="string">
                            <maxLength value="10"/>
                        </restriction>
                    </simpleType>
                </element>
                <element name="out__extn">
                    <annotation>
                        <appinfo source="http://www.wsadie.com/appinfo">
                            <initialValue kind="SPACE"/>
                        </appinfo>
                    </annotation>
                    <simpleType>
                        <restriction base="string">
                            <maxLength value="10"/>
                        </restriction>
                    </simpleType>
                </element>
                <element name="out__zip">
                    <annotation>
                        <appinfo source="http://www.wsadie.com/appinfo">
                            <initialValue kind="SPACE"/>
                        </appinfo>
                    </annotation>
                    <simpleType>
                        <restriction base="string">
                            <maxLength value="7"/>
                        </restriction>
                    </simpleType>
                </element>
                <element name="out__segno">
                    <annotation>
                        <appinfo source="http://www.wsadie.com/appinfo">
                            <initialValue kind="SPACE"/>
                        </appinfo>
                    </annotation>
                    <simpleType>
                        <restriction base="string">
                            <maxLength value="4"/>
                        </restriction>
                    </simpleType>
                </element>
            </sequence>
        </complexType>
        <element name="INPUTMSG" type="tns:INPUTMSG"/>
        <element name="OUTPUTMSG" type="tns:OUTPUTMSG"/>
    </schema>
```

The message to the IMS application then has the following format based on this schema:

```
<?xml version="1.0" encoding="UTF-8"?>
<INPUTMSG xmlns="">
 <in__cmd>DISPLAY</in__cmd>
 <in__name1>LAST1</in__name1>
 <in__name2></in__name2>
 <in__extn></in__extn>
 <in__zip></in__zip>
</INPUTMSG>
```

The message that the IMS application sends back to the client as the following format based on this schema:

```
<OUTPUTMSG xmlns="">
 <out__msg>ENTRY WAS DISPLAYED</out__msg>
 <out__cmd>DISPLAY</out__cmd>
 <out__name1>LAST1</out__name1>
 <out__name2></out__name2>
```

```
    <out__extn></out__extn>
    <out__zip></out__zip>
    <out__segno>0001</out__segno>
  </OUTPUTMSG>
```

Related concepts:

"XML-formatted IMS messages" on page 19
SOAP Gateway sends and receives messages in XML.

Related tasks:

"Writing a client application to access IMS applications" on page 231
Write a client application that sends a SOAP message to invoke the IMS application as a web service through SOAP Gateway.

Deploying a web service

Use the SOAP Gateway management utility to create a connection bundle and to deploy an IMS application as a web service.

Important: For security-related support, design, setup, and considerations, see the section on "Security for the web service provider scenario" on page 123.

To deploy a web service:

Related concepts:

"SOAP Gateway server startup options" on page 292
There are different methods to start the SOAP Gateway server depending on the host operating system.

Related tasks:

"Generating the WSDL file from an application in IBM Rational Developer for System z" on page 217
Use the Enterprise Service Tools wizard in IBM Rational Developer for System z to generate the WSDL file, the correlator file, and the XML converter driver that are needed to enable your IMS application to run as a web service.

Related reference:

"-deploy: Deploy a web service or callout application" on page 444
The -deploy command deploys a web service, callout application, or business event application to the active configuration of the SOAP Gateway server.

Creating a connection bundle entry and correlator file for the web service

Create a connection bundle entry and correlator file that defines the communication and connection properties between the client application, SOAP Gateway, IMS Connect, and IMS.

1. Create a correlator file.
   - If you used Rational Developer for System z to generate web service artifacts, the correlator file is automatically generated at the same time as the WSDL file and the XML converter driver. You must use the generated correlator file with the generated XML converter driver to use the XML adapter function in the target IMS Connect.
   - If you do not intend to use the XML adapter function, and you do not have Rational Developer for System z, you can create a correlator file by using the SOAP Gateway management utility `iogmgmt -corr -c` command.

   To create a correlator:

```
iogmgmt -corr -c -w wsdl_file -p operation_name
-i service_name -n connection_bundle -t trancode
```

Tip: By default, this entry is set to `IBM XML Adapter`. If you are not using the XML adapter function in IMS Connect, update your correlator file to specify that the adapter type is `No_Adapter`.

```
iogmgmt -corr -u -r correlator_name -p operation_name
 -i service_name -a No_Adapter
```

By specifying the `No_Adapter` value, the adapterType entry in the correlator is set to blank.

2. Create a connection bundle entry using the `iogmgmt -conn -c` command of the SOAP Gateway management utility.
3. Issue the `iogmgmt -corr -u -r service_name -i operation_name -n connection_bundle` command to update the correlator file with the name of the connection bundle entry created in step 2. You must specify the connection bundle entry name to allow SOAP Gateway to identify which set of connection properties to use for its associated web service.

Related reference:

"-conn: Create, update, or delete a connection bundle" on page 435
Use the -conn command to create, update, or delete a connection bundle.

"-corr: Create or update a correlator entry" on page 439
Use the -corr command to create or update the transaction and runtime properties of a correlator entry.

Deploying the web service

Deploying the web service to SOAP Gateway enables the application and allows it to begin processing client requests.

Before you deploy the web service, you must have the WSDL file name, correlator XML file name, and connection bundle entry name for the web service.

Tip: z/OS For z/OS systems, the correlator file and the WSDL file, when uploaded from a distributed platform, must be transferred in BINARY mode from your local workstation. Binary transfers provide a bit-by-bit copy that preserves the encoding on your system by instructing the FTP socket not to convert the encoding to the local system encoding (EBCDIC).

To deploy the web service:

1. Issue the command `iogmgmt -deploy -w wsdl_file -r correlator_file` with the name of the WSDL file and correlator XML file. The SOAP Gateway management utility parses the correlator XML file to determine the name of the linked connection bundle. If the connection bundle is not already active in the runtime cache, it is loaded when the service is deployed.
2. Optional: Specify a security token type (for example, `-t SAML11Token`) to enable web services security with SAML 1.1 unsigned tokens for the service.
3. Optional: Verify that the service is active with the `iogmgmt -view -correlatorfile ALL` command. The correlator file for the newly-deployed service appears in the list of active correlator files in the runtime configuration. If the server is stopped, the correlator file name appears in the list of correlator files in the master configuration instead.

The service WSDL and any referenced XSD files are included in the web service Axis Archive file (an AAR file) in the master configuration. The web service is made active in the runtime cache and the server can process client requests to the web service.

The next step is to create a client application to access the IMS application as a web service.

Related concepts:

"SOAP Gateway administrative console" on page 27
The SOAP Gateway administrative console lists the deployed web services when the server is started. Each item in the list is a link to the web services description language (WSDL) file for the web service.

Related tasks:

"Migrating correlator files to schema version 3.0" on page 302
IMS Enterprise Suite Version 3.1 SOAP Gateway requires correlator schema version 3.0. To migrate an existing correlator file from older versions to version 3.0, use the SOAP Gateway management utility iogmgmt -migrate correlator command.

Related reference:

"-migrate: Migrate and upgrade SOAP Gateway" on page 448
The -migrate command upgrades SOAP Gateway artifacts and settings to the latest version and generates a migration log.

Writing a client application to access IMS applications

Write a client application that sends a SOAP message to invoke the IMS application as a web service through SOAP Gateway.

The way in which you write the client application depends on whether you use client proxy code. If you do not have proxy code, the client application itself is responsible for these tasks. The client proxy code is usually generated by a tool or utility and performs the following tasks:

- Creates a connection to send and receive SOAP messages
- Wraps the transaction data in a SOAP message

When building the client application, ensure that the transaction data inside the SOAP message is in an XML format that is understood by either IMS Connect or the IMS application designed to process XML. The client and either IMS Connect or the IMS application must adhere to the format of the input and output message as described in the schema definition of the WSDL file.

Related reference:

"Modifying the IMS application for XML messages" on page 224
If you are *not* using the IMS Connect XML adapter function for data transformation, you must modify the IMS application by using the provided guidelines and samples.

Chapter 6. Enabling an IMS callout application as a web service consumer

To enable an IMS application as a web service consumer, you must modify your IMS application to issue callout requests, decide on your data transformation process, define an OTMA destination descriptor, generate the web service artifacts, and create a connection bundle.

SOAP Gateway supports both synchronous and asynchronous callout messages from IMS applications to external web services. For a synchronous callout request, the IMS application must wait in the dependent region for a response from the web service. The time to wait for the response is specified in the IMS call or in the OTMA destination descriptor that describes the routing of the messages.

Rational Developer for System z provides support for generating the web service artifacts to enable the IMS callout function in two approaches.

- The *meet-in-middle* approach generates the required converter driver and correlator file from the web service WSDL file and your IMS callout application by creating the message mapping files for both the input and the output messages. This approach can be used for both synchronous and asynchronous callout scenarios, and for COBOL and PL/I applications.
- The *top-down* approach generates the required converter driver, correlator file, and the COBOL copybook that contains data structures derived from the input and output messages of operations in the WSDL file. This approach is for synchronous callout only.

Ensure that you review the following topics before you proceed:

- "Thread management for callout messages retrieval" on page 174 and related concepts on correlating the callout messages and thread management
- "Security for the consumer (callout) scenario" on page 182 for supported security scenarios and considerations
- "Verifying the setup for the consumer (callout) usage scenario" on page 111 for environment setup verification

Tip: Any callout-related configuration and deployment changes that you make with the SOAP Gateway management utility take effect immediately.

Related concepts:

"Security for the consumer (callout) scenario" on page 182
Security support for the callout scenario is provided for messages from IMS to SOAP Gateway through SSL, and from SOAP Gateway to the web service through HTTPS.

Related information:

Samples on how to enable an IMS applications as a web service consumer are available from the IMS Enterprise Suite SOAP Gateway web page.
Download the samples from the IMS Exchange web site.

Modifying an IMS application for callout requests

You must modify your IMS application in order to place the callout request on the OTMA hold queue or route the request to an IMS Connect destination.

To modify your IMS application to issue a callout request:

1. Place the callout request in an IMS Connect destination that is defined in an OTMA destination descriptor.
   - For a synchronous callout request, your IMS application must issue an ICAL call to place the callout request on an IMS Connect destination that is defined in an OTMA destination descriptor.
   - For an asynchronous callout request, Your IMS application must specify an ISRT ALTPCB call to place the callout request on the OTMA asynchronous hold queue or to an IMS Connect destination that is defined in an OTMA destination descriptor.
2. If a response message is expected from the external application, correlate the response.
   - For a synchronous callout request, SOAP Gateway handles the correlation of the response to the corresponding callout request.
   - For a asynchronous callout request:
     - You might need to correlate the response back to the initial request message or to a different transaction for further processing.
     - You must specify the transaction code for the IMS transaction that will process the output response. In most cases, you should invoke a non-response mode IMS transaction to process your response message. If you specify a response mode transaction to process the response message, the output of the response mode transaction will be routed to the IOG$RESP tpipe.

   The best way to correlate the response in your IMS application might be to define some data, such as a message identifier or a unique request ID, in your callout request that can correlate with the initial input message. You might also have the IMS application program that issues the callout request store correlation data or other data in an IMS database for retrieval when the callout response is returned.

   For more information on the callout function and correlation of the data, see the “Callout requests from IMS” topic in IMS Version 13 Communications and Connections information.
3. See “Selecting the data transformation process for callout messages” for more information about why and how to decide on the data transformation process.
4. If you are not using the IMS Connect XML adapter to convert the callout request and response data between bytes and XML, you need to handle the data transformation in your application by preparing your own callout messages.

Selecting the data transformation process for callout messages

You must specify to SOAP Gateway how you want to map the data structures in your IMS application to that in the external web service. You must also determine how data transformation between bytes and XML will be handled.

Data mapping between web services and IMS applications

Data mapping is the process of relating parts of two existing data structures to one another. You can modify an existing IMS application or create a new IMS application to make the callout request message. If you are modifying an existing IMS application, because the data structure of the web service is already predefined, it is unlikely that the data structure of the web service will be exactly the same as the data structure that is used by the IMS callout application. Therefore, some data mapping must take place between the data structure of the IMS application that issues the callout and the input data structure of the external web service. Sometimes you must create a new data structure or modify an existing application.

IBM Rational Developer for System z provides an XML to COBOL Mapping wizard that lets you create a mapping definition between the data structure of the web service (in an XSD schema format) and the existing data structure of the IMS application (for example, in a COBOL copybook format). A converter is then generated based on the mapping definition to allow the mapping and transformation to be taken place at run time. You can perform data mapping as part of the process of generating the web service artifacts. This approach is called the *meet-in-middle* approach.

Rational Developer for System z Version 8.5.1 and later also provides COBOL data structure generation support from web service WSDL files. In addition to generating the required web service artifacts (correlator file and XML converter driver), the generated COBOL data structure can be used in your IMS COBOL synchronous callout application. This approach is called the top-down approach.

Each web service can have one or more operations and each operation can have its own input and output data definition. The data definition for each operation is defined as an XML Schema Definition (XSD) schema inside the WSDL file of the web service.

Data transformation

XML data transformation is optional for both the callout request and response. Because IMS applications send and receive data in bytes, you can use the IMS Connect XML adapter to convert the data between bytes and XML for you. If your application design requires that the data be sent in format other than bytes, you must handle the data transformation yourself.

If you specify that IMS Connect performs the data transformation by using the IMS Connect XML adapter, the IMS application's data is converted to an XML message that is designated for the specified web service operation.

Sending callout request and receiving response data in bytes

The IMS Connect XML adapter requires an XML converter for each web service operation the IMS application is calling out to. To create the XML converter:

1. Use Rational Developer for System z to map the XSD schema of the input data structure of the web service operation with the data structure of the IMS callout application. Rational Developer for System z uses this mapping definition to create both the correlator file and the XML converter.
2. Deploy the XML converter in IMS Connect.

The Rational Developer for System z converter adds correlation information, also known as the service data prefix, to the callout message. This information helps SOAP Gateway correlate the callout request message to the web service.

Sending callout request and receiving response data in other formats

If you must send the data in the callout request in other formats such as XML, you need to ensure that the callout request message is in an XML format that can be properly processed by SOAP Gateway and the target web service.

- You must ensure that the LL and ZZ fields in the IMS callout request message are not in the resulting XML message.
- You must ensure the service data prefix information is added to the callout request message.

Related concepts:

"Preparing callout messages"
If you are not using the IMS Connect XML adapter function to convert the data between bytes and XML, you must ensure that the callout message from your IMS application is in a valid XML format.

Related reference:

Format of synchronous callout messages (IMS Version 13)
For more information, see the format of synchronous callout messages information in IMS Version 13 Communications and Connections information.

Preparing callout messages

If you are not using the IMS Connect XML adapter function to convert the data between bytes and XML, you must ensure that the callout message from your IMS application is in a valid XML format.

Your IMS application must send the callout request message in an XML format that can be processed by SOAP Gateway and the target web service appropriately. You must add the `<IOGServiceData>` service data information to the callout request message.

Callout message XML schema sample

The following example is a sample callout message XML schema. The callout messages generated from Rational Developer for System z conform to this schema.

```
<schema targetNamespace=http://www.ibm.com/IMS/Callout
 xmlns:IMS="http://www.ibm.com/IMS/Callout" elementFormDefault="qualified"
 xmlns="http://www.w3.org/2001/XMLSchema">

<!-- Global element -->
  <element name="IOGCallout" type="IMS:IOGCalloutType"/>

  <!-- Message contents  -->
  <complexType name="IOGCalloutType">
  <sequence minOccurs="1" maxOccurs="1">
    <!-- Web service invocation information -->
    <element name="IOGServiceData" type="IMS:IOGServiceDataType"
     minOccurs="1"  maxOccurs="1"/>

  <!-- Outbound language structure expressed as XML -->
  <element name="IOGPayloadData" type="anyType" minOccurs="1"
     maxOccurs="1"/>
  </sequence>
  </complexType>
```

```
  <!-- Web serice invocation information -->
  <complexType name="IOGServiceDataType">
  <sequence>
    <!-- Typically the WSDL file name without the extension -->
    <element name="WSID" type="string" minOccurs="1" maxOccurs="1"/>

    <!-- Target namespace of WSDL -->
    <element name="Namespace" type="string" minOccurs="1" maxOccurs="1"/>

    <!-- Service container of Port of Operation to invoke -->
    <element name="Service" type="string" minOccurs="1" maxOccurs="1"/>

    <!-- Port container of Operation to invoke-->
    <element name="Port" type="string" minOccurs="1" maxOccurs="1"/>

    <!-- Operation to invoke on Port of Service -->
    <element name="Operation" type="string" minOccurs="1" maxOccurs="1"/>
  </sequence>
  </complexType>
</schema>
```

Tip: The values of the service name and operation name are used to determine the web service and the associated correlator.

Callout message example

The following example is a sample callout message.

```
<IMS:IOGCallout xmlns:IMS="http://www.ibm.com/IMS/Callout"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.ibm.com/IMS/Callout IOGCallout.xsd ">
  <IMS:IOGServiceData>
    <IMS:WSID>IMSPHBK</IMS:WSID>
    <IMS:Namespace>http://www.IMSPHBKI.com/IMSPHBKI</IMS:Namespace>
    <IMS:Service>IMSPHBKService</IMS:Service>
    <IMS:Port>IMSPHBKPort</IMS:Port>
    <IMS:Operation>IMSPHBKOperation</IMS:Operation>
  </IMS:IOGServiceData>
  <IMS:IOGPayloadData>
    <INPUTMSG
      xmlns="http://www.IMSPHBKI.com/schemas/IMSPHBKIInterface">
      <in_cmd>DISPLAY</in_cmd>
      <in_name1>last1</in_name1>
    </INPUTMSG>
  </IMS:IOGPayloadData>
</IMS:IOGCallout>
```

Sample IMS synchronous callout application

An IMS synchronous callout application must declare the IMS Application Interface Block (AIB) field for the ICAL call and the message processing program (MPP) request and response messages.

The following example demonstrates the required AIB field declaration for the ICAL call in the COBOL program.

```
01 AIB.
   02 AIBRID PIC x(8)  VALUE 'DFSAIB  '.
   02 AIBRLEN PIC 9(9) USAGE BINARY.
   02 AIBSFUNC PIC x(8) VALUE 'SENDRECV'.
   02 AIBRSNM1 PIC x(8) VALUE 'OTMADEST'.
   02 AIBRSNM2 PIC x(8).
   02 AIBRESV1 PIC x(8).
   02 AIBOALEN PIC 9(9) USAGE BINARY.
   02 AIBOAUSE PIC 9(9) USAGE BINARY.
   02 AIBRSFLD PIC 9(9) USAGE BINARY VALUE 10000.
```

```
   02 AIBRESV2 PIC x(8).
   02 AIBRETRN PIC 9(9) USAGE BINARY.
   02 AIBREASN PIC 9(9) USAGE BINARY.
   02 AIBERRXT PIC 9(9) USAGE BINARY.
```

The following example shows the declarations in the COBOL program for synchronous callout request (purchase order) and response messages (order confirmation).

```
01 PurchaseOrder.
   02 item-lim PIC S9(9) COMP-5.
   02 comment-lim PIC S9(9) COMP-5.
   02 orderDate-att PIC X(64) DISPLAY.
   02 shipTo.
      03 name PIC X(64) DISPLAY.
      03 street PIC X(64) DISPLAY.
      03 city PIC X(64) DISPLAY.
      03 state PIC X(64) DISPLAY.
      03 zip PIC X(64) DISPLAY.
   02 billTo.
      03 name1 PIC X(64) DISPLAY.
      03 street1 PIC X(64) DISPLAY.
      03 city1 PIC X(64) DISPLAY.
      03 state1 PIC X(64) DISPLAY.
      03 zip1 PIC X(64) DISPLAY.
   02 items.
      03 item
         OCCURS 0 TO 32 TIMES
         DEPENDING ON item-lim
         OF PurchaseOrder.
         04 partNum-att PIC X(64) DISPLAY.
         04 productName PIC X(64) DISPLAY.
         04 quantity PIC S9(31) COMP-3.
         04 USPrice PIC S9(25)V9(6) COMP-3.
         04 available PIC X DISPLAY.
         04 dateAvail PIC X(64) DISPLAY.
         04 comment PIC X(100) DISPLAY
            OCCURS 0 TO 2 TIMES
            DEPENDING ON comment-lim
            OF PurchaseOrder.

01 OrderConfirmation.
   02 comment-lim PIC S9(9) COMP-5.
   02 orderID PIC 9(9) COMP-5.
   02 itemCount PIC S9(31) COMP-3.
   02 USTotal PIC S9(25)V9(6) COMP-3.
   02 shipDate PIC X(64) DISPLAY.
   02 comment PIC X(100) DISPLAY
      OCCURS 0 TO 3 TIMES
      DEPENDING ON comment-lim
      OF OrderConfirmation.
```

Defining an OTMA destination descriptor for callout request messages

You can define an OTMA destination descriptor to route IMS callout requests to a hold queue, without the need to code assembler routing exits.

To use the OTMA destination descriptor:

1. Configure the descriptor in the DFSYDTx PROCLIB member.
2. Specify the ADAPTER and the CONVERTR values. Set the ADAPTER value to HWSXMLA0 if you want XML data transformation to be completed by IMS Connect. The following example OTMA destination descriptor spans multiple lines:

```
D SOAPGWAY TYPE=IMSCON TMEMBER=HWS2 TPIPE=HWS2SOAP
D SOAPGWAY ADAPTER=HWSXMLA0 CONVERTR=XMLCNVTR
```

- The descriptor for SOAPGWAY routes messages to IMS Connect target member HWS2 with tpipe HWS2SOAP.
- The ADAPTER is set to HWSXMLA0 for the data transformation to be performed by the IMS Connect XML adapter.
- The XML converter name is XMLCNVTR.

Important: Do not share the tpipe that you use for SOAP Gateway callout functions with business event data or callout functions in the IMS TM resource adapter.

Related concepts:

OTMA descriptors (IMS Version 13)
For more information, see the "OTMA descriptors" topic in IMS Version 13 Communications and Connections information.

Related reference:

DFSYDTx PROCLIB member data set for OTMA descriptors (IMS Version 13)
For more information, see the DFSYDTx PROCLIB member data set information in IMS Version 13 System Definition information..

Generating callout web service artifacts

You can generate the callout web service correlator file and the XML converter by using Rational Developer for System z. If you do not have Rational Developer for System z, you can use the SOAP Gateway management utility to create the correlator file.

Rational Developer for System z provides two development approaches for artifact generation:

- *Meet-in-middle*: This approach generates the XML converter and the correlator file from the web service WSDL file and the IMS callout application (synchronous or asynchronous, PL/I or COBOL).
- *Top-down*: This approach generates the XML converter, the correlator file, and the COBOL copybook from the web service WSDL file. This feature requires Rational Developer for System z Version 8.5.1 or later. The generated COBOL application contains the data structures that can be used in IMS COBOL synchronous callout applications.

Creating a correlator file for a callout application

You can manually create a correlator file with the SOAP Gateway management utility if you do not have IBM Rational Developer for System z.

A correlator file provides the information to invoke the outbound web service and return the response message back to IMS. The information includes:

- The WSDL file name for the web service or the WebSphere Business Events server to be invoked, or the XSD file name for the WebSphere Business Monitor server to be invoked
- The timeout value for waiting for a response from the web service or business event server
- The IMS transaction code and the connection bundle name for returning the callout response

Restrictions:

- The SOAP Gateway management utility does not support the creation of correlator files that work with an XML adapter in the target IMS Connect. Use IBM Rational Developer for System z instead.
- The SOAP Gateway management utility does not support the creation of correlator XML files for the WebSphere Business Monitor scenario. Use IBM Rational Developer for System z instead.

To create a correlator file:

1. Determine the specific callout scenario your SOAP Gateway callout application supports. A SOAP Gateway callout application supports one of the following scenarios:
   - An asynchronous call to a remote web service. A response may or may not be expected:
     - No response is expected from the target web service. This is also referred to as a "one-way" call. If the target server is expected to respond, it must invoke a separate SOAP Gateway web service.
     - For an asynchronous request-response application, SOAP Gateway returns the response to IMS Connect in a different transaction. In this case, you can optionally specify a separate connection bundle name, in addition to the normal callout connection bundle name, to handle the response message. If you specify only a callout connection bundle name, the response message is returned to the same host that issued the original callout request.
   - A synchronous call to a remote web service. A response from the target server is expected on the same connection as the request message. In this case, you can optionally specify a separate connection bundle name, in addition to the normal callout connection bundle name, to handle the response message. If you specify only a callout connection bundle name, the response message is returned to the same host that issued the original callout request.
   - A call to a business event monitoring server. SOAP Gateway can emit business event data to either WebSphere Business Events or WebSphere Business Monitor.
     - The WebSphere Business Events scenario is functionally the same as a one-way call to a remote web service. A WSDL file is required.
     - The WebSphere Business Monitor scenario uses the REST protocol and requires an XSD service definition file instead of a WSDL file. Additionally, the correlator XML file must be configured with the URI of the WebSphere Business Monitor server. Creation of correlator XML files for this scenario is only supported with IBM Rational Developer for System z.
2. Gather the information required to create a correlator for a callout application.
   a. Determine the WSDL or XSD file name for the application. The WSDL file contains the web service definition for a SOAP Gateway application. However, calls to WebSphere Business Monitor use the REST protocol and use an XSD file instead.
   b. Determine the service and operation names for the application. A WSDL file can contain multiple service and operation definitions, and so SOAP Gateway uses the combination of service and operation name as the unique identifier for the web service.
   c. Determine the callout connection bundle name. The callout connection bundle contains the properties that determine how SOAP Gateway handles message traffic that is sent from IMS to the target web service. Most

importantly, the callout connection bundle specifies one or more tpipes that are used for inbound and outbound messages.

d. Optional: For a request-response callout application, determine the connection bundle name. You can specify a non-callout connection bundle name (in addition to the required callout connection bundle name) in the correlator for a request-response callout application. This option allows you to configure how SOAP Gateway handles the response message in the same way that it handles incoming requests for web services.

e. Optional: Determine other correlation properties. A callout application correlator can contain other properties to override default interaction behaviors such as time out values. However, these properties are not required.

3. Create the correlator file by using Rational Developer for System z. If you are not using the XML adapter function, create the correlator file with the SOAP Gateway management utility `iogmgmt -corr -c` command.

   - For a one-way callout application correlator, or a synchronous request-response callout application correlator, issue the command `iogmgmt -corr -c -w wsdl_file -p operation_name -i service_name -d callout_connection_bundle_name`. You can specify multiple connection bundles for a synchronous callout correlator by separating the connection bundle names with commas: `iogmgmt -corr -c -w wsdl_file -p operation_name -i service_name -n connection_bundle_name -d callout_connection_bundle_name1,callout_connection_bundle_name2`.
   - For an asynchronous request-response callout application correlator, issue the command `iogmgmt -corr -c -w wsdl_file -p operation_name -i service_name -n connection_bundle_name -d callout_connection_bundle_name`.

   A new correlator XML file is created in the XML directory.

4. If you are not using the XML adapter function in IMS Connect, update the correlator and set the `-a` option (adapter type) to `No_Adapter`.

   ```
   iogmgmt -corr -u -r correlator file -p operation_name -i service_name  -a No_Adapter
   ```

   By default, this entry is set to `IBM XML Adapter`. By specifying `No_Adapter` for the **-a** option, the adapterType entry in the correlator is set to blank.

5. Optional: Save a copy of the correlator XML file outside of the XML directory. An undeploy operation deletes the associated correlator file for the web service from the XML directory. Saving a copy elsewhere preserves it when the associated web service is undeployed.

Related concepts:

"Correlator file" on page 22
The correlator file specifies transaction and runtime properties. This file also specifies the information that SOAP Gateway needs to match incoming requests to the appropriate backend IMS application and outgoing requests from an IMS application to a web service.

Related tasks:

Chapter 7, "Enabling an IMS application to emit a business event," on page 271
To enable an application to emit a business event, you must modify your IMS application, define an OTMA destination descriptor, generate the correlator file, the XML converter, and the data mapping XSD file, and configure SOAP Gateway for the business event server.

Related reference:

"-corr: Create or update a correlator entry" on page 439
Use the -corr command to create or update the transaction and runtime properties of a correlator entry.

Top-down: Generating callout web service artifacts for COBOL applications

The top-down approach for generating synchronous callout web service artifacts creates the required artifacts and COBOL data structures from the web service WSDL file, in similar fashion as the top-down PL/I support for the web service provider scenario.

Nested XSD import statements and Double Byte Character Set (DBCS) are supported.

The general steps are:

1. Identify the WSDL service, port, and operation(s) that your IMS application needs to invoke. The WSDL file must be local to the Rational Developer for System z installation.
2. Create top-down batch processor generation properties files, `Container.xml`, `PlatformProperties.xml`, and `ServiceSpecification.xml`. Specify the WSDL service, port, and a subset of the operations if needed.
3. Run the batch processor to generate the following for all or a subset of the operations:
   - COBOL copybook that contains 01-level structures for each input or output message.
   - Mapping session file for each input or output message.
   - COBOL XML converter for IMS synchronous callout request for each operation.
   - SOAP Gateway correlator file.

Optionally, you can use the graphical user interface of Rational Developer for System z (the **Generate COBOL Mapping** menu action) to generate COBOL mapping from the WSDL file. This approach might be useful if you prefer using a graphical user interface. However, this approach has the following behaviors and restrictions:

- This menu option generates only the mapping session files and the COBOL data structures, not the correlator file or converter driver. You must then follow the meet-in-middle approach based on the mapping session files to generate the correlator file and the converter driver.
- If multiple operations are involved, in the graphical user interface you must manually map one operation at a time.
- This option does not allow customization of the COBOL copybook that is to be generated. The **Generate COBOL Mapping** menu action behaves as if default values were specified for all of the attributes on the ServiceSpecification.xml WSDL2ELSSpec element that affect data structure generation, except for the suppressStructureComments, suppressPresenceFields, and suppressCounterFields attributes, which are all forced to true.

Related reference

WSDL2ELSSpec reference in Rational Developer for System z Version 8.5 Information Center

Batch processor and WSDL to COBOL mapping

The batch processor in Rational Developer for System z Version 8.5.1 or later generates a COBOL copybook, XML converters, and a correlator file from the WSDL file that describes the web service.

The batch processor is a command-line utility and requires three generation properties files as the input:

Table 31. Generation properties files for the top-down COBOL data structure and synchronous callout service artifact generation approach

File	Description
Container.xml	The top-level generation properties file is used by the batch processor to create web services implementation artifacts and message converters.
PlatformProperties.xml	This file specifies the default options properties that reflect your target runtime environment. The options affect the processing of the language types that are used in producing XML schema descriptions of web service messages that are based on that language type.
ServiceSpecification.xml	This file specifies the options required to generate new web service interfaces or web service implementations. You can also override certain options that you specify in the PlatformProperties.xml file.

WSDL to COBOL mapping

WSDL2COBOL batch processing mapping is based on the WSDL2ELS metadata. A COBOL2XSD mapping session file is generated for the target operation's input message, and an XSD2COBOL mapping session file is generated for the operation's output message. The input to the operation originates as a COBOL structure and becomes XML, while the output of the operation originates as XML and becomes a COBOL structure. These mapping files are in XML format and are used by the batch processor to generate XML converters and deployment metadata.

The batch processor generates COBOL structures for each operation's input and output message. Annotations are added to the generated source code to describe the relationships between the generated language structures and the XML schemas from which they are derived. The annotations appear as language comments immediately preceding the definitions of the language structures or language structure members to which they apply.

The correlator files generated by the batch processor are the same as those generated for the meet-in-middle synchronous callout support, except that they contain information for multiple operations and converters when appropriate. If you use the meet-in-middle support in Enterprise Service Tools (EST) wizard, you must map one operation at a time. Ensure that the **Update** option is selected instead of **Overwrite** so the new operation entry can be added to the correlator file.

The batch processor in the WSDL2COBOL synchronous callout scenario generates a correlator that contains information for all of the operations in the WSDL, unless it is instructed to process only a subset of the operations using the OperationSelectionArray element in the ServiceImplementationSpec in `ServiceSpecification.xml`.

WSDL2ELS does not support generation of language structures for SOAP faults, SOAP headers, or SOAP header faults in the WSDL2COBOL scenario.

For more information about the WSDL2COBOL usage details, the maximum nesting depth of XSD elements, and restrictions, see the Rational Developer for System z online help or information center.

Generating the artifacts using the batch mode

Run the `xsebatch.bat` command to generate COBOL data structures, XML converter driver, and the correlator file for your synchronous callout application.

You must first create three generation properties files to describe the web service and then run the Rational Developer for System z batch processor to generate the COBOL application with the data structure, XML converter driver(s), and the correlator file.

The best way to create these XML files is to modify the sample generation properties that are provided. The sample generation properties files are located at: *plugins_directory*`\ui_directory\BatchProcessorSamples\EISServiceImplementation` where:

- *plugins_directory* is the complete path to the named plugin subdirectory within the product installation directory, for example, `C:\Program Files\IBM\SDPShared\plugins`
- *ui_directory* is the directory `com.ibm.etools.est.ui_rrrrrr.vyyyymmdd_hhmm`
  - *rrrrrr* is a one-to-eight-character release number.
  - *vyyyymmdd_hhmm* is a time stamp that indicates the year, month, day, hour, and minute.

For example, the complete file path for the sample directories in Windows might be: `C:\Program Files\IBM\SDPShared\plugins\com.ibm.etools.est.ui_8.5.1.vxxxxxxxx_xxxx\BatchProcessorSamples\EISServiceImplementation`

The following `ServerSpecification.xml` file demonstrates how the service (wsdlServiceName), the port name (wsdlPortName), and various options for output generation are specified. For example, if you prefer not to have structure comments generated, set suppressStructureComments to true.

```
<?xml version="1.0" encoding="UTF-8"?>
<EISProject
 xmlns="http:///com/ibm/etools/xmlent/batch/emf/BatchProcessModel.ecore"
 name="PurchaseOrderCallout">
 <EISServiceImplementation runtime="IMS_SOAP_GATEWAY" type="SERVICE_REQUESTOR">
  <ServiceImplementationSpec
   importDirectory="./wsdl"
   importFile="purchaseOrder.wsdl"
   wsdlServiceName="purchaseOrderService"
   wsdlPortName="purchaseOrderSOAP">
   <DriverSpec
    fileContainer="/converters"
    fileNamePrefix="POCO@"
    driverType="IMS_SOAP" />
   <CorrelatorSpec
    fileContainer="/correlator"
    fileName="PurchaseOrder.xml"
    adapterType="IBM XML Adapater"
    socketTimeout="50000"
    executionTimeout="50000"
                calloutWSTimeout="30000"
```

```
    calloutConnBundleNames="ima2hws_callout" />
   <WSDL2ELSSpec
       fileContainer="/copybook"
       mappingDirectory="/mapping"
                metadataFileName="wsdl2els.xml"
                logFileName="wsdl2els.log"
    languageFileName="POCO.cpy"
    languageNameLimit="30"
    defaultCharMaxLength="64"
    defaultDateTimeLength="64"
    defaultTotalDigits="31"
    defaultFractionDigits="6"
    defaultMaxOccursLimit="32"
    suppressStructureComments="true"
    suppressCountFields="false"
    suppressPresenceFields="false" />
  </ServiceImplementationSpec>
 </EISServiceImplementation>
</EISProject>
```

See the Rational Developer for System z online help or information for more information about preparing the generation properties files and the batch processor.

Running the command-line batch processor:
Run the xsebatch.bat command to generate the web service artifacts. The batch processor is located in the *RDz_install_directory*/bin directory.

1. Run the xsebatch.bat command and specify where the Container.xml file that describes the web service is located.

   ```
   xsebatch.bat -f "Fully_Qualified_PathTo/Container.xml"
   -c -d "Fully_Qualified_PathTo/workspace" -verbose
   ```

   - -c *containerFile* indicates to generate the set of language converters, the drivers, and XML schemas based on the provided container file. You can override this option by using the generateConverters and the generateSeparateXSD options in the Container.xml file and in the ServiceSpecification.xml file.
   - -d *workspace* indicates the path to the workspace to be used for the import. Specify either a relative path or a fully qualified absolute path to the workspace.
   - -verbose causes the diagnostic messages to be printed to the console.
2. After the xsebatch.bat program finishes running, restart Rational Developer for System z to view the generated files in the workspace.

You can then use the generated artifacts to modify your IMS application for synchronous callout.

Generating the COBOL copybook and mapping files using the graphical user interface

If you prefer working with a graphical user interface, you can use the Enterprise Service Tools (EST) wizard to generate the mapping files from the web service WSDL file. Then use the meet-in-middle development approach to generate the SOAP Gateway correlator file and the XML converter driver.

However, this approach has the following behaviors and restrictions:

- This option does not allow customization of the COBOL copybook that is to be generated. The **Generate COBOL Mapping** menu action behaves as if default values were specified for all of the attributes on the ServiceSpecification.xml

WSDL2ELSSpec element that affect data structure generation, except for the suppressStructureComments, suppressPresenceFields, and suppressCounterFields attributes, which are all forced to true.

- This approach is supported only for the synchronous callout scenario.
- If multiple operations are involved, you must map one operation at a time. and choose the **Update** (rather than **Overwrite**) in the IMS Enterprise Suite SOAP Gateway correlator file page.

To generate the COBOL copybook and mapping files:

1. In Rational Developer for System z, open your local z/OS project and right-click your web service WSDL file.
2. In the context menu, select **Generate COBOL Mapping** > **Service Requester**. The following files are generated:

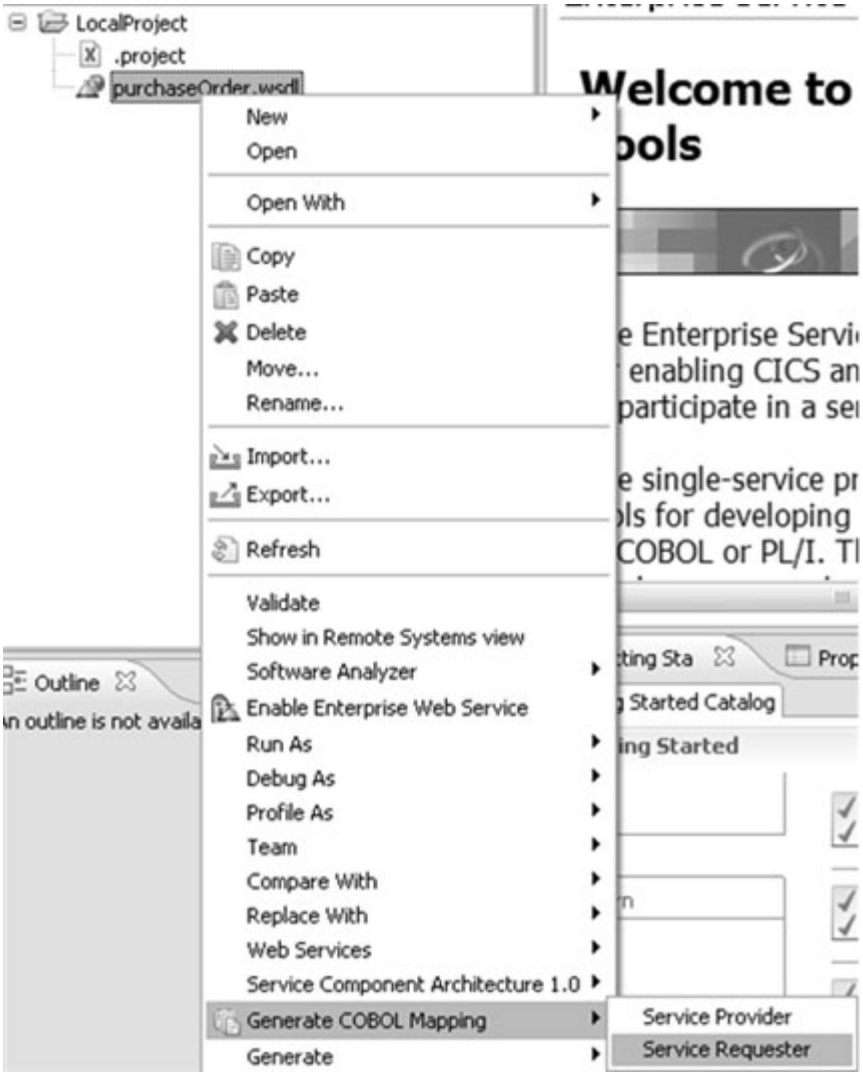


Figure 44. Generating COBOL mapping files

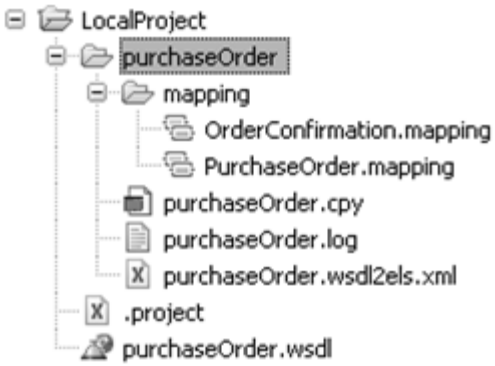


Figure 45. Generating COBOL mapping files

3. Use the generated mapping files to generate the web service artifacts by following the same step as described in "Generating web services files from the data mapping files" on page 257.

The generated COBOL copybook for synchronous callout

The WSDL2COBOL code-generation process adds annotations to the source code to describe the relationships between the generated language structures and the XML schemas from which they are derived if the suppressStructurecomments element is set to false in the `ServiceSpecification.xml` file.

The annotations display as language comments immediately preceding the definitions of the language structures or language structure members to which they apply. The generated COBOL copybook that contains the data structures can be imported into your synchronous callout application by using the COPY statement.

The name of the COBOL copybook is specified in the languageFileName attribute in the `ServerSpecification.xml` file. The following partial sample COBOL copybook demonstrated the generated COBOL data structures in the purchase order confirmation COBOL copybook (POCO) generated by Rational Developer for System z) based on the following values in the `ServerSpecification.xml` file:

If the following attributes are defined in the `ServerSpecification.xml` file:

```
languageFileName="POCO.cpy"
defaultMaxOccursLimit="32"
suppressStructureComments="true"
```

Assuming an imported XSD file has the following data structures:

```
<xsd:complexType name="PurchaseOrderType">
  <xsd:sequence>
   <xsd:element name="shipTo" type="po:USAddress" />
   <xsd:element name="billTo" type="po:USAddress" minOccurs="0" />
   <xsd:element name="items" type="po:Items" />
  </xsd:sequence>
  <xsd:attribute name="orderDate" type="po:DateType" />
 </xsd:complexType>
 <xsd:complexType name="USAddress">
  <xsd:sequence>
   <xsd:element name="name" type="po:NameType" />
   <xsd:element name="street" type="po:StreetType" />
   <xsd:element name="city" type="po:CityType" />
   <xsd:element name="state" type="po:StateType" />
   <xsd:element name="zip" type="po:ZipType" />
  </xsd:sequence>
 </xsd:complexType>
```

```
<xsd:complexType name="Items">
 <xsd:sequence>
  <xsd:element name="item" minOccurs="0" maxOccurs="1000">
   <xsd:complexType>
    <xsd:sequence>
     <xsd:element name="productName" type="po:ProductNameType" />
     <xsd:element name="quantity" type="xsd:positiveInteger" />
     <xsd:element name="USPrice" type="xsd:decimal" />
     <xsd:element name="available" type="xsd:boolean" />
     <xsd:element name="dateAvail" type="po:DateType"
      minOccurs="0" maxOccurs="1" />
     <xsd:element name="comment" type="po:CommentType"
      minOccurs="0" maxOccurs="3" />
    </xsd:sequence>
    <xsd:attribute name="partNum" type="po:SkuType" use="required" />
   </xsd:complexType>
  </xsd:element>
 </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="OrderConfirmationType">
 <xsd:sequence>
  <xsd:element name="orderID" type="xsd:unsignedInt" />
  <xsd:element name="itemCount" type="xsd:positiveInteger" />
  <xsd:element name="USTotal" type="xsd:decimal" />
  <xsd:element name="shipDate" type="po:DateType" />
  <xsd:element name="comment" type="po:CommentType"
   minOccurs="0" maxOccurs="3" />
 </xsd:sequence>
</xsd:complexType>
```

A POCO.cpy file is generated with the shipTo, billTo, and items sections for purchase order, and another section for purchase confirmation.

```
01 PurchaseOrder.
 02 item-lim PIC S9(9) COMP-5.
 02 comment-lim PIC S9(9) COMP-5.
 02 orderDate-att-bit PIC X DISPLAY.
 02 orderDate-att PIC X(64) DISPLAY.
 02 shipTo.
  03 name PIC X(128) DISPLAY.
  03 street PIC X(64) DISPLAY.
  03 city PIC X(32) DISPLAY.
  03 state PIC X(32) DISPLAY.
  03 zip PIC X(16) DISPLAY.
 02 billTo-bit PIC X DISPLAY.
 02 billTo.
  03 name1 PIC X(128) DISPLAY.
  03 street1 PIC X(64) DISPLAY.
  03 city1 PIC X(32) DISPLAY.
  03 state1 PIC X(32) DISPLAY.
  03 zip1 PIC X(16) DISPLAY.
 02 items.
  03 item
     OCCURS 0 TO 32 TIMES
     DEPENDING ON item-lim
     OF PurchaseOrder.
   04 partNum-att-bit PIC X DISPLAY.
   04 partNum-att PIC X(64) DISPLAY.
      ...

01 OrderConfirmation.
 02 comment-lim PIC S9(9) COMP-5.
 02 orderID PIC 9(9) COMP-5.
 02 itemCount PIC S9(31) COMP-3.
 02 USTotal PIC S9(25)V9(6) COMP-3.
 02 shipDate PIC X(64) DISPLAY.
```

```
 02 comment PIC X(64) DISPLAY
    OCCURS 0 TO 3 TIMES
    DEPENDING ON comment-lim
    OF OrderConfirmation.
```

The following example demonstrates how an IMS synchronous callout purchase order request program (PORQST) imports the generated COBOL copybook, POCO:

```
IDENTIFICATION DIVISION.
PROGRAM-ID. 'PORQST'.
DATA DIVISION.
WORKING-STORAGE SECTION.
* WSDL2ELS-generated language structures
COPY POCO.
```

This synchronous callout application must declare the IMS message processing program (MPP) I/O messages and the Application Interface Block (AIB) field for ICAL.

For a sample synchronous callout application, see the "Sample IMS synchronous callout application" topic.

Related concepts:

"Sample IMS synchronous callout application" on page 237
An IMS synchronous callout application must declare the IMS Application Interface Block (AIB) field for the ICAL call and the message processing program (MPP) request and response messages.

Meet-in-middle: Generating artifacts with the WSDL file and IMS callout application

Use IBM Rational Developer for System z to generate the correlator file and the XML converter that are needed to enable your IMS application to run as a web service consumer.

For Rational Developer for System z to perform data mapping and create the mapping session files between your IMS applications and the web service you are accessing, you must have the web service WSDL file and the file that contains the data structures describing the interface to your application on your workstation.

To generate web services artifact files:

1. Create a project in the z/OS Project perspective with your project source files in the z/OS project perspective.
2. Generate the request data mapping session file.
3. Generate the response data mapping session file.
4. Generate the web services files.

The artifacts generated are:

- The correlator file (.xml)
- The XML converters (.cbl)
- Optionally, the input and output data mapping XSD files (.xsd)

Related tasks:

"Deploying the XML converter to IMS Connect" on page 265
If you generate the XML converter by using Rational Developer for System z, you need to deploy it to IMS Connect.

Creating a project in the z/OS Projects perspective

To generate the data mapping files, you must first create a project in the z/OS Projects perspective to store the required source files: the COBOL copybook and the web service WSDL file.

To create a project in the z/OS Projects perspective with the source files:

1. Start Rational Developer for System z.
2. Open the z/OS Projects perspective if it is not already open.
   a. On the main menu, click **File** > **New** > **Project**. The Open Perspective window opens.
   b. In the Open Perspective window, create a local project.
      - In Rational Developer for System z , click **Workstation COBOL or PL/I** > **Local Project**. Click **Next**.
      - In older version of Rational Developer for System z:
        1) Click **z/OS Projects**. The z/OS Projects perspective opens.
        2) On the main menu, click **File** > **New** > **Project**.
        3) In the New Project wizard, click **Workstation COBOL or PL/I** > **Local Project**, and click **Next**.

      The New Local Project wizard opens.
3. In the New Local Project wizard, create your project.
   a. In the **Project name** field, type the name that you want to use for the project (for example, `MyProject`).
   b. Select the **Use default** check box if it is not already selected.
   c. Click **Do not associate the project with a property group**.
   d. Click **Finish**.

   The new project is added to the z/OS Projects view.
4. Create a folder for your source files (the COBOL copybook and the WSDL file for the web service that your IMS application is calling) in the project that you just created.
   a. In the z/OS Projects view, right-click the name of the new local project .
   b. Click **New** > **Folder**.
   c. In the New Folder wizard:
      1) Select the project that you created in step 3.
      2) In the **Folder name** field, type the name that you want to use for the new folder (for example, `source`).
      3) Click **Finish**.

      The folder is created.
5. Copy your project source files from a directory on your workstation into the proper folder in the Navigator view.
   a. Open the Navigator view:
      1) On the main menu, click **Window** > **Show View** > **Other**. The Show View wizard opens.
      2) In the Show View wizard:
         a) Expand **General**.
         b) Select **Navigator**.
         c) Click **OK**.
   b. In the Navigator view:

1) Expand your project (for example, `MyProject`).
2) Select the directory that you are working on (for example, `source`).
3) Open the directory on your workstation that contains the source files.
4) Drag the source files from the workstation directory to the new folder (source) in the Navigator view.

The source files are now added to your project.

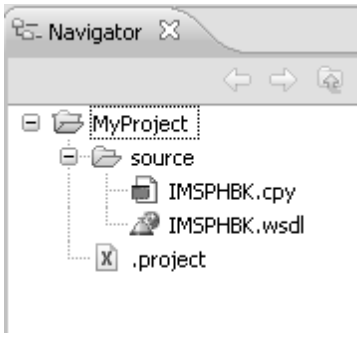


Figure 46. Source files in the local z/OS project

You have created a z/OS project and stored in it the source files that are required to create the data mapping session files.

Generating the request mapping session file

Create a request (IMS callout request) mapping session file with the IMS application source file as the source, and the web service WSDL file as the target to map the data structures between the IMS application and the web service for the request message.

To generate the request mapping session file:

1. Right-click the COBOL copybook in your project and select **Enable Enterprise Web Service.** The Enterprise Service Tools Wizard launchpad opens.
2. In the Enterprise Service Tools Wizard launchpad,
   a. Select the following settings:

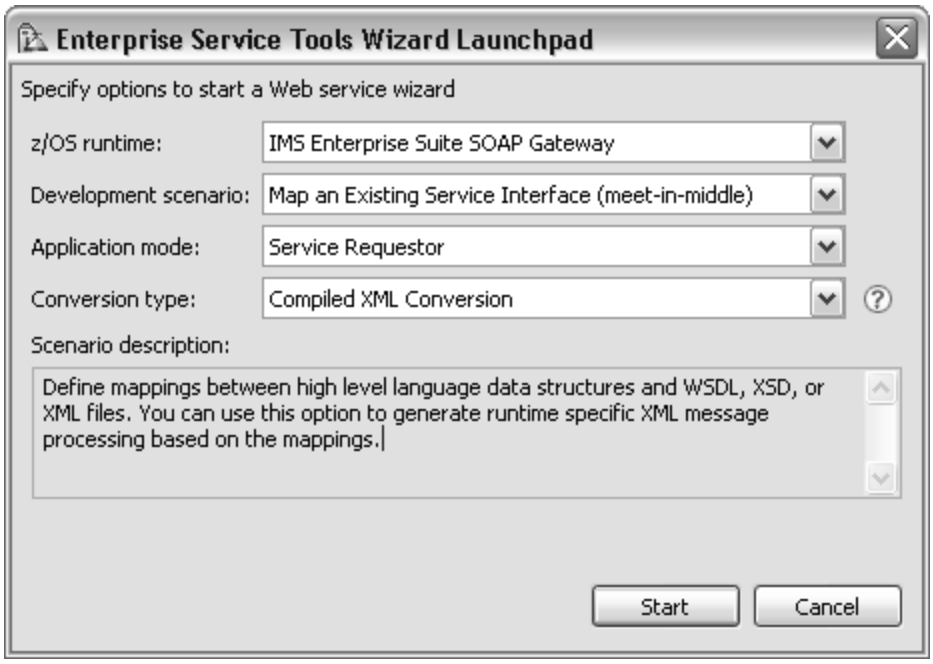


Figure 47. Selections in the Enterprise Service Tools Wizard Launchpad

- Host runtime: **IMS Enterprise Suite SOAP Gateway**
- Development scenario: **Map an Existing Service Interface (meet-in-middle)**
- Application mode: **web Service Requestor**
- Conversion type: **Compiled XML Conversion**

b. Click **Start**.

The Map an Existing Service Interface wizard opens.

3. Create a request mapping session file.
   a. On the New XML to COBOL or PL/I Mapping Session page of the wizard:

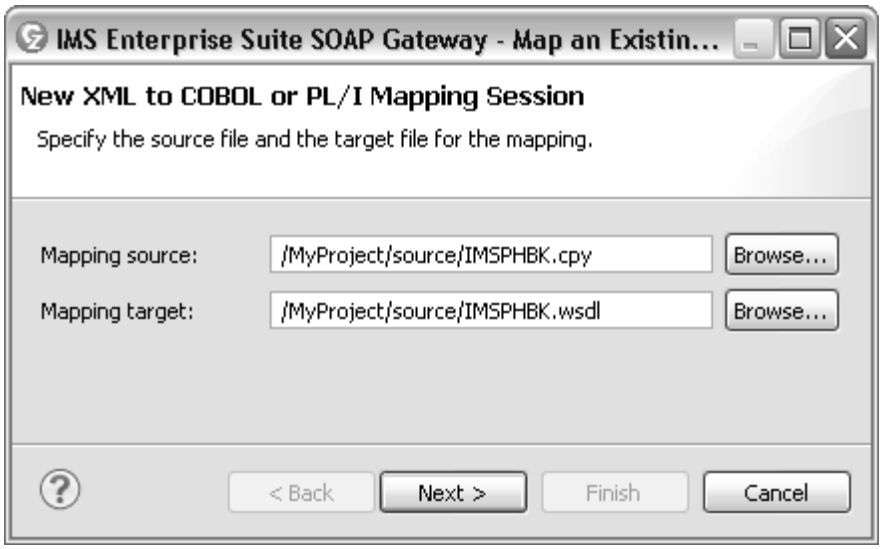


Figure 48. Specifying the mapping source and target

1) In the **Mapping source** field, the name of the copybook that you right-clicked is listed. If this is not the file you want to use for the source, click **Browse** and choose a different source file.
2) In the **Mapping target** field, specify the WSDL file for the web service that your IMS application is calling out to.
3) Click **Next**.

b. On the Root XML Element and Language Structure Selection page of the wizard, the fields are automatically completed based on the COBOL copybook source and the WSDL target file that you specified.
1) Adjust the values if they are not what you want by selecting from the selection lists.
2) Ensure that the correct source language structure is selected.
3) Click **Next**.

c. Create a new XML to COBOL mapping session.

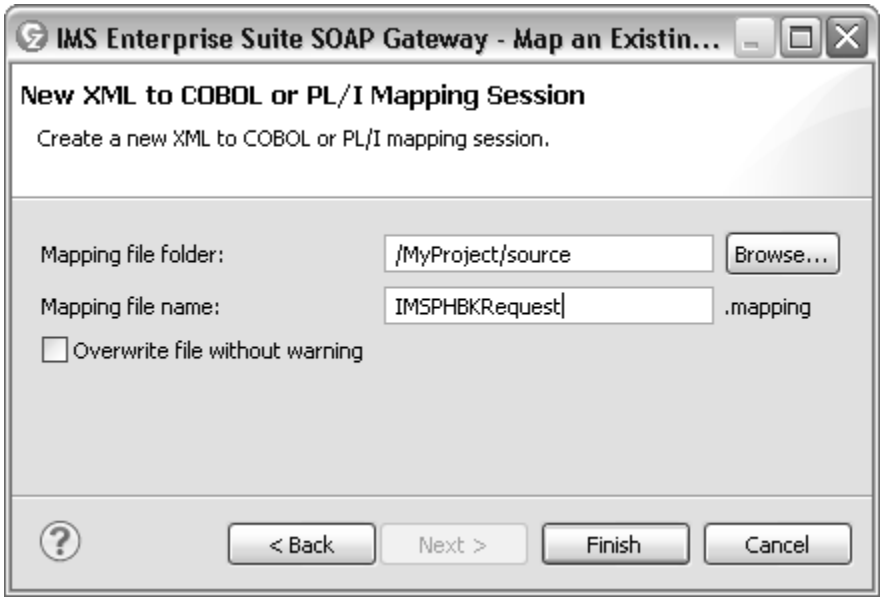


Figure 49. Creating request message mapping session file

1) In the **Mapping file folder** field, specify the path for the folder in which you want the new request mapping session file to be created.
2) In the **Map file name** field, type a name for the new request mapping session file.
3) Click **Finish**.

A mapping session file with a mapping extension is created in the file folder that you specified.

4. Use the mapping editor to create data mappings.
a. Double-click the mapping session file that you want to edit. The mapping editor opens.
b. In the mapping editor, for each mapping that you want to create, drag an element in the source data to an element in the target data. The editor displays a connecting line between the source element and the target element to indicate that a mapping exists.

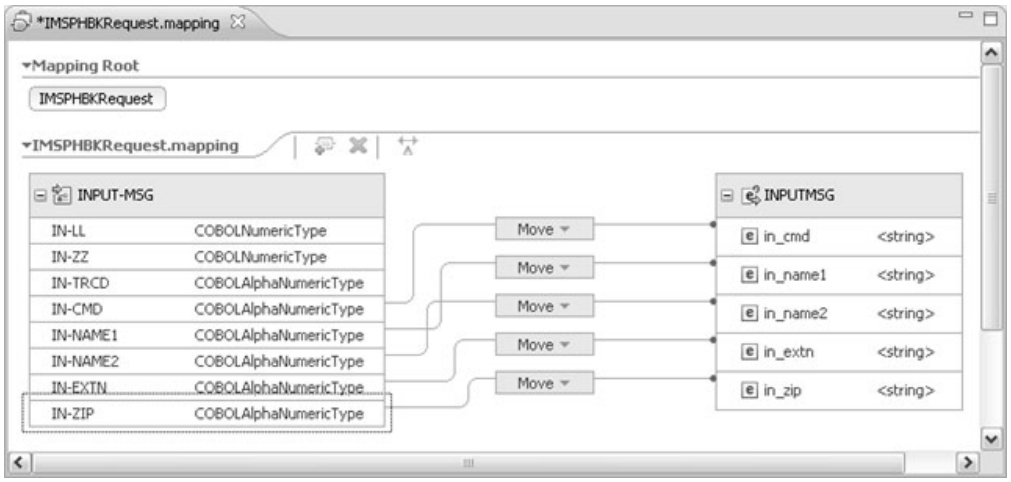


Figure 50. Request message mapping in the mapping editor

The request mapping session file is created and contains the mappings that you created.

Generating the response mapping session file

Create a response (web service response to the callout request from an IMS application) mapping session file with the web service WSDL file as the source, and the IMS application source file as the target to map the data structures between the IMS application and the web service for the response message.

To generate the response mapping session file:

1. Right-click the web service WSDL file and select **Enable Enterprise Web Service.** The Enterprise Service Tools Wizard launchpad opens.
2. In the Enterprise Service Tools Wizard launchpad, select the following options:

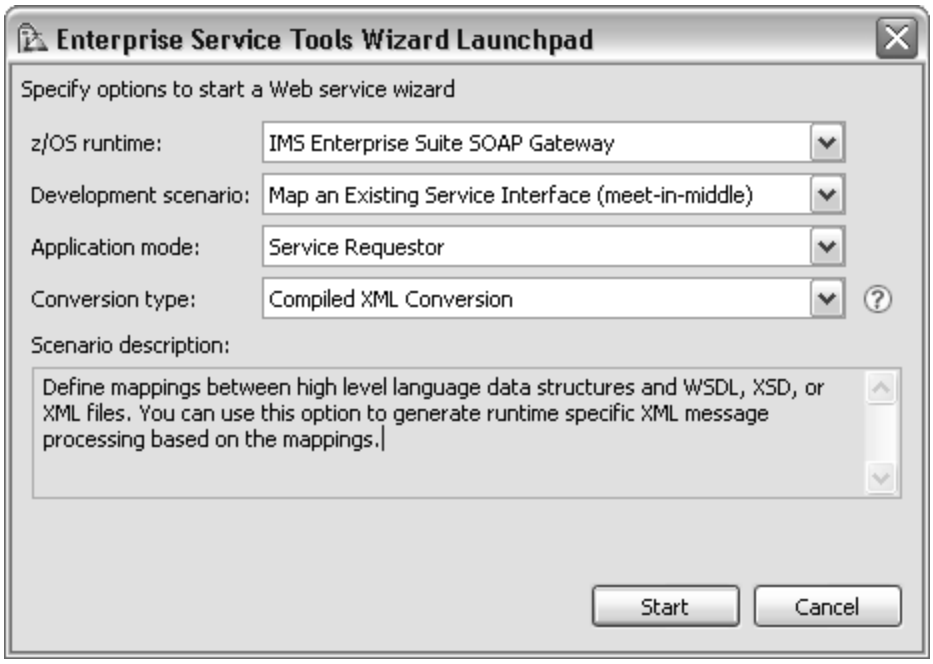


Figure 51. Selections in the Enterprise Service Tools Wizard Launchpad

- Host runtime: **IMS Enterprise Suite SOAP Gateway** (Rational Developer for System z Version 8.0.3.2 or later), or **IMS SOAP Gateway** in earlier versions.
- Development scenario: **Map an Existing Service Interface (meet-in-middle)**
- Application mode: **Web Service Requestor**
- Conversion type: **Compiled XML Conversion**

Click **Start**.

3. Create a response mapping session file.
   a. On the New XML to COBOL or PL/I Mapping Session page of the wizard:

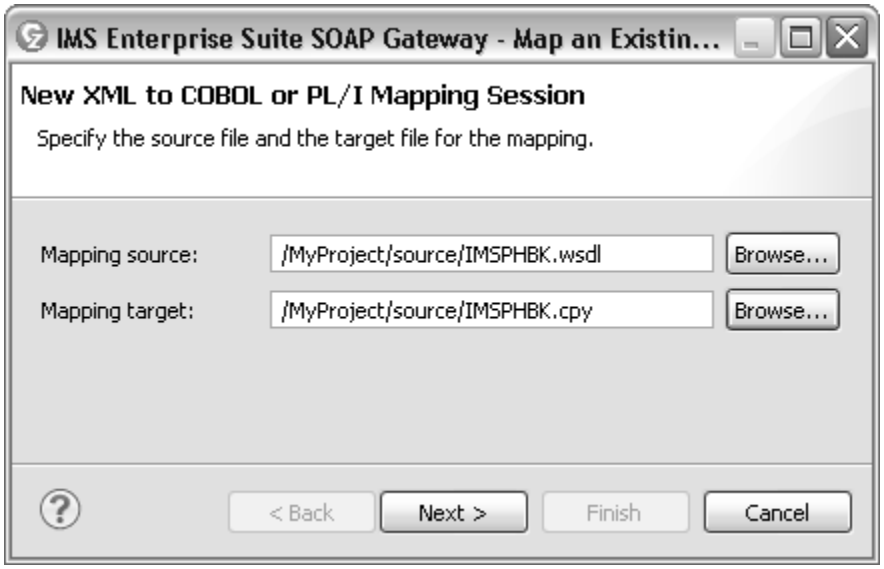


Figure 52. Specifying the mapping source and target

   1) In the **Mapping source** field, ensure that the name of the WSDL file is listed. If this is not the file you want to use for the source, click **Browse** and choose a different source file.
   2) In the **Mapping target** field, specify the source file for the IMS application that issues the callout request.
   3) Click **Next**.

   b. On the Root XML Element and Language Structure Selection page of the wizard, the fields are automatically completed based on the COBOL copybook source and the WSDL target file that you specified. Check the values.

Figure 53. Specifying the root XML element and language structure

1) Adjust the values if they are not what you want by selecting from the selection lists.
2) Ensure that the correct target language structure is selected.
3) Click **Next**.

c. Create a new XML to COBOL mapping session.

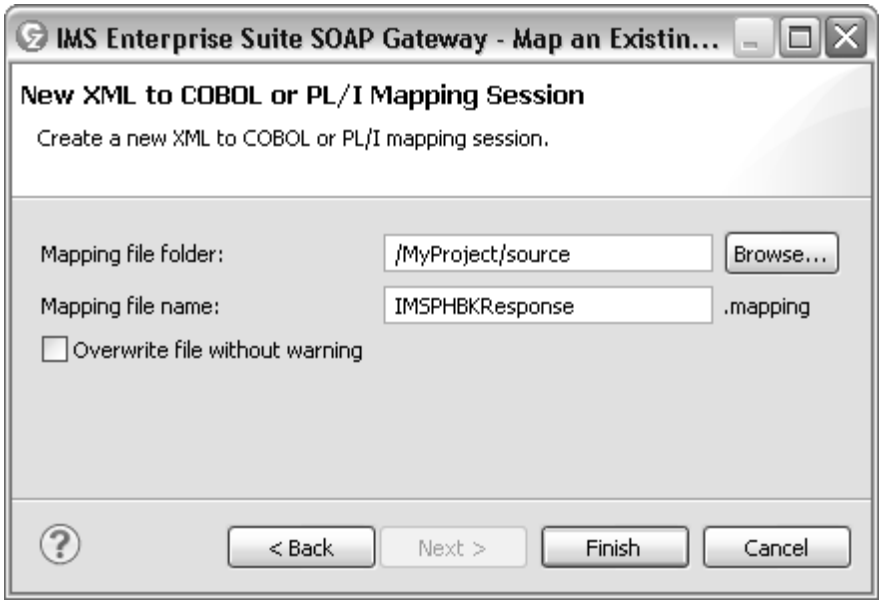


Figure 54. Creating the response message mapping session file

1) In the **Mapping file folder** field, specify the path for the folder in which you want the new response mapping session file to be created.
2) In the **Map file name** field, type a name for the new response mapping session file.
3) Click **Finish**.

A mapping session file with a mapping extension is created in the file folder that you specified.

4. Use the mapping editor to create data mappings:
   a. Double-click the mapping session file that you want to edit. The mapping editor opens.
   b. In the mapping editor:
      1) For each mapping that you want to create, drag an element in the source data to an element in the target data. The editor displays a connecting line between the source element and the target element to indicate that a mapping exists.
      2) When you are finished creating the mapping, close the mapping editor.

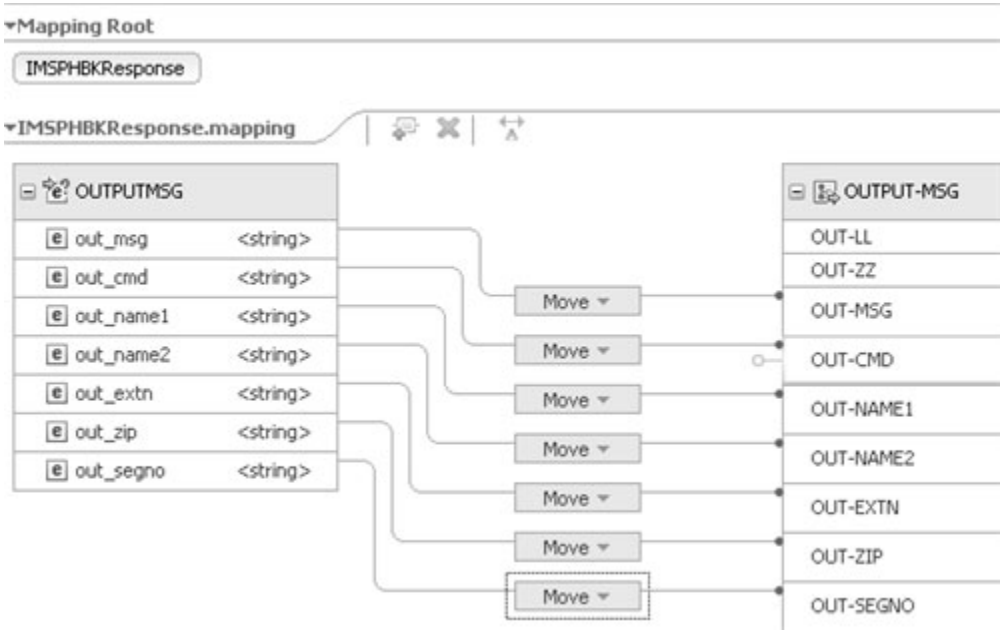


Figure 55. Creating the response message mapping

The response mapping session file is created and contains the mappings that you created.

Generating web services files from the data mapping files

To enable your IMS application to run as a web service consumer or to emit business event data, you must generate web services files from the data mapping session files.

1. In Rational Developer for System z, select both the request message and response message mapping files (hold down the Shift key for multiple selection).
2. Right-click anywhere in the view that contains the mapping files and select **Generate Conversion Code**.

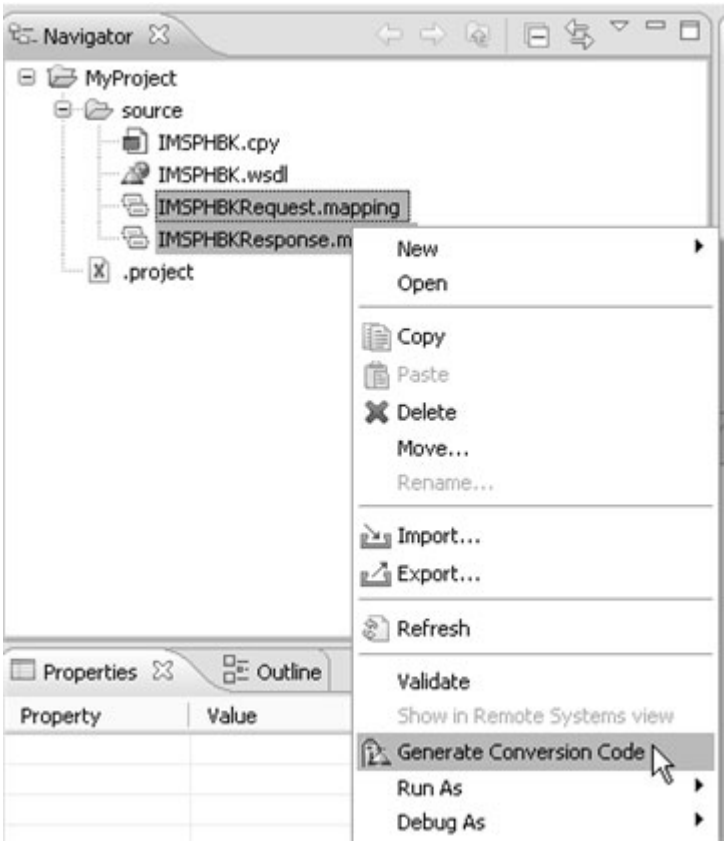


Figure 56. Generating the conversion code from the data mapping files

3. In the Enterprise Service Tools launchpad, select the following values:
   - Host runtime: **IMS Enterprise Suite SOAP Gateway**
   - Application mode: **Web Service Requestor**
4. Click **Start**. The Map to an Existing Service Interface (meet-in-middle) wizard opens.
5. In the Generation of conversion code page, verify that you have selected the correct mapping session files, and click **Next**.

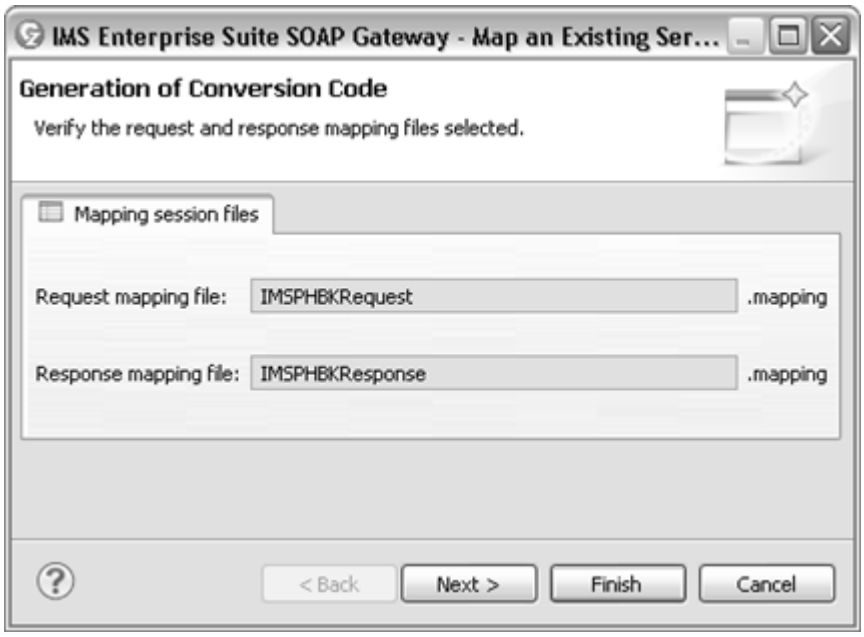


Figure 57. Selecting the request and response mapping files

6. In the Generation options page:

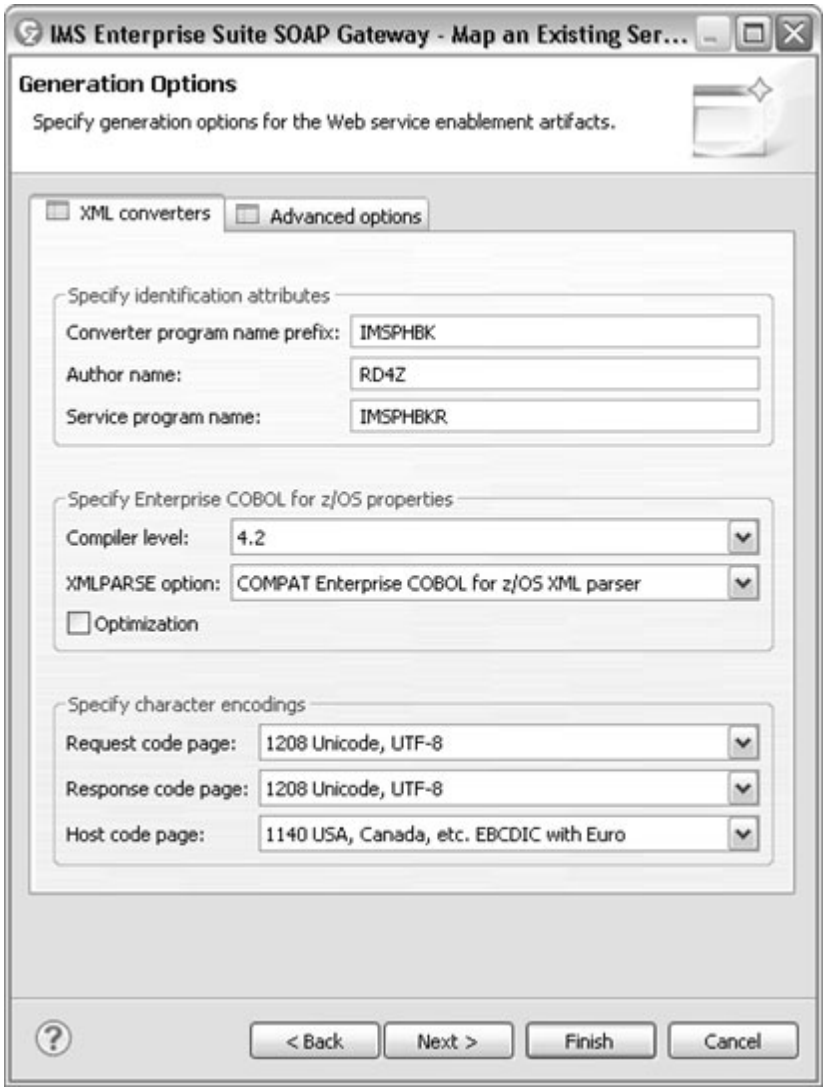


Figure 58. Specifying conversion options

a. In the **XML Converters** tab, select or specify the following settings:
   1) Host code page: Select the code page that the host uses.
   2) Inbound code page and outbound code page: Default is **1208 Unicode, UTF-8**. SOAP Gateway supports only UTF-8 and therefore you cannot change this value.

b. Click **Next**.

7. Specify properties on the SOAP Gateway Web Service Requestor page:
   a. Specify any service identification properties that need to be specified for your SOAP Gateway environment. This information is used
      1) In the File container field, specify the folder and subfolder in which you want the correlator file to be generated.
      2) Specify the IMS Connect interaction properties:

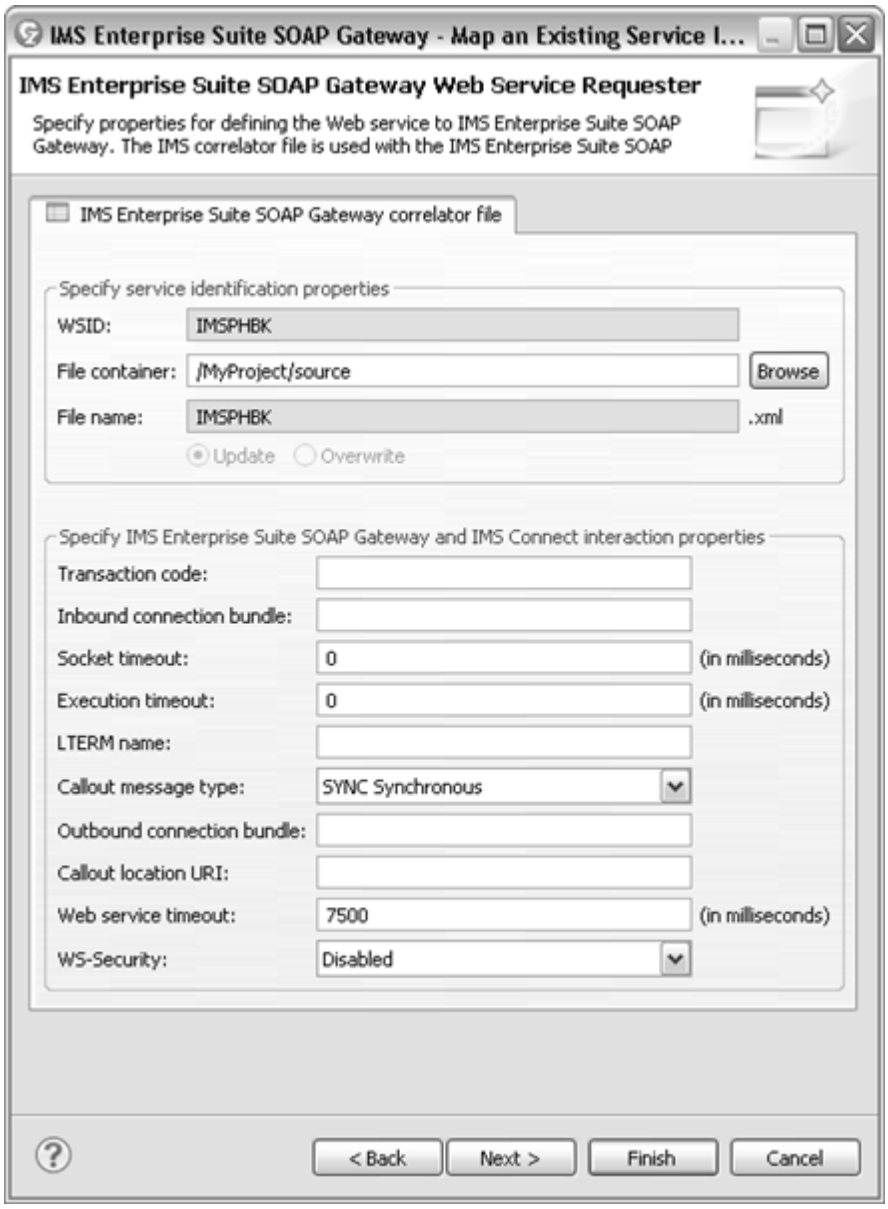


Figure 59. Specifying correlation properties

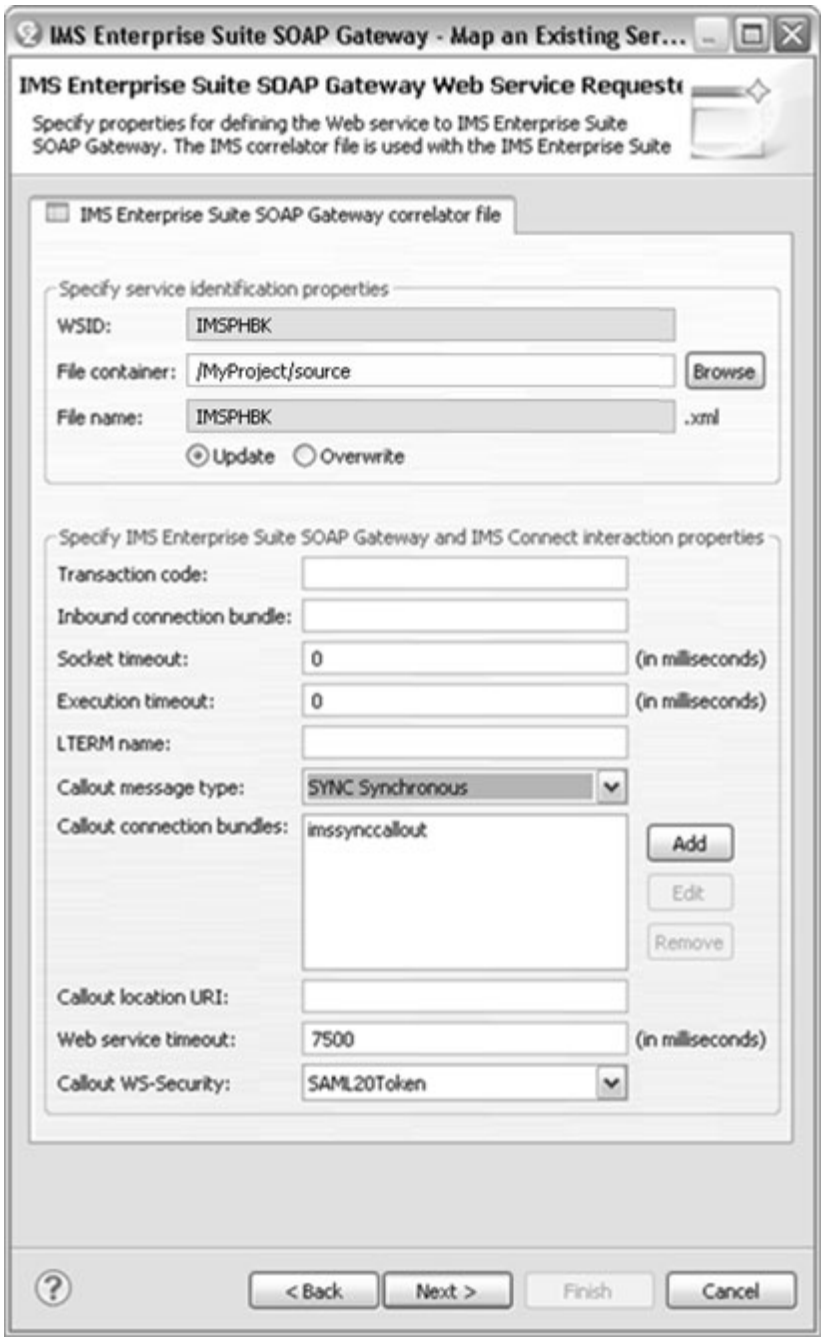


Figure 60. Specifying correlation properties

- For multiple operations, you must map one operation at a time. Ensure that **Update** is selected instead of **Overwrite** so the new operation entry can be added to the correlator file.
- Transaction code: Specify this value only for a request-response asynchronous web service invocation. That is, a response from the web service for an asynchronous callout request is expected.
- Inbound connection bundle: Specify this property if a response for the web service is expected.
- Socket timeout value in milliseconds.
- Execution timeout value in milliseconds.

- LTERM name.
- Callout message type.
- Callout connection bundles: Specify one or more callout connection bundle names. Use the **Add** button to add additional bundles.
- Web service timeout value in milliseconds.
- Callout WS-Security. Select the security type.

Tips:

- The transaction code value is needed only for responding to an asynchronous callout request. For synchronous callout requests, SOAP Gateway ignores the transaction code even if it is specified. SOAP Gateway handles the correlation of the response with the request by using the correlation token that is attached to the synchronous callout request. For one-way asynchronous callout requests that do not expect a response, the transaction code is also ignored.
- For the synchronous callout message type, SOAP Gateway does not receive or manage the LLZZ prefix. The data portion of the message contains only data. If you map the LLZZ prefix during data mapping, you must treat LLZZ as data in the callout web service.
- You can use the SOAP Gateway management utility to modify these correlator properties later if necessary.

b. Click **Next**.

8. On the File, data set, or member selection page of the wizard:
   a. In the **XML Converters** tab:
      1) For the converter file container, specify the folder and subfolder in which you want the converter file to be created.
      2) For the converter driver file name, specify the name in which you want the converter programs to be generated.
      3) Ensure that the check box **Generate all to driver** is selected. This selection causes all the generated web service programs (driver, inbound converter, and outbound converter) to be placed in the same file.

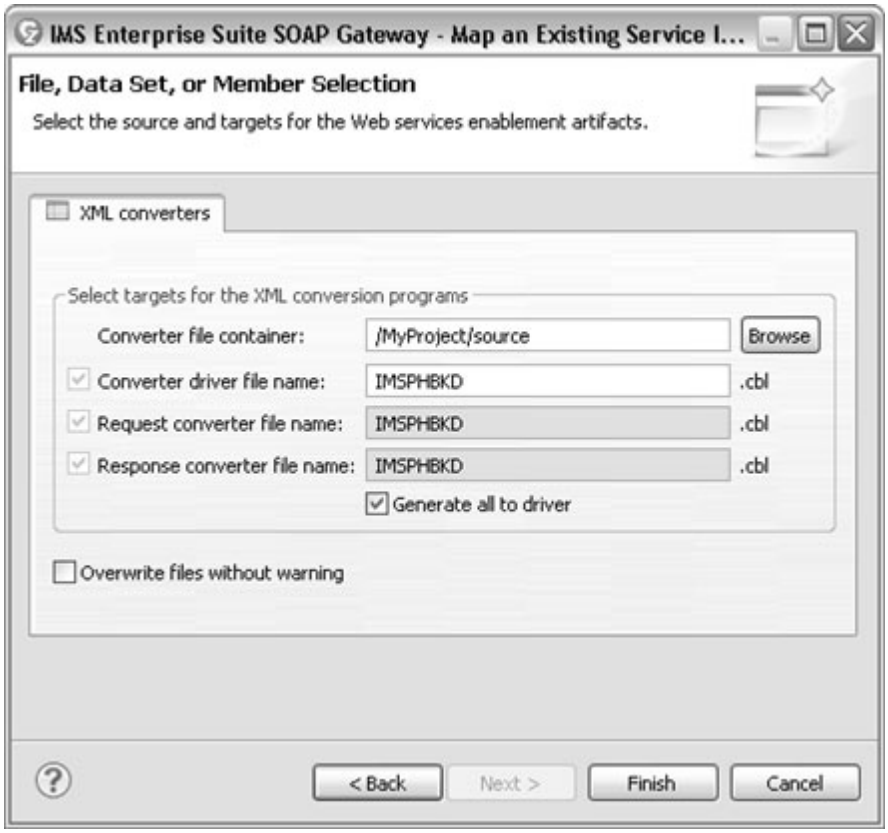


Figure 61. Specifying converter file name and container

b. Click **Finish**.

The following files are generated in the directories and file names you specified:

- The correlator file (.xml)
- The file that contains the web service driver and runtime XML converter (.cbl)

Important: The correlator file that is generated by Rational Developer for System z Version 9.0 or older versions is in a correlator schema older than version 3.0. The correlator file must be migrated to the new correlator scheme:

1. Store the correlator file in the *install_dir*/imssoap/xml directory.
2. Use the SOAP Gateway management utility iogmgmt -migrate correlator command to migrate the correlator file.

Important: Before you run the migration tool, ensure that the calloutConnBundleName property in the version 1.0 correlator schema is not empty. If the calloutConnBundleName property is empty, the migration tool would assume that this correlator is for the web service provider scenario and set the correlator mode value to call-in instead of call-out.

Related tasks:

"Migrating correlator files to schema version 3.0" on page 302
IMS Enterprise Suite Version 3.1 SOAP Gateway requires correlator schema version 3.0. To migrate an existing correlator file from older versions to version 3.0, use the SOAP Gateway management utility iogmgmt -migrate correlator command.

Related reference:

"-migrate: Migrate and upgrade SOAP Gateway" on page 448
The -migrate command upgrades SOAP Gateway artifacts and settings to the latest version and generates a migration log.

Deploying the XML converter to IMS Connect

If you generate the XML converter by using Rational Developer for System z, you need to deploy it to IMS Connect.

Prerequisites:

1. The XML adapter function must be configured in IMS Connect. For more information, see IMS Version 13 System Definition information.
2. The Rational Developer for System z host configuration must be completed. For more information, see the *Rational Developer for System z Host Configuration Guide*.
3. The Rational Developer for System z SFEKLOAD (V8.0.*x*) or SFEKLMOD (V8.5 or later) module must be concatenated to the STEPLIB of the IMS Connect startup JCL in order to generate IRZ error messages to assist troubleshooting.

To deploy the XML converter to IMS Connect:

1. Use FTP to copy the XML converter that is generated by Rational Developer for System z to your host data set.
2. Compile and bind the XML converter to the target host data set.

   To compile and bind the converters into a data set concatenated with the IMS Connect STEPLIB:

   - The converter must have an alias that is linked with the converter code, using the same name as the converter, with an X suffix. If the converter name is eight-character long, the last character for the alias must be changed to an X.
   - The converter and the load module name must end with the letter D.
   - The converter should be linked with the option REUS=SERIAL so that the converter is loaded into the memory only once.

   For example, if your converter file name is IMSSGWS1D. Your JCL would contain ENTRY, ALIAS, and NAME statements as follows:

```
//LINK    EXEC PGM=HEWL,COND=(4,LT),
//        PARM='XREF,COMPAT=MIN,REUS=SERIAL'
...
  ENTRY IMSSGWS1D
  ALIAS IMSSGWS1X
  NAME IMSSGWS1D(R)
```

Related tasks:

"Configuring IMS Connect for SOAP Gateway" on page 101
You must configure IMS Connect to allow SOAP Gateway to access IMS transactions.

Related reference:

DFSYDTx PROCLIB member data set for OTMA descriptors (IMS Version 13)
For more information, see the DFSYDTx PROCLIB member data set information in IMS Version 13 System Definition information.

Related information:

Rational Developer for System z Host Configuration Guide
For host configuration instructions, see the *Host Configuration Guide* for the version of Rational Developer for System z that you on the Rational Developer for System

z library page.

Creating a connection bundle entry for callout applications

Create a connection bundle entry that describes the connection properties for accessing IMS by using the SOAP Gateway management utility. The connection bundle entries are stored in the `connbundle.xml` file.

You must create a new connection bundle entry or modify an existing connection bundle entry to specify the following properties:

- The connection properties for IMS Connect
- The names of the tpipes that hold the synchronous and asynchronous callout requests that are sent by your IMS application

If a response message is expected from the web service, you must either create an additional connection bundle entry or reuse an existing connection bundle entry to specify connection properties for sending the output response message. The connection bundle entry with the connection properties for the response message is specified separately from the connection bundle entry with the connection properties for the callout message. Each is specified with a separate entry in the callout correlator XML file. However, both sets of properties can be stored in the same connection bundle entry.

1. Gather the required information for the connection bundle entry. A connection bundle entry for a callout application must contain the following information:
   - Name for the connection bundle entry
   - Name of the callout tpipe for the application
   - IMS Connect host name
   - IMS Connect listening port number (default is 9999)
   - IMS Connect datastore name
2. Optional: To enable callout basic authentication, gather the following information:
   - Callout basic authentication user ID
   - Callout basic authentication password

   This option configures SOAP Gateway to perform basic authentication with the target web service as part of a callout request.
3. Optional: To enable server SSL authentication for the callout request, gather the following information:
   - Callout SSL truststore name
   - Callout SSL truststore password

   This option configures SOAP Gateway to confirm the identity of the server by verifying the information in the server truststore.
4. Optional: To enable client SSL authentication for the callout request, gather the following information:
   - Callout SSL keystore name
   - Callout SSL keystore password

   This option configures SOAP Gateway to send the server a client certificate that the server can use to confirm the identity of the SOAP Gateway server. Server SSL authentication must also be configured in the connection bundle entry to use client SSL authentication.
5. Issue the `iogmgmt -conn -c` command to create the connection bundle entry.

Example 1. No security. The following example creates a connection bundle entry named `connbundle1` that connects to an IMS Connect host named ICONHOST1 on port 9998 and that uses a callout tpipe named tpipe1. Security is not enabled.

```
iogmgmt -conn -c -n connbundle1 -h ICONHOST1 -p 9998 -d IMSSTOR1 -i tpipe1
```

Example 2. Basic authentication. The following example creates a connection bundle entry named `connbundle2` that connects to the IMSSTOR2 data store on an IMS Connect host named ICONHOST2 on port 9995. The callout tpipe is tpipe2. Basic authentication user ID and password is specified for basic authentication.

```
iogmgmt -conn -c -n connbundle2 -h ICONHOST2 -p 9995 -d IMSSTOR2 -i tpipe2
 -m basicAuthID  -b basicAuthPwd
```

Example 3. Client authentication and basic authentication. The following example creates a connection bundle entry named `connbundle3` that connects to the IMSSTOR3 data store on an IMS Connect host named ICONHOST3 on port 9992. The callout tpipe is tpipe3. In addition to basic authentication, target keystore and truststore information is provided for client authentication.

```
iogmgmt -conn -c -n connbundle3 -h ICONHOST3 -p 9992 -d IMSSTOR3 -i tpipe3
-l callout_target_ks_name -y callout_target_ks_pwd
-v callout_target_ts_name -q callout_target_ts_pwd
-m basicAuthID -b basicAuthPwd
```

A message indicates that the connection bundle entry is created.

Related concepts:

"Connection bundle properties" on page 20
The connection bundle specifies the connection and security properties for SOAP Gateway when it communicates with IMS Connect.

Related tasks:

"Configuring SSL and HTTPS support with Java keystore (JKS)" on page 148
To use HTTPS between a SOAP Gateway client and SOAP Gateway, or SSL between SOAP Gateway and its server (IMS Connect), you must create the keystore and truststore, and configure the SOAP Gateway server.

Related reference:

"-conn: Create, update, or delete a connection bundle" on page 435
Use the -conn command to create, update, or delete a connection bundle.

Deploying a callout application to SOAP Gateway

Deploy a callout application or business event emitter to SOAP Gateway with the SOAP Gateway management utility.

A callout application requires a valid correlator XML file with interaction properties for the application, a valid WSDL file that describes the web service to clients (or XSD file for calls to a business event monitoring server), and a valid connection bundle with connection properties for the application.

1. Ensure that the required web service artifacts are located in the SOAP Gateway installation directory. The correlator XML file and WSDL (or XSD) file can either be in the XML and WSDL directories, or elsewhere in the SOAP Gateway directory. If the files are already located in the XML and WSDL directories when you deploy the web service, you can provide the file name without the fully qualified path to the file.

   Tip: z/OS For z/OS systems, the correlator file and the WSDL file, when uploaded from a distributed platform, must be transferred in BINARY mode

from your local workstation. Binary transfers provide a bit-by-bit copy that preserves the encoding on your system by instructing the FTP socket not to convert the encoding to the local system encoding (EBCDIC).

Restriction: Nested XSD import statements are supported only by Rational Developer for System z Version 8.5.1 or later in its top-down support for COBOL data structure generation for synchronous callout. Sharing of XSD schema files among callout applications is not supported. See the -deploy command reference for details.

2. If you are not using the XML adapter function in IMS Connect, update your correlator file by using the SOAP Gateway management utility iogmgmt -corr command and setting the `-a` option (adapter type) to `No_Adapter`. By specifying the `No_Adapter` value, the adapterType entry in the correlator is set to blank. By default, this entry is set to `IBM XML Adapter`.
3. Issue the command to deploy the application to the server:
   - For a one-way or request-response callout application, or a business event application that emits events to WebSphere Business Events, issue the command `iogmgmt -deploy -w` *`wsdl_file`* `-r` *`correlator_file`*`.`
   - For a business event application that emits events to WebSphere Business Monitor, issue the command `iogmgmt -deploy -w` *`xsd_file`* `-r` *`correlator_file`*`.`

SOAP Gateway is configured to pass callout requests or business events to the target web service or business event monitor.

Related reference:

"-deploy: Deploy a web service or callout application" on page 444
The -deploy command deploys a web service, callout application, or business event application to the active configuration of the SOAP Gateway server.

Starting the callout thread for a specific application

You must start a callout thread after a callout application is deployed to SOAP Gateway before the application can begin processing callout messages.

This task operates at the thread level. The behavior of this task depends on the thread policy that has been set. If the policy is to have one thread per tpipe, the start action affects only that particular tpipe. If the thread policy is set to one thread per connection bundle, the start action affects the callout thread for every tpipe defined in the connection bundle.

1. Ensure that the SOAP Gateway server has an active HTTP listening port. The SOAP Gateway management utility commands for thread management require an active HTTP listening port.
2. Ensure that the callout thread pool is started with the `iogmgmt -view -workerthreads` command. Callout threads poll messages from defined tpipes and send them to be picked up by worker threads in the thread pool. The worker threads then invoke external web services. If the thread pool is stopped while callout threads are running, the work queue fills up with unprocessed messages and callout processing halts.
3. Determine which callout thread must be started.
   - If your current thread policy is one thread per tpipe, you must know both the name of the connection bundle specified in the correlator XML file for the callout application and the name of each tpipe in the connection bundle that ais associated with the application.

- If your current thread policy is one thread per connection bundle, you only must know the name of the connection bundle.

4. Start the callout thread to begin processing callout requests for the application with the `iogmgmt -callout -startone` command.
   - If your thread policy is one thread per tpipe, the command is `iogmgmt -callout -startone -c` *`connection_bundle_name`* `-p` *`tpipe_name`*. You must issue the command for each tpipe associated with the application to start all of the relevant callout threads.
   - If your thread policy is one thread per connection bundle, the command is `iogmgmt -callout -startone -c` *`connection_bundle_name`*.
5. Optional: Verify that the callout thread for the application is running with the `iogmgmt -view -calloutthreads` command. A callout thread might take some time to start, depending on available system resources.

The callout thread for the connection bundle (and tpipe name, if applicable) that you specified begins to process callout messages.

Related concepts:

"Thread management for callout messages retrieval" on page 174
SOAP Gateway supports two options to determine how to manage the callout threads to send the requests to poll the hold queue for callout request messages: one thread per tpipe, or one thread per connection bundle.

Related reference:

"-callout -startone: Start a specific callout thread" on page 431
The -callout -startone command starts a specific callout thread based on the provided connection bundle name and tpipe name.

"-callout -stopone: Stop a specific callout thread" on page 432
The -callout –stopone command stops a specific callout thread based on the provided connection bundle name and tpipe name.

Chapter 7. Enabling an IMS application to emit a business event

To enable an application to emit a business event, you must modify your IMS application, define an OTMA destination descriptor, generate the correlator file, the XML converter, and the data mapping XSD file, and configure SOAP Gateway for the business event server.

Ensure that you understand the following concepts and considerations before you proceed:

- "Business events processing flow" on page 198
- "Design guidelines for emitting business events" on page 200
- "Correlating event messages to event processing services" on page 272
- "Security for business event requests" on page 201
- "Security for the consumer (callout) scenario" on page 182

The steps to enable an IMS application to emit a business event to be processed or monitored by a business event processing engine are similar to the steps in the one-way asynchronous callout scenario, with a few minor differences that are specific to WebSphere Business Events and WebSphere Business Monitor.

1. Use the ISRT ALPCB call to place the event data on the OTMA hold queue.
2. Use an OTMA destination descriptor for an IMS application to route business event data to a server that is accessible to SOAP Gateway without the need to code assembler routing exits.
3. Generate the correlator file, the XML converter, and the data mapping XSD file by using Rational Developer for System z from the IMS application source file.
4. Deploy the XML converter in IMS Connect.
5. Create a connection bundle by using the SOAP Gateway management utility with the required information for SOAP Gateway to connect to IMS Connect.
6. Configure the SOAP Gateway server to route the business events to the specified business event processing server. Configuration is done by using the SOAP Gateway management utility and by providing the WSDL or XSD file for the business event server.
7. Start the callout thread. The SOAP Gateway callout thread is used to pull the business event data from the OTMA hold queue.

Related concepts:

"Business event scenario" on page 197
IMS applications can emit business events to IBM business event engines, such as IBM WebSphere Business Events and IBM WebSphere Business Monitor, for business activities processing and monitoring through SOAP Gateway.

Related tasks:

"Defining an OTMA destination descriptor for callout request messages" on page 238
You can define an OTMA destination descriptor to route IMS callout requests to a hold queue, without the need to code assembler routing exits.

"Creating a connection bundle entry for callout applications" on page 266
Create a connection bundle entry that describes the connection properties for accessing IMS by using the SOAP Gateway management utility. The connection

bundle entries are stored in the `connbundle.xml` file.

"Deploying the XML converter to IMS Connect" on page 265
If you generate the XML converter by using Rational Developer for System z, you need to deploy it to IMS Connect.

"Setting up WebSphere Business Monitor for IMS business events" on page 287
Before you configure SOAP Gateway to emit business event data, set up WebSphere Business Monitor to be ready to receive business events.

"Creating a correlator file for a callout application" on page 239
You can manually create a correlator file with the SOAP Gateway management utility if you do not have IBM Rational Developer for System z.

Correlating event messages to event processing services

SOAP Gateway provides support for correlating an event request message to an external business event processing service.

Because the business event support in SOAP Gateway works similarly to the one-way asynchronous callout function, correlation of event messages to the external event processing services works in similar ways as the correlation for callout messages. A business event message also includes service data prefix and the payload data.

The following table describes the elements in the service data prefix.

Table 32. Elements in the service data prefix

Elements in the prefix	Description
web service identifier (WSID)	The WSID value is used to identify the outbound event processing service, and helps SOAP Gateway load the appropriate correlator file that is associated with the event processing service. The name of the WSID is used for the name of the correlator file. For example, if the WSID value is ES1, the associated correlator file name is `ES1.xml`. SOAP Gateway uses the CalloutURI property in the correlator file to communicate and emit the event data to WebSphere Business Monitor. For WebSphere Business Events, SOAP Gateway obtains the correlator file name from the WSID value, loads the correlator file and then loads the WSDL file. The URL address for WebSphere Business Events is then taken from the WSDL file.
Namespace	The target namespace of the event processing service description language WSDL file. This information is used only when the event message is emitted using the SOAP protocol. This element is used by WebSphere Business Events, not WebSphere Business Monitor. For WebSphere Business Monitor, this element contains null value.
Service name	The service name of the port of operation to be invoked. This element is used by WebSphere Business Events, not WebSphere Business Monitor. For WebSphere Business Monitor, this element contains null value.

Table 32. Elements in the service data prefix (continued)

Elements in the prefix	Description
Port name	The port name of the operation to be invoked. This element is used by WebSphere Business Events, not WebSphere Business Monitor. For WebSphere Business Monitor, this element contains null value.
Operation name	The operation name of the event service to be invoked. This element is used by WebSphere Business Events, not WebSphere Business Monitor. For WebSphere Business Monitor, this element contains null value.

SOAP Gateway uses the information from the correlator file to correlate to the appropriate event processing service. The correlator file can be generated by using IBM Rational Developer for System z or the SOAP Gateway management utility.

Generating and deploying artifacts for emitting business events to WebSphere Business Events

Generate the web service artifacts in Rational Developer for System z and WebSphere Business Events, and deploy the generated files by using the SOAP Gateway management utility.

These tasks describe how to enable an IMS application to emit business event data to WebSphere Business Events.

Related information:

WebSphere Business Events information center
WebSphere Business Events information center

Coding business event data and inserting the event emission point

You must first identify the event emission point in your IMS application and provide the event data in an existing or a new data structure.

The event data is placed on an OTMA hold queue with an identified OTMA destination descriptor, in the same way as asynchronous callout messages.

1. Create a new data structure or modify an existing one for business event data.
2. Use the ISRT ALTPCB DL/I call at the event emission point to place the business data on the hold queue that is defined in an OTMA destination descriptor.

Related concepts:

"Design guidelines for emitting business events" on page 200
You might want to add new fields to your data structure or generate the ALTPCB value in the PSB, depending on how the events are being processed and whether the IMS application has access to an ALTPCB.

Defining an OTMA destination descriptor for business events

Define an OTMA destination descriptor to route IMS business event data to a hold queue.

To use the OTMA destination descriptor:

1. Configure the descriptor in the DFSYDTx PROCLIB member.
2. Specify the ADAPTER and the CONVERTR values. Set the ADAPTER value to HWSXMLA0 if you want XML data transformation to be completed by IMS Connect. The following example OTMA destination descriptor spans multiple lines:

```
D SOAPGWAY TYPE=IMSCON TMEMBER=HWS2 TPIPE=HWS2SOAP
D SOAPGWAY ADAPTER=HWSXMLA0 CONVERTR=XMLCNVTR
```

   - The descriptor for SOAPGWAY routes messages to IMS Connect target member HWS2 with tpipe HWS2SOAP.
   - The ADAPTER is set to HWSXMLA0 for the data transformation to be performed by the IMS Connect XML adapter.
   - The XML converter name is XMLCNVTR.

   Important: Do not share the tpipe that you use for SOAP Gateway callout functions with business event data or callout functions in the IMS TM resource adapter.

Generating the XML schema file (XSD file)

An XSD file describes the XML scheme and is required for the exchange of XML messages between IMS and the business event server. Generate the XSD file by using IBM Rational Developer for System z.

1. Start Rational Developer for System z.
2. Open the Enterprise Service Tools Projects perspective if it is not already open.
   a. On the main menu, click **Window** > **Open Perspective** > **Other**. The Open Perspective window opens.
   b. Select **Enterprise Service Tools** and click **OK**.
3. Create a new Batch, TSO, z/OS UNIX Project.
   a. Click **File** > **New** > **Batch, TSO, z/OS UNIX Project**.
   b. Select the following options:
      - Project name: Specify a project name.
      - Development scenario: **Create New Service Interface (bottom-up)**
      - Application mode: **Service Provider**

      Click **Next**.
4. Import the IMS COBOL application source file that contains the event data structure and click **Finish**. The Language structures page appeared.
5. Generate an XML schema for the event data structure in the IMS application source file.
   a. On the Language structures page:
      - In the **Request Language Structure** tab, select the event data structure.
      - Leave the response language structure as is. Because no response is expected from the business event server, none of the check boxes in the response language structure should be selected.
      - Ensure that the COBOL preference is set to z/OS.
      - Click **Next**.
   b. On the Generation options page:
      1) Click the **WSDL and XSD** tab.
      2) In the **Specify request XML Schema properties** area, update the **Target Namespace** and the **Root Element name** fields if necessary.

3) Click the **Advanced options** tab.
4) Ensure that the **Generate qualified elements in XML Schemas** check box is selected (default).
5) Click **Next**.

c. On the File, data set, or member selection page:
1) Click the **XML Converters** tab.
2) Deselect the **Generate all to driver** check box, and then clear the **Converter driver file name** check box, and the **Request Converter file name** check box.
3) Click the **WSDL and XSD** tab.
4) Clear the WSDL file name check box.
5) Ensure that the **Request XSD file name** check box is selected.
6) Update the XSD file name, if necessary.
7) Click **Finish**.

The XML Schema file (XSD) for the event data structure is generated in the `Targets` directory.

Generating the WSDL file for WebSphere Business Events

Generate the WSDL file in WebSphere Business Events Design Data by importing the XML schema (XSD) file that is generated by Rational Developer for System z. The WSDL definition is used to submit an event as a web service to the WebSphere Business Events Runtime.

Prerequisite: You must be familiar with WebSphere Business Events Design Data and WebSphere Business Events Design.

Use WebSphere Business Events Design Data to define a touchpoint to be the container that holds all the events and actions. Business users can use WebSphere Business Events Design to create an event flow.

Related information:

Creating a WSDL file that describes an event in WebSphere Business Events
See the WebSphere Business Events information center for more information about how to create a WSDL file that describes an event.

Defining events and actions in WebSphere Business Events
See the WebSphere Business Events information center for more information about how to define events and actions.

Generating the data mapping file for business events

IBM Rational Developer for System z to generate a request (IMS business event) mapping session file to map the data structures between the IMS application and the business event server for the business event data.

Prerequisites:

- The IMS application source file.
- For WebSphere Business Events, the web service WSDL file that is generated by WebSphere Business Events.
- For WebSphere Business Monitor, the XSD XML scheme file that is generated earlier by Rational Developer for System z.

1. Start Rational Developer for System z.

2. In the Enterprise Service Tools perspective, create an IMS SOAP Gateway project by clicking **File** > **New** > **IMS Enterprise Suite SOAP Gateway project**.
3. In the New IMS Enterprise Suite SOAP Gateway Project window:
   a. Specify a project name.
   b. For **Development scenario**, select **Map an Existing Service Interface (meet-in-middle)**.
   c. For **Application mode**, select **Service Requestor**.
   d. Click **Next**.
4. Import the IMS application source code (copybook) and either an XSD or WSDL file, depending on the target business event processing server. You can find the application files in the Targets directory in your project.
   - For WebSphere Business Events, import the generated WSDL file.
   - For WebSphere Business Monitor, import the XSD file from the **Generation** > **Targets** directory in your project.

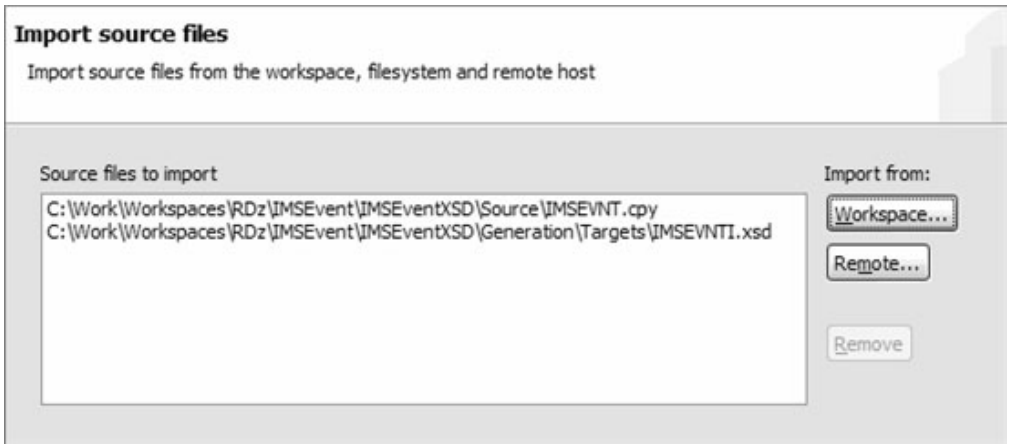


Figure 62. Importing the copybook file and the generated XSD file for WebSphere Business Monitor

5. Click **Finish**. The files are imported into the project, and are displayed under the `Source` directory.
6. Right-click the project and select **Create mappings**.
7. On the Create mappings page, select **Notification** as the operation type and click **Next**.
8. On the New XML to COBOL or PL/I Mapping Session page, use the default values.
   - For WebSphere Business Events, the copybook is the language source file, and the WSDL file is the XML target.
   - For WebSphere Business Monitor, the copybook is the language source file, and the XSD file is the XML target.

   Click **Next** to continue.
9. On the Root XML Element and Language Structure Selection page, select the appropriate **Source root element** and **Target root element**. Click **Next**.
10. Click **Finish**. A `Request.mapping` file is created and opened in the mapping tool.

11. In the mapping tool, map the fields from the COBOL copybook source and the target WSDL (WebSphere Business Events) or XSD (WebSphere Business Monitor) file.

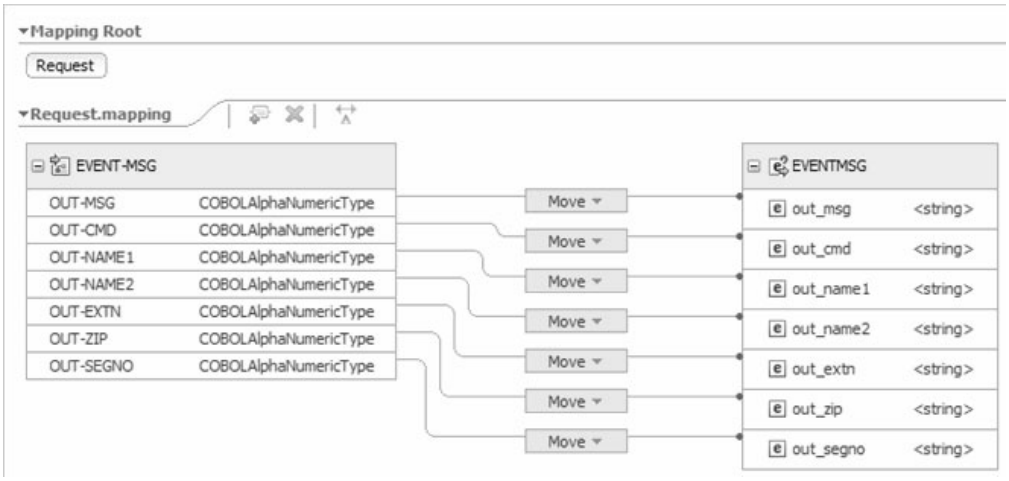


Figure 63. Mapping the data from the source (copybook) to the target (WSDL or XSD)

12. When you are done with the mapping, save the mapping file by clicking **File** > **Save.**

Figure 64. The created Request.mapping file in the Mapping directory

Generating the correlator file and the XML converter for business events

Use the request data mapping file that you generated previously to generate the correlator file and the XML converter for business events.

1. Start Rational Developer for System z.
2. In the Enterprise Service Tools perspective, open the IMS Enterprise Suite SOAP Gateway project that you just previously created a data mapping file for.
3. Right-click the project, and select **Generate IMS Enterprise Suite SOAP Gateway resources**. The Generation of conversion code window opens.
4. The request mapping file is automatically loaded. Click **Next**.
5. On the Generation options page, update the host code page if necessary. Click **Next**.
6. In the IMS Enterprise Suite SOAP Gateway Web Service Requestor page:
   a. For the **Callout message type** drop-down list, select **ASYNC Asynchronous**.
   b. For the **WS-Security** drop-down list, select **Disabled**.
   c. Click **Next**.

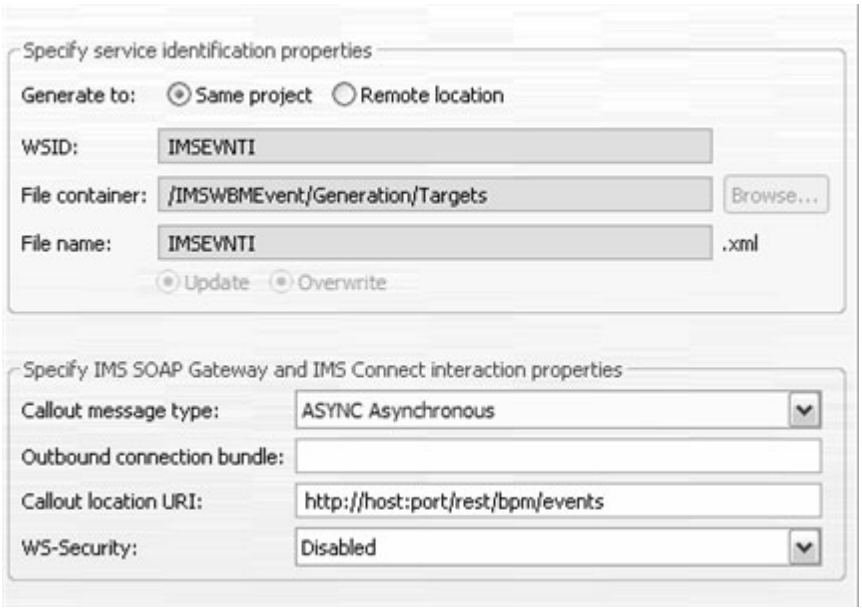


Figure 65. Specifying service identification and interaction properties

7. On the File, data set, or member selection page: take the default value, or update the converter driver file name is needed.
   a. In the **XML Converters** tab, update the converter driver file name if necessary.
   b. Ensure that the **Generate all to driver** check box is selected.
   c. Click **Finish**.

The callout (business event) correlator file and the XML converter are generated.

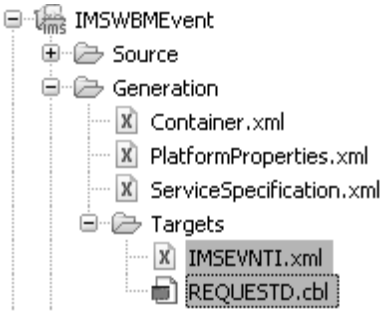


Figure 66. The generated correlator file and XML converter

Deploying the XML converter to IMS Connect

After you generate the XML converter by using Rational Developer for System z, you need to deploy it to IMS Connect.

Prerequisites:

1. The XML adapter function must be configured in IMS Connect. For more information, see IMS Version 13 System Definition information.
2. The Rational Developer for System z host configuration must be completed. For more information, see the *Rational Developer for System z Host Configuration Guide*.

3. The Rational Developer for System z SFEKLOAD (V8.0.*x*) or SFEKLMOD (V8.5 or later) module must be concatenated to the STEPLIB of the IMS Connect startup JCL in order to generate IRZ error messages to assist troubleshooting.

To deploy the XML converter to IMS Connect:

1. Use FTP to copy the XML converter that is generated by Rational Developer for System z to your host data set.
2. Compile and bind the XML converter to the target host data set.

   To compile and bind the converters into a data set concatenated with the IMS Connect STEPLIB:

   - The converter must have an alias that is linked with the converter code, using the same name as the converter, with an X suffix. If the converter name is eight-character long, the last character for the alias must be changed to an X.
   - The converter and the load module name must end with the letter D.
   - The converter should be linked with the option REUS=SERIAL so that the converter is loaded into the memory only once.

   For example, if your converter file name is IMSSGWS1D. Your JCL would contain ENTRY, ALIAS, and NAME statements as follows:

```
//LINK    EXEC PGM=HEWL,COND=(4,LT),
//        PARM='XREF,COMPAT=MIN,REUS=SERIAL'
...
  ENTRY IMSSGWS1D
  ALIAS IMSSGWS1X
  NAME IMSSGWS1D(R)
```

Related tasks:

"Configuring IMS Connect for SOAP Gateway" on page 101
You must configure IMS Connect to allow SOAP Gateway to access IMS transactions.

Related information:

Rational Developer for System z Host Configuration Guide
For host configuration instructions, see the *Host Configuration Guide* for the version of Rational Developer for System z that you on the Rational Developer for System z library page.

Developing an application in WebSphere Business Events

Develop a WebSphere Business Events application to define the rules on how the event would be consumed and processed, and how what actions would be triggered.

Prerequisite: You must be familiar with WebSphere Business Events Design Data or WebSphere Business Events Design.

The WSDL definition can then be used to submit an event as a web service to the WebSphere Business Events Runtime.

Use WebSphere Business Events Design Data to define a touchpoint to be the container that holds all the events and actions. Business users can use WebSphere Business Events Design to create an event flow.

Related information:

Developing the components in WebSphere Business Events Design Data
See the WebSphere Business Events information center for more information about

how to develop the components.

Developing the business logic in WebSphere Business Events Design
See the WebSphere Business Events information center for more information about how to develop the business logic.

Configuring SOAP Gateway to emit business events

Use the SOAP Gateway management utility to deploy the correlator file and either the WSDL file or the XSD file for the business event server to which your IMS application is emitting business data.

Have the following files ready:

- The correlator file created in Rational Developer for System z.
- For WebSphere Business Events, the WSDL file that is generated by Rational Developer for System z.
- For WebSphere Business Monitor, the XSD file for the event emitter.

Tip: z/OS For z/OS systems, the correlator file and the WSDL file, when uploaded from a distributed platform, must be transferred in BINARY mode from your local workstation. Binary transfers provide a bit-by-bit copy that preserves the encoding on your system by instructing the FTP socket not to convert the encoding to the local system encoding (EBCDIC).

Important: The correlator file that is generated by Rational Developer for System z Version 9.0 or older versions is in a correlator schema older than version 3.0. The correlator file must be migrated to the new correlator scheme:

1. Store the correlator file in the *install_dir*/imssoap/xml directory.
2. Use the SOAP Gateway management utility iogmgmt -migrate correlator command to migrate the correlator file.

Before you run the migration tool, ensure that the calloutConnBundleName property in the version 1.0 correlator schema is not empty. If the calloutConnBundleName property is empty, the migration tool would assume that this correlator is for the web service provider scenario and set the correlator mode value to call-in instead of call-out.

To deploy web service artifacts for business events:

1. Issue the command `iogmgmt -deploy -w wsdl_or_xsd_file -r correlator_file`.
2. Verify the deployment with the `iogmgmt -view -CorrelatorFile ALL` command. The name of the newly deployed correlator file for the business event emitter appears in the list of active correlator files in the runtime cache.

SOAP Gateway is ready to send business events to the target business event server.

Start a callout thread to pull business event messages from the callout work queue, or restart the server.

Related tasks:

"Migrating correlator files to schema version 3.0" on page 302
IMS Enterprise Suite Version 3.1 SOAP Gateway requires correlator schema version 3.0. To migrate an existing correlator file from older versions to version 3.0, use the SOAP Gateway management utility iogmgmt -migrate correlator command.

"Starting the callout thread for a specific application" on page 268
You must start a callout thread after a callout application is deployed to SOAP Gateway before the application can begin processing callout messages.

Related reference:

"-migrate: Migrate and upgrade SOAP Gateway" on page 448
The -migrate command upgrades SOAP Gateway artifacts and settings to the latest version and generates a migration log.

"-deploy: Deploy a web service or callout application" on page 444
The -deploy command deploys a web service, callout application, or business event application to the active configuration of the SOAP Gateway server.

"-deploy: Deploy a web service or callout application" on page 444
The -deploy command deploys a web service, callout application, or business event application to the active configuration of the SOAP Gateway server.

Generating and deploying artifacts for emitting business events to WebSphere Business Monitor

Generate the web service artifacts in Rational Developer for System z, and deploy the generated files in SOAP Gateway and WebSphere Business Monitor.

These tasks describe how to enable an IMS application to emit business event data to WebSphere Business Monitor:

Related information:

WebSphere Business Monitor information center
WebSphere Business Monitor information center

Coding business event data and inserting the event emission point

You must first identify the event emission point in your IMS application and provide the event data in an existing or a new data structure.

The event data is placed on an OTMA hold queue with an identified OTMA destination descriptor, in the same way as asynchronous callout messages.

1. Create a new data structure or modify an existing one for business event data.
2. Use the ISRT ALTPCB DL/I call at the event emission point to place the business data on the hold queue that is defined in an OTMA destination descriptor.

Related concepts:

"Design guidelines for emitting business events" on page 200
You might want to add new fields to your data structure or generate the ALTPCB value in the PSB, depending on how the events are being processed and whether the IMS application has access to an ALTPCB.

Defining an OTMA destination descriptor for business events

Define an OTMA destination descriptor to route IMS business event data to a hold queue.

To use the OTMA destination descriptor:

1. Configure the descriptor in the DFSYDTx PROCLIB member.

2. Specify the ADAPTER and the CONVERTR values. Set the ADAPTER value to HWSXMLA0 if you want XML data transformation to be completed by IMS Connect. The following example OTMA destination descriptor spans multiple lines:

```
D SOAPGWAY TYPE=IMSCON TMEMBER=HWS2 TPIPE=HWS2SOAP
D SOAPGWAY ADAPTER=HWSXMLA0 CONVERTR=XMLCNVTR
```

   - The descriptor for SOAPGWAY routes messages to IMS Connect target member HWS2 with tpipe HWS2SOAP.
   - The ADAPTER is set to HWSXMLA0 for the data transformation to be performed by the IMS Connect XML adapter.
   - The XML converter name is XMLCNVTR.

   Important: Do not share the tpipe that you use for SOAP Gateway callout functions with business event data or callout functions in the IMS TM resource adapter.

Generating the XML schema file (XSD file)

An XSD file describes the XML scheme and is required for the exchange of XML messages between IMS and the business event server. Generate the XSD file by using IBM Rational Developer for System z.

1. Start Rational Developer for System z.
2. Open the Enterprise Service Tools Projects perspective if it is not already open.
   a. On the main menu, click **Window** > **Open Perspective** > **Other**. The Open Perspective window opens.
   b. Select **Enterprise Service Tools** and click **OK**.
3. Create a new Batch, TSO, z/OS UNIX Project.
   a. Click **File** > **New** > **Batch, TSO, z/OS UNIX Project**.
   b. Select the following options:
      - Project name: Specify a project name.
      - Development scenario: **Create New Service Interface (bottom-up)**
      - Application mode: **Service Provider**

      Click **Next**.
4. Import the IMS COBOL application source file that contains the event data structure and click **Finish**. The Language structures page appeared.
5. Generate an XML schema for the event data structure in the IMS application source file.
   a. On the Language structures page:
      - In the **Request Language Structure** tab, select the event data structure.
      - Leave the response language structure as is. Because no response is expected from the business event server, none of the check boxes in the response language structure should be selected.
      - Ensure that the COBOL preference is set to z/OS.
      - Click **Next**.
   b. On the Generation options page:
      1) Click the **WSDL and XSD** tab.
      2) In the **Specify request XML Schema properties** area, update the **Target Namespace** and the **Root Element name** fields if necessary.
      3) Click the **Advanced options** tab.

4) Ensure that the **Generate qualified elements in XML Schemas** check box is selected (default).
5) Click **Next**.

c. On the File, data set, or member selection page:
1) Click the **XML Converters** tab.
2) Deselect the **Generate all to driver** check box, and then clear the **Converter driver file name** check box, and the **Request Converter file name** check box.
3) Click the **WSDL and XSD** tab.
4) Clear the WSDL file name check box.
5) Ensure that the **Request XSD file name** check box is selected.
6) Update the XSD file name, if necessary.
7) Click **Finish**.

The XML Schema file (XSD) for the event data structure is generated in the `Targets` directory.

Generating the data mapping file for business events

IBM Rational Developer for System z to generate a request (IMS business event) mapping session file to map the data structures between the IMS application and the business event server for the business event data.

Prerequisites:

- The IMS application source file.
- For WebSphere Business Events, the web service WSDL file that is generated by WebSphere Business Events.
- For WebSphere Business Monitor, the XSD XML scheme file that is generated earlier by Rational Developer for System z.

1. Start Rational Developer for System z.
2. In the Enterprise Service Tools perspective, create an IMS SOAP Gateway project by clicking **File** > **New** > **IMS Enterprise Suite SOAP Gateway project**.
3. In the New IMS Enterprise Suite SOAP Gateway Project window:
   a. Specify a project name.
   b. For **Development scenario**, select **Map an Existing Service Interface (meet-in-middle)**.
   c. For **Application mode**, select **Service Requestor**.
   d. Click **Next**.
4. Import the IMS application source code (copybook) and either an XSD or WSDL file, depending on the target business event processing server. You can find the application files in the Targets directory in your project.
   - For WebSphere Business Events, import the generated WSDL file.
   - For WebSphere Business Monitor, import the XSD file from the **Generation** > **Targets** directory in your project.

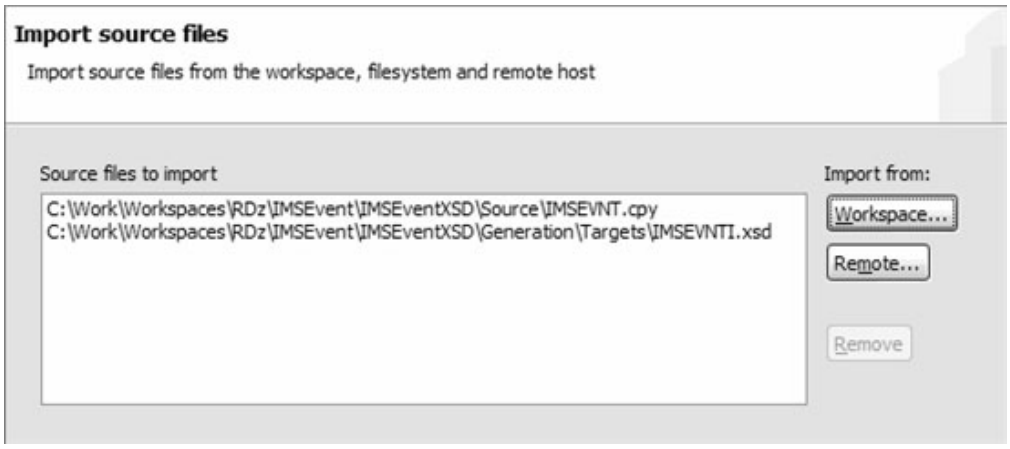


Figure 67. Importing the copybook file and the generated XSD file for WebSphere Business Monitor

5. Click **Finish**. The files are imported into the project, and are displayed under the `Source` directory.
6. Right-click the project and select **Create mappings**.
7. On the Create mappings page, select **Notification** as the operation type and click **Next**.
8. On the New XML to COBOL or PL/I Mapping Session page, use the default values.
   - For WebSphere Business Events, the copybook is the language source file, and the WSDL file is the XML target.
   - For WebSphere Business Monitor, the copybook is the language source file, and the XSD file is the XML target.

   Click **Next** to continue.
9. On the Root XML Element and Language Structure Selection page, select the appropriate **Source root element** and **Target root element**. Click **Next**.
10. Click **Finish**. A `Request.mapping` file is created and opened in the mapping tool.
11. In the mapping tool, map the fields from the COBOL copybook source and the target WSDL (WebSphere Business Events) or XSD (WebSphere Business Monitor) file.

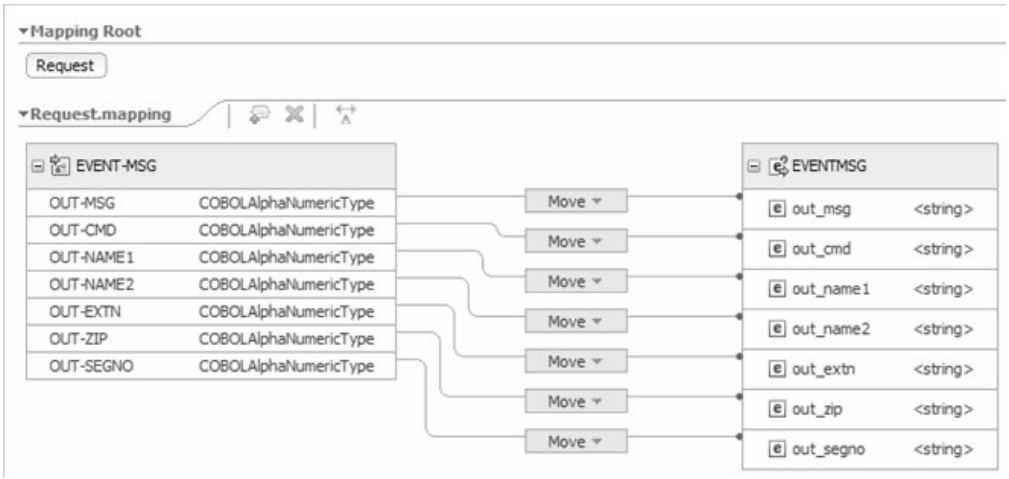


Figure 68. Mapping the data from the source (copybook) to the target (WSDL or XSD)

12. When you are done with the mapping, save the mapping file by clicking **File** > **Save.**

Figure 69. The created Request.mapping file in the Mapping directory

Generating the correlator file and the XML converter for business events

Use the request data mapping file that you generated previously to generate the correlator file and the XML converter for business events.

1. Start Rational Developer for System z.
2. In the Enterprise Service Tools perspective, open the IMS Enterprise Suite SOAP Gateway project that you just previously created a data mapping file for.
3. Right-click the project, and select **Generate IMS Enterprise Suite SOAP Gateway resources**. The Generation of conversion code window opens.
4. The request mapping file is automatically loaded. Click **Next**.
5. On the Generation options page, update the host code page if necessary. Click **Next**.
6. In the IMS Enterprise Suite SOAP Gateway Web Service Requestor page:
   a. For the **Callout message type** drop-down list, select **ASYNC Asynchronous**.
   b. For the **WS-Security** drop-down list, select **Disabled**.
   c. Click **Next**.

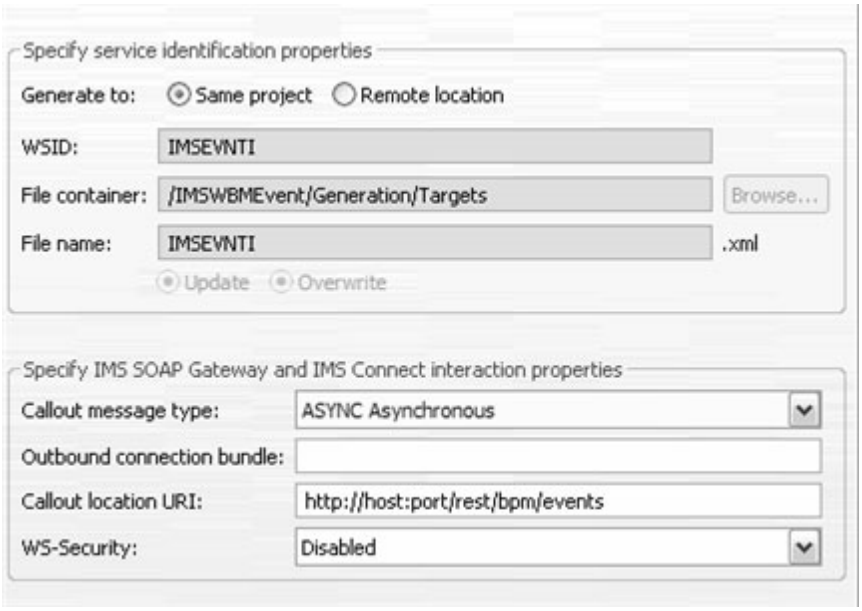


Figure 70. Specifying service identification and interaction properties

7. On the File, data set, or member selection page: take the default value, or update the converter driver file name is needed.
   a. In the **XML Converters** tab, update the converter driver file name if necessary.
   b. Ensure that the **Generate all to driver** check box is selected.
   c. Click **Finish**.

The callout (business event) correlator file and the XML converter are generated.

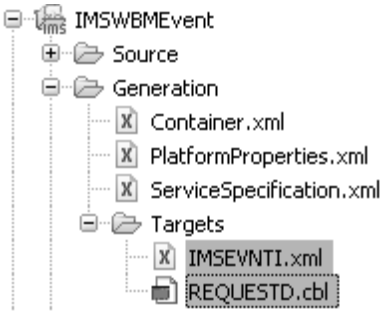


Figure 71. The generated correlator file and XML converter

Deploying the XML converter to IMS Connect

After you generate the XML converter by using Rational Developer for System z, you need to deploy it to IMS Connect.

Prerequisites:

1. The XML adapter function must be configured in IMS Connect. For more information, see IMS Version 13 System Definition information.
2. The Rational Developer for System z host configuration must be completed. For more information, see the *Rational Developer for System z Host Configuration Guide*.

3. The Rational Developer for System z SFEKLOAD (V8.0.*x*) or SFEKLMOD (V8.5 or later) module must be concatenated to the STEPLIB of the IMS Connect startup JCL in order to generate IRZ error messages to assist troubleshooting.

To deploy the XML converter to IMS Connect:

1. Use FTP to copy the XML converter that is generated by Rational Developer for System z to your host data set.
2. Compile and bind the XML converter to the target host data set.

   To compile and bind the converters into a data set concatenated with the IMS Connect STEPLIB:

   - The converter must have an alias that is linked with the converter code, using the same name as the converter, with an X suffix. If the converter name is eight-character long, the last character for the alias must be changed to an X.
   - The converter and the load module name must end with the letter D.
   - The converter should be linked with the option REUS=SERIAL so that the converter is loaded into the memory only once.

   For example, if your converter file name is IMSSGWS1D. Your JCL would contain ENTRY, ALIAS, and NAME statements as follows:

```
//LINK    EXEC PGM=HEWL,COND=(4,LT),
//        PARM='XREF,COMPAT=MIN,REUS=SERIAL'
...
  ENTRY IMSSGWS1D
  ALIAS IMSSGWS1X
  NAME IMSSGWS1D(R)
```

Related tasks:

"Configuring IMS Connect for SOAP Gateway" on page 101
You must configure IMS Connect to allow SOAP Gateway to access IMS transactions.

Related information:

Rational Developer for System z Host Configuration Guide
For host configuration instructions, see the *Host Configuration Guide* for the version of Rational Developer for System z that you on the Rational Developer for System z library page.

Setting up WebSphere Business Monitor for IMS business events

Before you configure SOAP Gateway to emit business event data, set up WebSphere Business Monitor to be ready to receive business events.

To set up WebSphere Business Monitor:

1. Create a business monitor project to create a monitor model.
2. Generate a J2EE project for the model and deploy it to the WebSphere Business Monitor Runtime.
3. Define the dashboard and Key Performance Indicator (KPI) to monitor the event.

Related information:

WebSphere Business Monitor information center
WebSphere Business Monitor information center

WebSphere Business Monitor tutorials and demos
WebSphere Business Monitor tutorials and demos on IBM Education Assistant

Configuring SOAP Gateway to emit business events

Use the SOAP Gateway management utility to deploy the correlator file and either the WSDL file or the XSD file for the business event server to which your IMS application is emitting business data.

Have the following files ready:

- The correlator file created in Rational Developer for System z.
- For WebSphere Business Events, the WSDL file that is generated by Rational Developer for System z.
- For WebSphere Business Monitor, the XSD file for the event emitter.

Tip: z/OS For z/OS systems, the correlator file and the WSDL file, when uploaded from a distributed platform, must be transferred in BINARY mode from your local workstation. Binary transfers provide a bit-by-bit copy that preserves the encoding on your system by instructing the FTP socket not to convert the encoding to the local system encoding (EBCDIC).

Important: The correlator file that is generated by Rational Developer for System z Version 9.0 or older versions is in a correlator schema older than version 3.0. The correlator file must be migrated to the new correlator scheme:

1. Store the correlator file in the *install_dir*/imssoap/xml directory.
2. Use the SOAP Gateway management utility iogmgmt -migrate correlator command to migrate the correlator file.

Before you run the migration tool, ensure that the calloutConnBundleName property in the version 1.0 correlator schema is not empty. If the calloutConnBundleName property is empty, the migration tool would assume that this correlator is for the web service provider scenario and set the correlator mode value to call-in instead of call-out.

To deploy web service artifacts for business events:

1. Issue the command `iogmgmt -deploy -w wsdl_or_xsd_file -r correlator_file`.
2. Verify the deployment with the `iogmgmt -view -CorrelatorFile ALL` command. The name of the newly deployed correlator file for the business event emitter appears in the list of active correlator files in the runtime cache.

SOAP Gateway is ready to send business events to the target business event server.

Start a callout thread to pull business event messages from the callout work queue, or restart the server.

Related tasks:

"Migrating correlator files to schema version 3.0" on page 302
IMS Enterprise Suite Version 3.1 SOAP Gateway requires correlator schema version 3.0. To migrate an existing correlator file from older versions to version 3.0, use the SOAP Gateway management utility iogmgmt -migrate correlator command.

"Starting the callout thread for a specific application" on page 268
You must start a callout thread after a callout application is deployed to SOAP Gateway before the application can begin processing callout messages.

Related reference:

"-migrate: Migrate and upgrade SOAP Gateway" on page 448
The -migrate command upgrades SOAP Gateway artifacts and settings to the latest version and generates a migration log.

"-deploy: Deploy a web service or callout application" on page 444
The -deploy command deploys a web service, callout application, or business event application to the active configuration of the SOAP Gateway server.

"-deploy: Deploy a web service or callout application" on page 444
The -deploy command deploys a web service, callout application, or business event application to the active configuration of the SOAP Gateway server.

Chapter 8. Administering the SOAP Gateway server

Administer the SOAP Gateway server with the SOAP Gateway management utility.

Administering SOAP Gateway

Use the SOAP Gateway management utility to:

- Change SOAP Gateway server properties.
- Create and deploy artifacts that SOAP Gateway uses to enable IMS applications as web services.
- Create and deploy artifacts that SOAP Gateway uses to send IMS callout requests and business events to external web services.

  Restriction: Creation of correlator XML files for WebSphere Business Monitor and WebSphere Business Events is supported only with Rational Developer for System z.

The SOAP Gateway management utility supports execution of multiple commands in batch mode by using the iogmgmt -batch command. This feature facilitates service deployment and server administration tasks by executing the commands in one JVM instance. For more information about the batch mode support, see the SOAP Gateway management utility -batch command reference.

Related concepts:

"SOAP Gateway master configuration and runtime configuration" on page 25
The master configuration is the authoritative configuration of the SOAP Gateway server, and is stored in the file system. The active server configuration in memory is the runtime configuration.

Related reference:

"-batch: Run management utility commands in batch mode" on page 430
The -batch command runs multiple SOAP Gateway management utility commands as a batch in one JVM instance.

Invoking the SOAP Gateway management utility on z/OS

z/OS

You must issue SOAP Gateway management utility commands from the z/OS UNIX shell.

SOAP Gateway must be configured with a path to a compatible IBM SDK for Java before you can start the management utility. By default, the SDK is installed with the Base Services component of the IMS Enterprise Suite and the server is pre-configured with the correct path. If the server is using a different SDK installation, you must set the path with the `iogmgmt -prop -u -java -h` *`java_sdk_directory`* command.

The SOAP Gateway management utility, iogmgmt, is only available from within a UNIX System Services (USS) session. USS is a required component of z/OS.

1. Invoke a USS session with the `OMVS` command.

2. Switch to the SOAP Gateway management utility directory at *-PathPrefix-*/usr/lpp/ims/imses/V*x*R*x*/soap_gw/imsserver/deploy with the cd command.
3. Invoke the SOAP Gateway management utility by entering ./iogmgmt *arguments* for each command.

Related concepts:

"SOAP Gateway management utility" on page 26
The SOAP Gateway management utility provides a command line or batch interface for configuring server properties, managing the server run time, and working with web service artifacts.

OMVS command shell session
See more information about OMVS command shell sessions in the z/OS basic skills information center.

Related reference:

Chapter 11, "SOAP Gateway management utility reference," on page 429
The SOAP Gateway management utility provides a command line interface to manage the SOAP Gateway server runtime, configure server properties, and work with web service artifacts.

SOAP Gateway server startup options

There are different methods to start the SOAP Gateway server depending on the host operating system.

Select one of the following startup methods:

- Linux Windows Use the SOAP Gateway management utility iogmgmt -start command.
- Windows From the Windows Start menu, select: **Start** > **All Programs** > **IBM IMS Enterprise Suite V3.1** > **SOAP Gateway** > **Start Server**.

  If SOAP Gateway server is installed as a Windows service, use the SOAP Gateway management utility iogmgmt -service -start command.
- Windows On Windows 7 systems, depending on the SOAP Gateway installation directory, you might need to start the SOAP Gateway server as an administrator. To start the SOAP Gateway server as an administrator:
  - From the Windows Start menu, select **Start** > **All Programs** > **IBM IMS Enterprise Suite V3.1** > **SOAP Gateway**. Right-click **Start Server** and select **Run as administrator**.
  - Or, open a command prompt as an administrator by selecting **Start**, right-clicking **Command Prompt** , and selecting **Run as administrator**. In the command prompt, change directories to the SOAP Gateway management utility directory (*install_dir*/imsserver/deploy). Then issue the iogmgmt -start command.
- Linux Run the iogstart.sh script in the *install_dir*/imsserver/bin directory. This script starts the server as a TTY application.
- z/OS Use the /S AEWIOGPR procedure. Before using the procedure, you must customize it according to the guidelines given in the procedure comments. The procedure is shipped in the SAEWBASE data set.

Related tasks:

"Deploying a web service" on page 229
Use the SOAP Gateway management utility to create a connection bundle and to

deploy an IMS application as a web service.

Related reference:

"-start: Start the SOAP Gateway server" on page 456
The -start command starts the SOAP Gateway server.

SOAP Gateway server shutdown options

There are different methods to stop the SOAP Gateway server, depending on the host operating system.

When you stop the server, SOAP Gateway attempts to shut down the server gracefully by:

1. Blocking all subsequent incoming web service provider requests. Subsequent provider requests to the server before it is completely shutdown would receive an IOGS0125E message.
2. Interrupting the IMS Connect socket that is waiting for callout requests from IMS, to stop further IMS callout requests or business event emission destined for SOAP Gateway.
3. Processing all in-flight messages.
4. Stopping the thread pool.
5. Shutting down the server.

SOAP Gateway waits for a maximum of 5 minutes for graceful processing of in-flight messages and stopping all callout threads and the thread pool. If the graceful shutdown does not happen in 5 minutes, the server would force the shutdown. You can also force the shutdown if for some reason the server is not stopped, or you must shut down the server immediately without waiting for the server to finish processing in-flight messages.

z/OS

- For a graceful shutdown, use the STOP AEWIOGPR procedure to initiate a graceful shutdown of the server. AEWIOGPR is the default job name. Replace it with your job name if it is renamed.
- To force an immediate shutdown, use CANCEL AEWIOGPR.

Windows

- If the SOAP Gateway server is installed and run as a Windows service:
  - For a graceful shutdown, use the SOAP Gateway management utility iogmgmt -service -stop command.
  - To force an immediate shutdown, use the SOAP Gateway management utility iogmgmt -service -stop -force command.
- If the SOAP Gateway server is not run as a Windows service:
  - For a graceful shutdown:
    - Use the SOAP Gateway management utility iogmgmt -stop command
    - Use the desktop shortcut, **Start** > **All Programs** > **IBM IMS Enterprise Suite V3.1** > **SOAP Gateway** > **Stop Server**.
  - Forced immediate shutdown is not supported, because the server does not run in the background. Closing the console window or pressing Ctrl-C on the keyboard would initiate a graceful shutdown.

Linux

- For a graceful shutdown:
  - Issue the iogmgmt -stop command.
  - Run the `iogstop.sh` script in the *install_dir*`/imsserver/bin`.
- To force an immediate shutdown, Issue the iogmgmt -stop -force command.

Important: If the server is started by using the `iogstart.sh` script, the server does not run in the background. Closing the console window or pressing Ctrl-C on the keyboard would initiate a graceful shutdown. Forced immediate shutdown is not supported in this case

Related reference:

"-stop: Stop the SOAP Gateway server" on page 456
The -stop command stops the SOAP Gateway server.

"-service -stop: Stop the SOAP Gateway server as a Windows service" on page 455
Use the -service -stop command to stop the SOAP Gateway server as a Windows service.

Managing the SOAP Gateway service as a Windows service

Windows

Several SOAP Gateway management utility commands are provided for installing, uninstall, starting, stopping, and viewing the status of the SOAP Gateway server as a Windows service.

- To register the SOAP Gateway server as a Windows service, use the SOAP Gateway management utility iogmgmt -service -install command.
- To remove the SOAP Gateway server as a Windows service registry, use the SOAP Gateway management utility iogmgmt -service -uninstall command.
- To start the SOAP Gateway server as a Windows service after it is properly installed, use the SOAP Gateway management utility iogmgmt -service -start command.
- To stop the SOAP Gateway server as a Windows service, use the SOAP Gateway management utility iogmgmt -service -stop command.
- To view the status of the SOAP Gateway server as a Windows service, use the SOAP Gateway management utility iogmgmt -service -status command.

Related reference:

"-service -install: Install the SOAP Gateway server as a Windows service" on page 453
Use the -service -install command to install and register the SOAP Gateway server as a Windows service.

"-service -start: Start the SOAP Gateway server as a Windows service" on page 454
Use the -service -start command to start the SOAP Gateway server as a Windows service.

"-service -status: View the SOAP Gateway server status as a Windows service" on page 454
Use the -service -status command to view the SOAP Gateway server status as a Windows service.

"-service -stop: Stop the SOAP Gateway server as a Windows service" on page 455
Use the -service -stop command to stop the SOAP Gateway server as a Windows service.

"-service -uninstall: Unistall the SOAP Gateway server as a Windows service" on page 455
Use the -service -uninstall command to remove the SOAP Gateway server as a Windows service.

Viewing deployed web services

Start the SOAP Gateway Administrative Console to view your deployed web services, or use the SOAP Gateway management utility.

The SOAP Gateway server must be started before proceeding.

If the SOAP Gateway is stopped, issue the `iogmgmt -view -correlatorfile ALL` command. The web-based Administrative Console is not available if the server is stopped. However, this command retrieves information from the master (file system) configuration of a stopped server instead of the runtime configuration.

1. To start the SOAP Gateway Administrative Console:
   - Windows For Windows: From the Start menu, select **Start** > **Programs** > **IBM IMS Enterprise Suite V*x.x*** > **SOAP Gateway** > **Administrative Console**.
   - z/OS Linux For Linux on System z, and z/OS: From a web browser, type: `http://hostname:port/imssoap`

     where *hostname* is the hostname and *port* is the port number where SOAP Gateway is running. The default port number is 8080.

   The Administrative Console opens in a web browser.

   Tip: If you start the Administrative Console from the same workstation on which SOAP Gateway is installed, you can use the default port number (8080) and `http://localhost:8080/imssoap` as the URL.
2. Click **View Deployed web services**. The list of the currently deployed web services is displayed. Each item in the list is a link to the web service's WSDL file.

Related concepts:

"SOAP Gateway administrative console" on page 27
The SOAP Gateway administrative console lists the deployed web services when the server is started. Each item in the list is a link to the web services description language (WSDL) file for the web service.

Changing the port number of the SOAP Gateway server

To change the port number of SOAP Gateway, use the SOAP Gateway management utility.

By default, SOAP Gateway is configured to listen for SOAP requests on HTTP port 8080. If the HTTPS port is enabled, the default is port 8443.

Important: z/OS z/OS requires a separate shutdown port number. By default the listening shutdown port number is set to 8005. If you have multiple copies of SOAP Gateway running on a single z/OS system, all ports must be unique.

To change the default listening port number:

1. Issue the SOAP Gateway management utility iogmgmt -prop -u -p *XXXX* command, where *XXXX* is the new port number. After the port number

changes, the URL used to access the web services running on SOAP Gateway is changed. Therefore, you must modify the WSDL file and the URL you use to access the web service.

2. Modify the port information in the WSDL file.
3. Change the URL that is used to access the web services in your client applications.

After the port number is changed, restart the SOAP Gateway server.

Related reference:

"-prop: Set SOAP Gateway properties" on page 450
Use the -prop command to modify the SOAP Gateway server properties.

Connection bundle management

Manage connection bundle entry names and usage by web services to ensure stable and predictable SOAP Gateway server behavior.

A connection bundle entry contains properties that define how SOAP Gateway interacts with IMS Connect host systems and with external web services. Each connection bundle entry has a unique name. However, a single connection bundle entry can contain properties for both callout applications and web services. Any number of correlator XML files, and therefore any number of web services or callout applications, can share a single connection bundle entry. To ensure that SOAP Gateway web services and applications behave predictably, it is important to understand how the SOAP Gateway server handles commands that affect connection bundle behavior.

Server startup behavior

When the SOAP Gateway server starts, all valid connection bundles in the master configuration are loaded into the server runtime configuration. A connection bundle entry is considered valid if it references a valid WSDL file.

Behavior of SOAP Gateway management utility commands sent to an active SOAP Gateway server

Active connection bundles are protected to ensure the integrity of web services and in-flight messages on the server. Commands from the SOAP Gateway management utility to create, modify, or delete connection bundles are carried out in the master configuration and do not take effect until the next time the SOAP Gateway starts.

When a web service or callout application is deployed to an active SOAP Gateway, the associated correlator XML file is checked for the connection bundle name or callout connection bundle name that is used by the web service. If the specified connection bundle does not exist in the runtime configuration, it is loaded from the master configuration so that the web service is immediately available for use. However, if the specified connection bundle does exist in the runtime configuration, it is not updated from the master configuration. If your web service requires a change to the runtime version of the connection bundle, you can either:

- Change the connection bundle entry properties and then restart the server. The updated properties are propagated from the master configuration to the runtime configuration when the server starts.
- Create a connection bundle entry with an unused name, and modify the correlator XML file for the new web service or callout application to use the new

connection bundle entry. The new connection bundle entry is loaded in the runtime configuration when the web service or application is deployed.

Behavior of shared connection bundle entries

A single connection bundle entry can be shared by any number of web services or callout applications. Referencing a single connection bundle entry in multiple correlator XML files can improve the manageability of the server. However, consider the following guidelines when you configure multiple web services to share a single connection bundle entry:

- A connection bundle entry is only valid for callout applications if it specifies at least one callout tpipe name. Ensure that all of the listed tpipe names are valid for all of the callout applications that use the connection bundle entry. Secure callout web services must use a different tpipe from non-secure callout web services.
- A connection bundle entry with callout tpipe names cannot be used for a web service provider correlator
- Each connection bundle entry can specify only one IMS Connect host system, port number, and data store.
- A connection bundle entry can contain only one set of security authentication properties. However, WS-Security is service-specific and is not defined in the connection bundle entry.

Related concepts:

"Connection bundle properties" on page 20
The connection bundle specifies the connection and security properties for SOAP Gateway when it communicates with IMS Connect.

Related reference:

"-conn: Create, update, or delete a connection bundle" on page 435
Use the -conn command to create, update, or delete a connection bundle.

Connections and connection pools

The SOAP Gateway server creates one connection pool for each unique IMS Connect host name and port number that it communicates with. Each connection pool can contain multiple idle connections. SOAP Gateway manages the connection pools internally.

Connection error recovery

Connection errors between SOAP Gateway and the target IMS Connect can occur in several circumstances. IMS Connect might be unreachable because of a network problem, it might be restarting, or it might be experiencing a severe internal error that causes it to refuse connections. When the IMS Connect host is unreachable or is recycled, the SOAP Gateway client application receives an IOGC003E error message for the request to SOAP Gateway. To prevent duplicate messages, SOAP Gateway does not resubmit the message on behalf of the client. After the first such error message for a given IMS Connect host, SOAP Gateway notes the time of failure in the connection pool for the host.

On subsequent attempts to connect to the same host with a reused connection, SOAP Gateway first attempts to ping the target IMS Connect. If the ping succeeds, a new connection is created and the client message is sent. Because of this internal

checking, client applications should anticipate one connection error message due to prior IMS Connect outage. Subsequent attempts to connect succeed if the host becomes available.

If multiple messages are in-flight on the same connection when a connection error occurs, all of the messages are affected and discarded, and the client would receive multiple error messages.

Idle connection cleanup

A SOAP Gateway server is an IMS Connect client application in both the callout and provider scenarios. Therefore, each time the SOAP Gateway server processes a transaction, it must either create or reuse a connection to the destination IMS Connect instance. A connection is either created at the time of the initial service request, or an idle connection is taken from the connection pool for that IMS Connect host and reused. This connection is used to send and receive message information about the SOAP message and (for a synchronous callout message) the response from IMS. After the transaction is completed, the SOAP Gateway server places the connection into the connection pool for that IMS Connect host. When new SOAP messages are received, the server first attempts to reuse an idle connection for that IMS Connect host rather than creating a connection. If no idle connections are available, a new connection is created automatically.

Over time, the number of idle connections held in the connection pool can grow to the maximum number of concurrent connections ever made to the IMS Connect host. You can enable idle connection cleanup to improve server performance. When idle connection cleanup is enabled, the SOAP Gateway server periodically checks the connection pools for connections that have been idle for a specified length of time and deletes them.

By default, idle connection cleanup is disabled. You can enable it with the iogmgmt -prop command. The command provides three parameters to configure idle connection cleanup:

-r The number of minutes between idle connection checks. More frequent idle connection checks result in more server resource usage. The default value is 0, which disables idle connection cleanup.

-v The number of minutes a connection must remain idle before it is flagged for cleanup. The default value is 20.

-m The minimum number of idle connections to keep in the connection pool for each IMS Connect host. The SOAP Gateway server does not automatically create this many connections to each IMS Connect host. The value is used only as a lower limit when SOAP Gateway is cleaning up existing connections. The default value is 0.

You must restart the SOAP Gateway server after changing the server properties.

Related reference:

"-prop: Set SOAP Gateway properties" on page 450
Use the -prop command to modify the SOAP Gateway server properties.

Configuring compliance for FIPS 140-2 and NIST SP800-131a

You can configure SOAP Gateway to communicate with its clients and IMS Connect over secure sockets by using Java Secure Socket Extension files that are required by FIPS 140-2. In addition, NIST SP800-131a requires the use of TLS V1.2.

Prerequistie:

To enable FIPS, you need to specify to use the IBM Java Cryptographic Extension (JCE) FIPS Provider, IBMJCEFIPS. This cryptographic module supports FIPS-approved cryptographic operations through the Java APIs.

1. Stop the SOAP Gateway server if it is running.
2. Specify to use the IBMJCEFIPS FIPS provider. To do so, modify the `java.security` file in the IBM Java SDK to enable FIPS.

   Important: The following steps are required each time you update the IBM Java SDK.

   a. Go to *java_install_dir*`/jre/lib/security` directory.
   b. Save a copy of the existing `java.security` file as a backup. For example, name the backup copy `java.security.nofips`.
   c. In a text editor, open the `java.security` file and make the following changes.
      1) Find the following the list of security providers. The list might look as follows:

```
security.provider.1=com.ibm.jsse2.IBMJSSEProvider2
security.provider.2=com.ibm.crypto.provider.IBMJCE
security.provider.3=com.ibm.security.jgss.IBMJGSSProvider
security.provider.4=com.ibm.security.cert.IBMCertPath
security.provider.5=com.ibm.security.sasl.IBMSASL
security.provider.6=com.ibm.xml.crypto.IBMXMLCryptoProvider
security.provider.7=com.ibm.xml.enc.IBMXMLEncProvider
security.provider.8=com.ibm.security.jgss.mech.spnego.IBMSPNEGO
security.provider.9=sun.security.provider.Sun
```

      2) Add the following line as the first line to enable FIPS:

```
security.provider.1=com.ibm.crypto.fips.provider.IBMJCEFIPS
```

      3) Because the IBMJCEFIPS provider is now the first provider, increase the existing provider numbers by 1:

```
security.provider.1=com.ibm.crypto.fips.provider.IBMJCEFIPS
security.provider.2=com.ibm.jsse2.IBMJSSEProvider2
security.provider.3=com.ibm.crypto.provider.IBMJCE
security.provider.4=com.ibm.security.jgss.IBMJGSSProvider
security.provider.5=com.ibm.security.cert.IBMCertPath
security.provider.6=com.ibm.security.sasl.IBMSASL
security.provider.7=com.ibm.xml.crypto.IBMXMLCryptoProvider
security.provider.8=com.ibm.xml.enc.IBMXMLEncProvider
security.provider.9=com.ibm.security.jgss.mech.spnego.IBMSPNEGO
security.provider.10=sun.security.provider.Sun
```

      4) Towards the end of the file, locate the section that specifies the default key and trust manager factory algorithms for SSL.

```
ssl.KeyManagerFactory.algorithm=IbmX509
ssl.TrustManagerFactory.algorithm=PKIX
```

      5) Append the following two lines:

```
ssl.SocketFactory.provider=com.ibm.jsse2.SSLSocketFactoryImpl
ssl.ServerSocketFactory.provider=com.ibm.jsse2.SSLServerSocketFactoryImpl
```

      The resulting section might look as follows:

```
ssl.KeyManagerFactory.algorithm=IbmX509
ssl.TrustManagerFactory.algorithm=PKIX
ssl.SocketFactory.provider=com.ibm.jsse2.SSLSocketFactoryImpl
ssl.ServerSocketFactory.provider=com.ibm.jsse2.SSLServerSocketFactoryImpl
```

6) For NIST SP800-131a compliance, check if the following lines are present. If not, add them to the end of the file. These lines disable cryptographic algorithms that are deemed unacceptable by the SP800-131a standard.

```
jdk.tls.disabledAlgorithms = RSA keySize < 2048, DSA keySize < 2048, EC keySize < 224, MD5
jdk.certpath.disabledAlgorithms = RSA keySize < 2048, DSA keySize < 2048, EC keySize < 224, SHA1, MD5
```

d. Save your changes.

3. Specify to use the FIPS provider module, and for NIST SP800-131a, to set TLS v1.2 as the protocol for both HTTPS and SSL communications in the `server.xml` file.

a. Edit the `server.xml` file in the *install_dir*`/imsbase/conf/master` directory. In the Connector element, add the following attributes and values if they do not exist yet:

```
SSLEnabled="true"
sslEnabledProtocols="TLSv1.2"
```

Tip: TLS v1.2 is required for NIST SP800-131a. If the sslEnabledProtocols attribute value contains a spelling error or the cases do not match, the server automatically adopts a lower SSL if multiple protocols are specified.

b. Remove `sslProtocol="SSL"` if this attribute exists in the `server.xml` file.

c. Go to the *install_dir*`/imsserver/bin` directory.

d. Take one of the following steps, depending on your platform.

- z/OS Modify the AEWIOGCF sample JCL job as described in Step 3 in "Configuring SOAP Gateway on z/OS" on page 80, if you have not yet done so.

  This file contains several settings to enable the FIPS provider and TLS V1.2 for communication wit IMS Connect and SOAP Gateway clients.

  1) Remove the # sign in the beginning of the following line to uncomment it:

  ```
  # IJO="$IJO -Dcom.ibm.jsse2.usefipsprovider=true"
  ```

  2) Remove the # sign in the beginning of the following line to uncomment it to enable the SSL support for communications between SOAP and IMS Connect:

  ```
  # IJO="$IJO -Dcom.ibm.ims.soap.sslProtocolType=TLSv1.2"
  ```

  3) Remove the # sign in the beginning of the following line to uncomment it to enable HTTPS for communications between the external server and SOAP Gateway in the callout scenario:

  ```
  # IJO="$IJO -Dcom.ibm.ims.soap.httpsProtocolType=TLSv1.2"
  ```

  4) Remove the # sign in the beginning of the following line to uncomment it to assist troubleshooting security handshake issues:

  ```
  # IJO="$IJO -Djavax.net.debug=ALL"
  ```

  5) Add the following line for NIST SP800-131a compliance:

  ```
  IJO="$IJO -Dcom.ibm.jsse2.sp800-131=strict"
  ```

  6) Save your changes.

- Linux Windows Replace the SOAP Gateway startup batch file or shell script with the version that supports FIPS settings.

  The FIPS-enabled versions are:

– `iogstart.sh.secure`
– `iogstart.bat.secure`

These files are provided so minimal manual modification is needed, which could be error-prone.

1) Linux For Linux for System z:
   a) Rename `iogstart.sh` to `iogstart.sh.nonfips`.
   b) Rename the FIPS-compliant version of the server startup shell script `iogstart.sh.secure` to `iogstart.sh`.
2) Windows For Windows:
   a) Rename `iogstart.bat` to `iogstart.bat.nonfips`.
   b) Rename the FIPS-compliant version of the server startup batch file `iogstart.bat.secure` to `iogstart.bat`.
3) For NIST SP800-131a compliance, in the new `iogstart.sh` or `iogstart.bat`, search for the following line:

   ```
   -Dcom.ibm.ims.soap.sslProtocolType=TLSv1.2
   ```

   There are two instances of this line. Append the following line to both instances:

   ```
   -Dcom.ibm.jsse2.sp800-131=strict
   ```

   The modified line would look as follows:

   ```
   ... -Dcom.ibm.ims.soap.sslProtocolType=TLSv1.2 -Dcom.ibm.jsse2.sp800-131=strict ...
   ```
4) Save your changes.

4. Restart the server.

SOAP Gateway is enabled for FIPS.

In addition to enabling FIPS on the SOAP Gateway server:

- IMS Connect must be configured to match the required version of TLS (TLS v1.2 for NIST SP800-131a) and required cipher suites strength.

  For more information about IMS Connect SSL setup, see the "Configuring IMS Connect for SOAP Gateway" on page 101 topic.
- Connection bundles must be created or updated to use the STRONG encryption type for SSL connections with IMS Connect.

  For more information about IMS Connect SSL setup, see the "-conn: Create, update, or delete a connection bundle" on page 435 topic.
- For the callout scenario, the connection bundle must include the provider keystore and truststore information.
- For client applications:
  – Java clients must be run with FIPS enabled. If you are using the IBM Java SDK in IMS Enterprise Suite, turn on the Java FIPS provider flag by using the following system property:

    ```
    -Dcom.ibm.jsse2.usefipsprovider=true
    ```

    You must also use the `java.security.fips` file for the pre-configured security settings.

    For other versions of Java, check their documentation.
  – For NIST SP800-131a, the keystore and truststore that you create must meet the minimum requirements of 112-bit key strength or a key length of 2048. Java clients must be run with the HTTPS protocol flag turned on by using the following system property:

```
-Dhttps.protocols=TLSv1.2
```

– For ease of troubleshooting, you might want to turn on debugging:

```
-Djavax.net.debug=ALL
```

Related concepts:

"FIPS 140-2 and NIST SP800-131a" on page 39
Federal Information Processing Standards (FIPS) are standards and guidelines issued by the United States National Institute of Standards and Technology (NIST) for federal government computer systems. FIPS can be enabled for SOAP Gateway.

Related tasks:

"Creating the server keystore for SOAP Gateway and exporting the public key as a certificate" on page 150
Create a server keystore for SOAP Gateway and export the public key as a server certificate that the SOAP Gateway client can use to verify that the server is trusted.

"Creating the server truststore for SOAP Gateway" on page 152
Create a truststore for SOAP Gateway to store the HTTPS client certificates, or the SSL server certificate (from IMS Connect).

"Exporting the certificate from IMS Connect" on page 152
Use the RACDCERT command to export the certificate to a data set.

"Example: Configuring the client authentication and basic authentication security scheme" on page 192
This example demonstrates how to create self-signed certificates to configure client authentication and basic authentication when the web service is hosted on an Apache Tomcat server on Windows. The actual location of the key management utility might be different based on your server environment.

Migrating correlator files to schema version 3.0

IMS Enterprise Suite Version 3.1 SOAP Gateway requires correlator schema version 3.0. To migrate an existing correlator file from older versions to version 3.0, use the SOAP Gateway management utility iogmgmt -migrate correlator command.

This **correlator** keyword indicates that only the correlator files are to be migrated. The correlator files to be migrated must be stored in the SOAP Gateway XML directory, *install_dir*/imssoap/xml. Correlator files that are generated by Rational Developer for System z Version 8.0.3.*x* are of version 1.0 of the schema. Correlator files that are generated by Rational Developer for System z Version 8.5.*x* and Version 9.0 are of version 2.0. Correlator files of both versions must be migrated to version 3.0.

The migration tool works by:

1. Copying existing non-3.0 schema correlator files to the imssoap/tools/migration/xml/ directory as a backup.
2. Iterating through the files in the *install_dir*/imssoap/tools/migration/xml backup directory, migrating the correlator schema, and storing the migrated new correlator files into the *install_dir*/imssoap/xml directory.

New correlator files that are successfully migrated are stored back into the *install_dir*/imssoap/xml directory. All the original correlator files remain in the backup directory.

To migrate a correlator file to the new version 3.0 schema:

1. Put your correlator file in the *install_dir*/imssoap/xml.
2. Go to the directory where the SOAP Gateway management utility is at: *install_dir*/imsserver/deploy.
3. Issue the following command:

   z/OS Linux

   ```
   ./iogmgmt -migrate correlator
   ```

   Windows

   ```
   iogmgmt -migrate correlator
   ```

If the migration fails for a correlator file, to rectify:

1. Check and correct the correlator file in the backup directory.
2. Move the corrected correlator files into the *install_dir*/imssoap/xml directory.

Important: For efficiency, the migration tool checks and migrates a correlator file in the SOAP Gateway XML directory only if there is no file of the same name in the backup directory. When you rerun the migration tool on an updated correlator, always put the updated correlator file in the SOAP Gateway XML directory, and remove the old copy in the backup directory.

Tip: Always save a backup copy of your correlator, WSDL, and XSD files. When a web service is deployed in the provider scenario, the WSDL and any referenced XSD files are bundled into the web service archive files (.aar files). When the web service is undeployed, the web service archive files and the correlator files are removed from the imssoap/WEB-INF/services/ and imssoap/xml/ directories. Saving a copy of the correlator, WSDL, and XSD files in your own backup directory allows you to redeploy a web service later.

Related tasks:

"Migrating from IMS Enterprise Suite Version 2.1 SOAP Gateway" on page 104
After you install IMS Enterprise Suite Version 3.1 SOAP Gateway, migrate your web service files and server properties by using the SOAP Gateway management utility iogmgmt -migrate command.

Related reference:

"-migrate: Migrate and upgrade SOAP Gateway" on page 448
The -migrate command upgrades SOAP Gateway artifacts and settings to the latest version and generates a migration log.

SOAP Gateway logs

SOAP Gateway provides a server log with diagnostic information and a transaction log that records message traffic.

The following table shows a comparison of the SOAP Gateway logs:

Table 33. Comparison of SOAP Gateway server logging features

Feature	Purpose
Server log	The server log provides diagnostic and usage information for SOAP Gateway server administrators. This log includes messages from the SOAP Gateway management utility. The server log file is stored on the local file system.

Table 33. Comparison of SOAP Gateway server logging features (continued)

Feature	Purpose
Transaction log	The transaction log provides a complete record of every request made to a SOAP Gateway web service provider (provider scenario) or an external web service (callout scenario). The information for each transaction includes the request and response processing details for IMS Connect and the target (provider) or source (callout) IMS application. This information can be used for diagnostic purposes by the SOAP Gateway server administrator, by IMS and IMS Connect administrators, and by client application developers. Because the transaction log contains a record of every message processing event associated with the requests, it is also useful for auditing the activity of SOAP Gateway applications and gathering performance data. When transaction logging is enabled by using the iogmgmt -tranLog command, the transaction log file is stored on the local file system. The server administrator must manage the log files to ensure that server has adequate storage space. SOAP Gateway can also send the information to a remote transaction collector with theIBM Tivoli Composite Application Manager for Transactions (ITCAM) Transaction Tracking API (TTAPI) by using the iogmgmt -tranAgent command.

Setting the trace level for SOAP Gateway

You can turn on internal tracing for SOAP Gateway to help diagnose problems. The trace level can be changed to control the amount of logging.

The trace level determines what type of log entries are captured in the SOAP Gateway server log. To set the trace level for SOAP Gateway:

1. Determine what level of internal logging is required. SOAP Gateway performs best when logging is low or disabled, but troubleshooting may require more logging.

Table 34. Trace level settings for the SOAP Gateway server log file

Trace level value	Trace level	Description
0	z/OS Off Linux Windows This trace level is not valid	• z/OS The SOAP Gateway server log file and console appenders are disabled. Messages are not written to the SOAP Gateway server log file or job log. Error and fatal messages are still sent to WTO. • Linux Windows A command to set this trace level results in an IOGD0650W warning message. The command is ignored.
1	Fatal	Only errors that result in an immediate server shutdown are logged.

Table 34. Trace level settings for the SOAP Gateway server log file (continued)

Trace level value	Trace level	Description
2	Error	• In addition to logging errors that result in an immediate server shutdown, other errors and exceptions are also logged. • NACK responses for synchronous callout applications that use the send-only with acknowledgement protocol are logged. • This is the default trace level.
3	Warn	In addition to what is logged at the error level, warning messages are also logged.
4	Information	• In addition to what is logged at the warning level, the entry and exit of important events and functions are also logged. • All ACK and NACK messages for synchronous callout applications that use the send-only with acknowledgement protocol are logged.
5	Debug	In addition to what is logged at the information level, the contents of buffers sent to and received from IMS Connect and SOAP Gateway are also logged.

2. Issue the command `iogmgmt -prop -u -l level` where *level* is the desired trace level. For example, issuing the command `iogmgmt -prop -u -l 5` captures all available error, tracing, and debugging information.

Related reference:

"-prop: Set SOAP Gateway properties" on page 450
Use the -prop command to modify the SOAP Gateway server properties.

Changing the server log file encoding for z/OS

z/OS

The SOAP Gateway server log file in IMS Enterprise Suite is stored in ASCII encoding. To view the log file on z/OS system, you must change the encoding from ASCII to EBCDIC.

To change the encoding:

1. Locate the *-PathPrefix-*`/usr/lpp/ims/imses/V3R1/soap_gateway/imsbase/conf/native.env.properties` file in your installation directory.
2. Change the property SG_LOG4j_LOGFILE_ENCODING value to IBM-1047:
   `SG_LOG4j_LOGFILE_ENCODING=IBM-1047`
3. Restart the SOAP Gateway server.

Changing the server log file location

You can change the location of the SOAP Gateway server log file by using the SOAP Gateway management utility.

By default, SOAP Gateway logs messages in the `imssoap.log` file. The log file location is *install_dir*`/imsbase/logs`.

Tips:

- On z/OS systems, error messages are always logged to the z/OS syslog when the trace level is set to anything other than fatal (trace level 1).
- You could disable the `imssoap.log` file if you do not want a separate SOAP Gateway log file. When the log file is disabled, fatal messages are sent to the spool.

To change the log file location for SOAP Gateway:

1. Use the SOAP Gateway management utility **-prop -u** command to change a server property, and the **-f** parameter to specify the file location. Specify an absolute path to the log. The directory must already exist.

   ```
   iogmgmt -prop -u -f c:\mylogs\soap
   ```

2. Stop the SOAP Gateway server and then restart it for the change to take effect.

Related tasks:

"Disabling the server log file"
For z/OS systems, SOAP Gateway error messages are logged to the z/OS syslog by default. If you intend to log only error messages, and prefer using only the z/OS syslog, you can disable the separate SOAP Gateway server log file.

Related reference:

"-prop: Set SOAP Gateway properties" on page 450
Use the -prop command to modify the SOAP Gateway server properties.

Removing old server log files

z/OS Linux

You might want to remove or archive old server log files to save disk space. A sample log removal script is provided for z/OS and Linux on System z that demonstrates how to remove log files that are older than a specified number of days.

This `soapLogDelete.sh` sample script is located in the *install_dir*`/imsserver/tools` directory and is provided as is without support. Modify this sample script to remove old SOAP Gateway logs based on two variables:

- LOG_DIRECTORY: specifies where the SOAP Gateway log directory is
- DAYS: specifies the number of days since the log file was last modified

This script parses the log files in the LOG_DIRECTORY directory, and files older than the specified number of DAYS are forcefully removed.

The script also provides instructions on how to configure the cron daemon to execute the script. You can set up the cron daemon to automatically run in the background at regular intervals.

You can also modify the script to copy or move the log files to a different location before removing them.

Disabling the server log file

For z/OS systems, SOAP Gateway error messages are logged to the z/OS syslog by default. If you intend to log only error messages, and prefer using only the z/OS syslog, you can disable the separate SOAP Gateway server log file.

When logging to the `imssoap.log` file is disabled, all error and fatal (unrecoverable) messages are sent to the z/OS syslog. If a message is issued but the SOAP

Gateway server is not yet up and running, the message is sent to the spool. Spooling is a process that facilitates simultaneous processing by providing a temporary storage area for work that is not yet completed.

Important: Do not disable the log file unless you are using the z/OS syslog.

To disable the log file:

1. Use the SOAP Gateway management utility **-prop** parameter to change a server property, and the **-l** parameter to set the trace level to 0.

   ```
   iogmgmt.sh -prop -u -l 0
   ```

2. Restart the server.

Logging to the `imssoap.log` file is disabled.

Related tasks:

"Changing the server log file location" on page 305
You can change the location of the SOAP Gateway server log file by using the SOAP Gateway management utility.

Transaction log format

The SOAP Gateway transaction logger creates JSON files with a record for each message processing event.

However, the events that are associated with a specific SOAP request message might not be grouped together in the log file. Use the value of the linkID field in the verticalID element of each message processing event to uniquely correlate every event that is related to a specific SOAP request message. The vertical link ID for a request is always unique, and the value is included in every event log entry.

The event type formats are presented here in the order that they appear for a normal web service request or callout request, followed by the response from the IMS application (provider) or the external web service (callout). Not all of the events occur for a request that results in an error.

Related concepts:

"SOAP Gateway message processing events" on page 341
Each request message sent to a SOAP Gateway web service provider (and the associated response message) generates several uniquely identified message processing events. Likewise, each callout request generates several message processing events, from receiving the IMS callout request to receiving the response from the external web service and sending the response back to IMS.

"IDs for transaction correlation" on page 343
SOAP Gateway uses two different types of IDs to help track and correlate SOAP requests and responses: vertical IDs and horizontal IDs.

Provider request transaction log format by event type

Use the value of the linkID field in the verticalID element of each message processing event to uniquely correlate every event related to a specific SOAP request message. The vertical link ID for a request is always unique, and the value is included in every event log entry.

The event type formats are presented here in the order that they appear for a normal SOAP request from a client application, followed by the response from the IMS application. Not all of the events occur for a request that results in an error.

Related concepts:

"SOAP Gateway message processing events" on page 341
Each request message sent to a SOAP Gateway web service provider (and the associated response message) generates several uniquely identified message processing events. Likewise, each callout request generates several message processing events, from receiving the IMS callout request to receiving the response from the external web service and sending the response back to IMS.

Transaction log for provider event type 0:

Provider event type 0 indicates that SOAP request was received by SOAP Gateway.

Type Identifies the type of event.

Timestamp
The time that the event was processed. This value is the fixed-point number of fractional seconds, measured in milliseconds, between the current time and midnight, January 1, 1970 UTC.

VerticalContext
The VerticalContext element contains the following properties:

ComponentName
IMS Enterprise Suite SOAP Gateway

ApplicationName
The host name and port number of the SOAP Gateway server.

ServerName
The host name of the SOAP Gateway server.

VerticalID
The VerticalID element contains the following properties:

CallerID
CallerID 24 is SOAP Gateway.

LinkID
A unique identifier that links all transaction tracking events generated by SOAP Gateway for a specific request. This ID is not propagated to IMS Connect.

InstanceID
The InstanceID element contains the following properties:

TransactionData
The TransactionData element contains the following properties:

messageID
The value of the messageID element in the SOAP request message.

hlink The horizontal tracking ID that is propagated to IMS Connect. You can configure how this value is generated with the iogmgmt -tracking command.

hlinkHex
The hexadecimal equivalent of the encoded hlink value.

The following example shows the JSON format for this event type:

```
{"event":
 {"instanceId":
  {"transactiondata":
   {
    "messageID":"urn:uuid:DCA3E8B163DADDA0EB1309983896543",
```

```
    "hlink":"urn:uuid:DCA3E8B163DADDA0EB1309983896543",
    "hlinkHex":"75726e3a757569643a4443413345384231363344414444413045423133303939383333839363533433"
   }
  },
 "verticalId":
  {
   "linkId":"6997c28c7d0456fc02ac84d600b232858fe335ff071fb55b",
   "caller":24
  },
 "timestamp":1340917997.997000,
 "verticalContext":
  {
   "ComponentName":"IMS Enterprise Suite SOAP Gateway",
   "ApplicationName":"TestA:8080",
   "ServerName":"TestA"
  },
 "type":0
 }
}
```

Transaction log for provider event type 10:

Provider event type 10 indicates that SOAP Gateway received the callout request.

Type Identifies the type of event.

Timestamp
The time that the event was processed. This value is the fixed-point number of fractional seconds, measured in milliseconds, between the current time and midnight, January 1, 1970 UTC.

VerticalContext
The VerticalContext element contains the following properties:

TransactionName
The fully qualified operation name specified by the request. The format is: {*nameSpace*}*serviceName*:*operationName*

VerticalID
The VerticalID element contains the following properties:

CallerID
CallerID 24 is SOAP Gateway.

LinkID
A unique identifier that links all transaction tracking events generated by SOAP Gateway for a specific request. This ID is not propagated to IMS Connect.

InstanceID
The InstanceID element contains the following properties:

TransactionData
The TransactionData element contains the following properties:

messageID
The value of the messageID element in the SOAP request message.

hlink The horizontal tracking ID that is propagated to IMS Connect. You can configure how this value is generated with the iogmgmt -tracking command.

hlinkHex
The hexadecimal equivalent of the encoded hlink value.

The following example shows the JSON format for this event type:

```
{"event":
 {"instanceId":
  {"transactiondata":
   {
    "messageID":"urn:uuid:DCA3E8B163DADDA0EB1309983896543",
    "hlink":"urn:uuid:DCA3E8B163DADDA0EB1309983896543",
    "hlinkHex":"75726e3a757569643a444341334538423136334441444441304542313330393938333839363534 33"
   }
 },
 "verticalId":
  {
   "linkId":"6997c28c7d0456fc02ac84d600b232858fe335ff071fb55b",
   "caller":24
  },
 "timestamp":1340917997.997000,
 "verticalContext":
  {
   "TransactionName":"{file:\/\/target.files1281569418803}IMSPHBKService:IMSPHBKOperation"
  },
 "type":10
 }
}
```

Transaction log for provider event type 12:

Provider event type 12 indicates that SOAP Gateway sent callout request to the external web service.

Type Identifies the type of event.

Timestamp
The time that the event was processed. This value is the fixed-point number of fractional seconds, measured in milliseconds, between the current time and midnight, January 1, 1970 UTC.

HorizontalContext
The HorizontalContext element contains the following properties:

TranCode
The IMS trancode specified in the request message.

ApplicationName
The host name, port number, and data store name of the target IMS Connect.

ComponentName
The type of destination.

HorizontalID
The HorizontalID element contains the following properties:

CallerID
CallerID 13 is IMS Connect.

LinkID
The horizontal tracking ID that is propagated to IMS Connect. You can configure how this value is generated with the iogmgmt -tracking command.

InstanceID
The InstanceID element contains the following properties:

TransactionData
The TransactionData element contains the following properties:

messageID
The value of the messageID element in the SOAP request message.

hlinkHex
The hexadecimal equivalent of the encoded HorizontalID LinkID value.

UserID
The user ID specified in the request message.

MessgeLen
The length of the request message in bytes.

The following example shows the JSON format for this event type:

```
{"event":
 {"instanceId":
  {"transactiondata":
   {
    "messageID":"urn:uuid:DCA3E8B163DADDA0EB1309983896543",
    "UserID":"UserA",
    "hlinkHex":"75726e3a757569643a444341334538423136334441444441304542313330393938333839363534333",
    "MessageLen":"953"
   }
  },
 "verticalId":
  {
   "linkId":"6997c28c7d0456fc02ac84d600b232858fe335ff071fb55b",
   "caller":24
  },
 "timestamp":1340917997.997000,
 "type":12,
 "horizontalContext":
  {
   "ComponentName":"IMS Connect",
   "TransactionName":"IVTNO",
   "ApplicationName":"ec03581.vmec.svl.ibm.com:9999:IMS1",
   "ServerName":"ec03581.vmec.svl.ibm.com"
  },
 "horizontalId":
  {
   "linkId":"urn:uuid:DCA3E8B163DADDA0EB1309983896543",
   "caller":13
  }
 }
}
```

Transaction log for provider event type 11:

Provider event type 11 indicates that SOAP Gateway received response message from the external web service.

Type Identifies the type of event.

Timestamp
The time that the event was processed. This value is the fixed-point number of fractional seconds, measured in milliseconds, between the current time and midnight, January 1, 1970 UTC.

HorizontalContext
The HorizontalContext element contains the following properties:

TranCode
The IMS trancode specified in the request message.

ApplicationName
The host name, port number, and data store name of the target IMS Connect.

ComponentName
The type of destination.

HorizontalID

The HorizontalID element contains the following properties:

CallerID

CallerID 13 is IMS Connect.

LinkID

The horizontal tracking ID that is propagated to IMS Connect. You can configure how this value is generated with the iogmgmt -tracking command.

VerticalID

The VerticalID element contains the following properties:

CallerID

CallerID 24 is SOAP Gateway.

LinkID

A unique identifier that links all transaction tracking events generated by SOAP Gateway for a specific request. This ID is not propagated to IMS Connect.

InstanceID

The InstanceID element contains the following properties:

TransactionData

The TransactionData element contains the following properties:

messageID

The value of the messageID element in the SOAP request message.

hlinkHex

The hexadecimal equivalent of the encoded HorizontalID LinkID value.

MessgeLen

The length of the request message in bytes.

FaultString

If an error occurred, this element contains error information from IMS Connect, IMS, or SOAP Gateway.

The following example shows the JSON format for this event type:

```
{"event":
 {"instanceId":
  {"transactiondata":
   {
    "messageID":"urn:uuid:DCA3E8B163DADDA0EB1309983896543",
    "hlinkHex":"75726e3a757569643a444341334538423136334441444441304542313330393938333839363534 33",
    "MessageLen":"430"
   }
  },
 "verticalId":
 {
  "linkId":"6997c28c7d0456fc02ac84d600b232858fe335ff071fb55b",
  "caller":24
 },
 "timestamp":1340917997.997000,
 "type":11,
 "horizontalContext":
  {
   "ComponentName":"IMS Connect",
   "ApplicationName":"ec03581.vmec.svl.ibm.com:9999:IMS1",
   "ServerName":"ec03581.vmec.svl.ibm.com"
  },
 "horizontalId":
  {
   "linkId":"urn:uuid:DCA3E8B163DADDA0EB1309983896543",
```

```
   "caller":13
  }
 }
}
```

Transaction log for provider event type 17:

Provider event type 17 indicates that SOAP Gateway sent the response to IMS Connect.

Type Identifies the type of event.

Timestamp
The time that the event was processed. This value is the fixed-point number of fractional seconds, measured in milliseconds, between the current time and midnight, January 1, 1970 UTC.

VerticalID
The VerticalID element contains the following properties:

CallerID
CallerID 24 is SOAP Gateway.

LinkID
A unique identifier that links all transaction tracking events generated by SOAP Gateway for a specific request. This ID is not propagated to IMS Connect.

InstanceID
The InstanceID element contains the following properties:

TransactionData
The TransactionData element contains the following properties:

messageID
The value of the messageID element in the SOAP request message.

hlink The horizontal tracking ID that is propagated to IMS Connect. You can configure how this value is generated with the iogmgmt -tracking command.

hlinkHex
The hexadecimal equivalent of the encoded hlink value.

MessgeLen
The length of the request message in bytes.

FaultString
If an error occurred, this element contains error information from IMS Connect, IMS, or SOAP Gateway.

The following example shows the JSON format for this event type:

```
{"event":
 {"instanceId":
  {"transactiondata":
   {
    "messageID":"urn:uuid:DCA3E8B163DADDA0EB1309983896543",
    "hlink":"urn:uuid:DCA3E8B163DADDA0EB1309983896543",
    "hlinkHex":"75726e3a757569643a444341334538423136334441444441304542313330393938333839363534333"
   }
  },
 "verticalId":
  {
   "linkId":"6997c28c7d0456fc02ac84d600b232858fe335ff071fb55b",
   "caller":24
  },
```

```
 "timestamp":1340917997.997000,
 "type":17
 }
}
```

Callout request transaction log format by event type

The SOAP Gateway transaction logger creates JSON files with a record for each callout message processing event.

The record includes multiple entries, each representing an event type, from a resume tpipe call to IMS Connect, to sending the web service response back to IMS Connect.

Use the value of the linkID field in the verticalID element of each message processing event to uniquely correlate every event related to a specific callout request message. The vertical link ID for a request is always unique, and the value is included in every event log entry.

The events that are logged include the following event types:

- Event type 0 indicates a resume tpipe call was issued.
- For a callout request with the send-only-with-ack flag set, the pattern of 10, 8, 12, 11, 13, 8 is repeated for each callout request if the thread policy is set to one thread per tpipe. The pattern of 0, 10, 8, 12, 11, 13, 8 is repeated for each callout request if the thread policy is set to one thread per connection bundle.
- For a callout request without the send-only-with-ack flag set, the pattern of 10, 8, 12, 11, 17 is repeated for each callout request if the thread policy is set to one thread per tpipe. The pattern of 0, 10, 8, 12, 11, 17 is repeated for each callout request if the thread policy is set to one thread per connection bundle.

The event type formats are listed in the order that they appear for a normal SOAP callout request, followed by the response from the external web service.

Not all of the events occur for a request if the request results in an error. For non-response mode callout request, event type 17 does not occur.

If an error is encountered during the callout processing (request or response), the details of the error is included in the transaction log. The following is a sample transaction log for event type 0 when an error occurred, with an IOGC001E error reported.

```
{"event":
    {"timestamp":1402523998.984000,
     "verticalId":
        {"caller":24,
         "linkId":"1402523998984%%%cb_callout%%%SGPSING1"},
    "instanceId":
        {"transactiondata":
            {"Scenario":"CONSUMER",
             "MessageLen":"0",
             "Tpipe":"SGPSING1",
             "UserID":null,
             "Error":"IOGC001E: Connection to IMS Connect failed. Hostname
             [ec32629.my.server.com], port [9999]. [Connection refused: connect]"}
        },
    "type":0,
    "verticalContext":
        {"ApplicationName":"ec32629.my.server.com:9999:IMS1",
         "ServerName":"ec32629.my.server.com",
         "ComponentName":"IMS Enterprise Suite SOAP Gateway"}
    }
}
```

Related concepts:

"SOAP Gateway message processing events" on page 341
Each request message sent to a SOAP Gateway web service provider (and the associated response message) generates several uniquely identified message processing events. Likewise, each callout request generates several message processing events, from receiving the IMS callout request to receiving the response from the external web service and sending the response back to IMS.

Transaction log for callout event type 0:

Callout event type 0 indicates that SOAP Gateway sent IMS Connect a resume tpipe call to retrieve callout messages.

Type Identifies the type of event.

Timestamp
The time that the event was processed. This value is the fixed-point number of fractional seconds, measured in milliseconds, between the current time and midnight, January 1, 1970 UTC.

VerticalID
The VerticalID element contains the following properties:

Caller Caller 24 is SOAP Gateway.

LinkID
A unique identifier that links all transaction tracking events generated by SOAP Gateway for a specific callout request. A linkID has two parts. The first part is a generated ID followed by the client ID, connection bundle name, and tpipe name. The second part is a generated ID that identifies a set of messages that are related to a particular callout request. The LinkID is not propagated to IMS Connect.

InstanceID
The InstanceID element contains the following properties:

TransactionData
The TransactionData element contains the following properties:

Scenario
The value is always CONSUMER.

MessageLen
The length of the message.

Tpipe The tpipe name.

UserID
The RACF user ID that is specified in the connection bundle for connection with IMS for the resume tpipe call.

VerticalContext
The VerticalContext element contains the following properties:

ApplicationName
The host name and port number of the host system on which IMS Connect is running.

ServerName
The host name of the host system on which IMS Connect is running.

ComponentName
IMS Enterprise Suite SOAP Gateway

The following example shows the JSON format for this event type:

```
{"event":
    {"timestamp":1402520468.547000,
     "verticalId":
        {"caller":24,"linkId":"1402520468547IOGW6SOW"},
     "instanceId":
        {"transactiondata":
            {"Scenario":"CONSUMER","MessageLen":"128","Tpipe":"SGPIPE02","UserID":""}
        },
    "type":0,
    "verticalContext":
        {"ApplicationName":"csdmec04.vmec.mycom.com:9999:IMS1",
         "ServerName":"csdmec04.vmec.mycom.com",
         "ComponentName":"IMS Enterprise Suite SOAP Gateway"
        }
    }
}
```

Transaction log for callout event type 10:

Callout event type 10 indicates that SOAP Gateway received the callout request from IMS.

Type Identifies the type of event.

Timestamp
The time that the event was processed. This value is the fixed-point number of fractional seconds, measured in milliseconds, between the current time and midnight, January 1, 1970 UTC.

VerticalID
The VerticalID element contains the following properties:

Caller Caller 24 is SOAP Gateway.

LinkID
A unique identifier that links all transaction tracking events generated by SOAP Gateway for a specific callout request. A linkID has two parts. The first part is a generated ID followed by the client ID, connection bundle name, and tpipe name. The second part is a generated ID that identifies a set of messages that are related to a particular callout request. The LinkID is not propagated to IMS Connect.

InstanceID
The InstanceID element contains the following properties:

TransactionData
The TransactionData element contains the following properties:

Service Name
The name of the web service to invoke.

WSID Web service identifier.

Scenario
The scenario is always CONSUMER.

Namespace
The target namespace of the web services. A namespace is a unique identifier of the elements and attributes in the form of a Uniform Resource Identifier or web address.

Port Name
The port name of the operation to invoke.

Operation Name
The operation name of the web service to invoke.

VerticalContext
The VerticalContext element contains the following properties:

ApplicationName
The host name and port number of the SOAP Gateway server.

ServerName
The host name of the SOAP Gateway server.

ComponentName
The component in this context, which is IMS Enterprise Suite SOAP Gateway.

The following example shows the JSON format for this event type:

```
{"event":
    {"timestamp":1402521954.570000,
     "verticalId":
        {"caller":24,
         "linkId":"1402520468547IOGW6SOW%%%sync_connbundle%%%SGPIPE02%%%1402521954570"
        },
    "instanceId":
        {"transactiondata":
           {"Service Name":"HELLOService",
            "WSID":"HELLO",
            "Scenario":"CONSUMER",
            "Namespace":"http:\/\/www.example.org\/HELLO\/",
            "Port Name":"HELLOPort",
            "Operation Name":"HelloOperation"
           }
        },
    "type":10,
    "verticalContext":
        {"ApplicationName":"IBM-HIMAKAR:8089",
         "ServerName":"IBM-HIMAKAR",
         "ComponentName":"IMS Enterprise Suite SOAP Gateway"
        }
    }
}
```

Transaction log for callout event type 8:

Callout event type 8 indicates that SOAP Gateway notified IMS Connect that the callout request is received and committed for processing.

Type Identifies the type of event.

Timestamp
The time that the event was processed. This value is the fixed-point number of fractional seconds, measured in milliseconds, between the current time and midnight, January 1, 1970 UTC.

VerticalID
The VerticalID element contains the following properties:

Caller Caller 24 is SOAP Gateway.

LinkID
A unique identifier that links all transaction tracking events generated by SOAP Gateway for a specific callout request. A linkID has two parts. The first part is a generated ID followed by the client ID, connection bundle name, and tpipe name. The second

part is a generated ID that identifies a set of messages that are related to a particular callout request. The LinkID is not propagated to IMS Connect.

InstanceID
The InstanceID element contains the following properties:

TransactionData
The TransactionData element contains the following properties:

Scenario
The value is always CONSUMER.

Acknowledgment for Send Only with Ack
This entry is included if the callout web service is deployed with the send-only-with-ack protocol flag turned on.

Operation Name
The operation name of the web service to invoke.

VerticalContext
The VerticalContext element contains the following properties:

ApplicationName
The host name and port number of the SOAP Gateway server.

ServerName
The host name of the SOAP Gateway server.

ComponentName
The component in this context, which is IMS Enterprise Suite SOAP Gateway.

The following example shows the JSON format for this event type, when the send-only-with-ack flag is not set.

```
{"event":
    {"timestamp":1402522397.909000,
     "verticalId":
        {"caller":24,
         "linkId":"1402522303207IOGPRF6U%%%sync_connbundle%%%SGPIPE02%%%1402522397646"},
     "instanceId":
        {"transactiondata":
            {"Scenario":"CONSUMER"}
        },
     "type":8,
     "verticalContext":
        {"ApplicationName":"IBM-HIMAKAR:8089",
         "ServerName":"IBM-HIMAKAR",
         "ComponentName":"IMS Enterprise Suite SOAP Gateway"}
    }
}
```

The following example shows the JSON format for this event type, when the send-only-with-acknowledgement flag is set.

```
{"event":
    {"timestamp":1402690207.999000,
     "verticalId":
        {"caller":24,
       "linkId":"1402690091247IOG5F9KQ%%%sync_connbundle%%%SGPIPE02%%%1402690207013"},
     "instanceId":
      {"transactiondata":
         {"Service Name":"HELLOService",
          "Scenario":"CONSUMER",
          "Acknowledgment for Send Only with Ack":"",
          "Operation Name":"HelloOperation"}
      },
    "type":8,
    "verticalContext":
```

```
      {"ApplicationName":"IBM-HIMAKAR:8089",
       "ServerName":"IBM-HIMAKAR",
       "ComponentName":"IMS Enterprise Suite SOAP Gateway"}
    }
}
```

Transaction log for callout event type 9:

Callout event type 9 indicates that SOAP Gateway could not commit to processing the callout request or an issue with receiving an acknowledge of the response message has occurred.

Type Identifies the type of event.

Timestamp
The time that the event was processed. This value is the fixed-point number of fractional seconds, measured in milliseconds, between the current time and midnight, January 1, 1970 UTC.

VerticalID
The VerticalID element contains the following properties:

Caller Caller 24 is SOAP Gateway.

LinkID
A unique identifier that links all transaction tracking events generated by SOAP Gateway for a specific callout request. A linkID has two parts. The first part is a generated ID followed by the client ID, connection bundle name, and tpipe name. The second part is a generated ID that identifies a set of messages that are related to a particular callout request. The LinkID is not propagated to IMS Connect.

InstanceID
The InstanceID element contains the following properties:

TransactionData
The TransactionData element contains the following properties:

Scenario
The value is always CONSUMER.

Acknowledgment for Send Only with Ack
This entry is included if the callout web service is deployed with the send-only-with-acknowledge protocol flag turned on.

Operation Name
The operation name of the web service to invoke.

VerticalContext
The VerticalContext element contains the following properties:

ApplicationName
The host name and port number of the SOAP Gateway server.

ServerName
The host name of the SOAP Gateway server.

ComponentName
The component in this context, which is IMS Enterprise Suite SOAP Gateway.

The following example shows the JSON format for this event type, when the send-only-with-acknowledgement flag is set and SOAP Gateway received a negative acknowledgement (NACK) from IMS Connect.

```
{"event":
    {"timestamp":1408139292.858000,
     "verticalId":
        {"caller":24,
         "linkId":"1408139236008IOGM7FU5%%%sync_connbundle%%%SGPIPE02%%%1408139268410"},
     "instanceId":
        {"transactiondata":
            {"Service Name":"HELLOService",
             "Scenario":"CONSUMER",
             "Negative acknowledgment for Send Only with Ack":"",
             "Operation Name":"HelloOperation",
             "Error":"IOGS0082E: SOAP Gateway received a negative acknowledgement (NACK)
              from IMS for the following callout application: service HELLOService,
              operation HelloOperation, and target namespace
              http:\/\/www.example.org\/HELLO\/.  The reason for the NACK response
              was: IOGC041E: IMS Connect returns an error. Return code: [16]. Reason
              code:[1009]. [HWS_RETCODE_16: OTMA error encountered.  Check the OTMA
              sense code and take appropriate action.]."}
        },
     "type":9,
     "verticalContext":
        {"ApplicationName":"IBM-HIMAKAR:8089",
         "ServerName":"IBM-HIMAKAR",
         "ComponentName":"IMS Enterprise Suite SOAP Gateway"}
    }
}
```

Transaction log for callout event type 12:

Callout event type 12 indicates that SOAP Gateway sent the callout request message to the external web service.

Type Identifies the type of event.

Timestamp
The time that the event was processed. This value is the fixed-point number of fractional seconds, measured in milliseconds, between the current time and midnight, January 1, 1970 UTC.

VerticalID
The VerticalID element contains the following properties:

Caller Caller 24 is SOAP Gateway.

LinkID
A unique identifier that links all transaction tracking events generated by SOAP Gateway for a specific callout request. A linkID has two parts. The first part is a generated ID followed by the client ID, connection bundle name, and tpipe name. The second part is a generated ID that identifies a set of messages that are related to a particular callout request. The LinkID is not propagated to IMS Connect.

InstanceID
The InstanceID element contains the following properties:

TransactionData
The TransactionData element contains the following properties:

Service Name
The name of the web service to invoke.

Target Endpoint
The address of the external web service.

Scenario
The scenario is always CONSUMER.

Web Service Timeout
The amount of time in milliseconds that SOAP Gateway would wait to receive a response from the web service.

Callout Request Length
The length of the callout request message in bytes.

WSDL File Name
The name of the web service WSDL file.

Operation Name
The operation name of the web service to invoke.

Callout Security Enabled
Whether callout security is enabled.

VerticalContext
The VerticalContext element contains the following properties:

ApplicationName
The host name and port number of the SOAP Gateway server.

ServerName
The host name of the SOAP Gateway server.

ComponentName
The component in this context, which is IMS Enterprise Suite SOAP Gateway.

The following example shows the JSON format for this event type:

```
{"event":
    {"timestamp":1402522398.068000,
     "verticalId":
        {"caller":24,
         "linkId":"1402522303207IOGPRF6U%%%sync_connbundle%%%SGPIPE02%%%1402522397646"},
     "instanceId":{
        "transactiondata":
            {"Service Name":"HELLOService",
             "Target Endpoint":"Address: http:\/\/localhost:8081\/axis2\/services\/HELLOService",
             "Scenario":"CONSUMER",
             "Web Service Timeout":"7500",
             "Callout Request Length":"285",
             "WSDL File Name":"HELLO.wsdl",
             "Operation Name":"HelloOperation",
             "Callout Security Enabled":"false"}
        },
     "type":12,
     "verticalContext":
        {"ApplicationName":"IBM-HIMAKAR:8089",
         "ServerName":"IBM-HIMAKAR",
         "ComponentName":"IMS Enterprise Suite SOAP Gateway"}
    }
}
```

Transaction log for callout event type 11:

Callout event type 11 indicates that SOAP Gateway received the response from the external web service.

Type Identifies the type of event.

Timestamp
The time that the event was processed. This value is the fixed-point number of fractional seconds, measured in milliseconds, between the current time and midnight, January 1, 1970 UTC.

VerticalID

The VerticalID element contains the following properties:

Caller Caller 24 is SOAP Gateway.

LinkID

A unique identifier that links all transaction tracking events generated by SOAP Gateway for a specific callout request. A linkID has two parts. The first part is a generated ID followed by the client ID, connection bundle name, and tpipe name. The second part is a generated ID that identifies a set of messages that are related to a particular callout request. The LinkID is not propagated to IMS Connect.

InstanceID

The InstanceID element contains the following properties:

TransactionData

The TransactionData element contains the following properties:

Service Name

The name of the web service to invoke.

Callout Response Length

The length of the callout response message in bytes.

Scenario

The scenario is always CONSUMER.

Operation Name

The operation name of the web service that was invoked.

VerticalContext

The VerticalContext element contains the following properties:

ApplicationName

The host name and port number of the SOAP Gateway server.

ServerName

The host name of the SOAP Gateway server.

ComponentName

The component in this context, which is IMS Enterprise Suite SOAP Gateway.

The following example shows the JSON format for this event type:

```
{"event":
    {"timestamp":1402521957.575000,
     "verticalId":
        {"caller":24,
         "linkId":"1402520468547IOGW6SOW%%%sync_connbundle%%%SGPIPE02%%%1402521954570"},
     "instanceId":
        {"transactiondata":
            {"Service Name":"HELLOService",
             "Callout Response In Length":"242",
             "Scenario":"CONSUMER",
             "Operation Name": "HelloOperation"}
        },
     "type":11,
     "verticalContext":
        {"ApplicationName":"IBM-HIMAKAR:8089",
         "ServerName":"IBM-HIMAKAR",
         "ComponentName":"IMS Enterprise Suite SOAP Gateway"}
    }
}
```

Transaction log for callout event type 17:

Callout event type 17 indicates that SOAP Gateway sent the callout response from the external web service to IMS Connect. The transaction is complete and SOAP Gateway is not waiting for an acknowledgement from IMS Connect.

Type Identifies the type of event.

Timestamp
The time that the event was processed. This value is the fixed-point number of fractional seconds, measured in milliseconds, between the current time and midnight, January 1, 1970 UTC.

VerticalID
The VerticalID element contains the following properties:

Caller Caller 24 is SOAP Gateway.

LinkID
A unique identifier that links all transaction tracking events generated by SOAP Gateway for a specific callout request. A linkID has two parts. The first part is a generated ID followed by the client ID, connection bundle name, and tpipe name. The second part is a generated ID that identifies a set of messages that are related to a particular callout request. The LinkID is not propagated to IMS Connect.

InstanceID
The InstanceID element contains the following properties:

TransactionData
The TransactionData element contains the following properties:

Service Name
The name of the web service to invoke.

Scenario
The scenario is always CONSUMER.

Callout Finished Response Out Length
The length of the callout response message to IMS Connect in bytes.

Operation Name
The operation name of the web service that was invoked.

VerticalContext
The VerticalContext element contains the following properties:

ApplicationName
The host name and port number of the SOAP Gateway server.

ServerName
The host name of the SOAP Gateway server.

ComponentName
The component in this context, which is IMS Enterprise Suite SOAP Gateway.

The following example shows the JSON format for this event type:

```
{"event":
    {"timestamp":1402521957.949000,
     "verticalId":{
        "caller":24,
        "linkId":"1402520468547IOGW6SOW%%%sync_connbundle%%%SGPIPE02%%%1402521954570"},
```

```
     "instanceId":{
        "transactiondata":{
            "Service Name":"HELLOService",
            "Scenario":"CONSUMER",
            "Callout Finished Response Out Length":"242",
            "Operation Name":"HelloOperation"}
        },
     "type":17,
     "verticalContext":{
        "ApplicationName":"IBM-HIMAKAR:8089",
        "ServerName":"IBM-HIMAKAR",
        "ComponentName":"IMS Enterprise Suite SOAP Gateway"}
    }
}
```

Transaction log for callout event type 13:

Callout event type 13 indicates that SOAP Gateway sent the callout response from the external web service to IMS Connect. SOAP Gateway is waiting for a subsequent acknowledgement (send-only-with-acknowledgement) from IMS Connect with a callout event type 8.

Type Identifies the type of event.

Timestamp
The time that the event was processed. This value is the fixed-point number of fractional seconds, measured in milliseconds, between the current time and midnight, January 1, 1970 UTC.

VerticalID
The VerticalID element contains the following properties:

Caller Caller 24 is SOAP Gateway.

LinkID
A unique identifier that links all transaction tracking events generated by SOAP Gateway for a specific callout request. A linkID has two parts. The first part is a generated ID followed by the client ID, connection bundle name, and tpipe name. The second part is a generated ID that identifies a set of messages that are related to a particular callout request. The LinkID is not propagated to IMS Connect.

InstanceID
The InstanceID element contains the following properties:

TransactionData
The TransactionData element contains the following properties:

Service Name
The name of the web service to invoke.

Scenario
The scenario is always CONSUMER.

Callout Response Out Length
The length of the callout response message to IMS Connect in bytes.

Operation Name
The operation name of the web service that was invoked.

VerticalContext
The VerticalContext element contains the following properties:

ApplicationName
The host name and port number of the SOAP Gateway server.

ServerName
The host name of the SOAP Gateway server.

ComponentName
The component in this context, which is IMS Enterprise Suite SOAP Gateway.

The following example shows the JSON format for this event type:

```
{"event":
    {"timestamp":1402523996.115000,
     "verticalId":{
        "caller":24,
        "linkId":"1402522303207IOGPRF6U%%%sync_connbundle%%%SGPIPE02%%%1402523995690"},
     "instanceId":{
        "transactiondata":{
            "Service Name":"HELLOService",
            "Scenario":"CONSUMER",
            "Callout Response Out Length":"242",
            "Operation Name":"HelloOperation"}
        },
     "type":13,
     "verticalContext":{
        "ApplicationName":"IBM-HIMAKAR:8089",
        "ServerName":"IBM-HIMAKAR",
        "ComponentName":"IMS Enterprise Suite SOAP Gateway"}
    }
}
```

Configuring the transaction log

The SOAP Gateway transaction log records information about every inbound and outbound request and the associated response message to and from IMS.

When transaction logging is enabled, SOAP Gateway creates a record for every inbound and outbound web service request and their related response messages that pass through the SOAP Gateway server.

Restriction: The transaction log does not record information for WebSphere Business Events emitters.

Recommendation: For the provider scenario, enable horizontal message tracking ID generation with the iogmgmt -tracking -on command before you activate the transaction logger. You can use the message IDs to correlate incoming request messages processing events with the following response message processing events in the transaction log.

This information can also be sent to a remote transaction collector with the IBM Tivoli Composite Application Manager for Transactions (ITCAM) Transaction Tracking API (TTAPI). See “Configuring the IBM Tivoli Composite Application Manager for Transactions (ITCAM) Transaction Tracking API (TTAPI)” on page 346.

To enable the transaction log:

1. Determine where to store the transaction log files. By default, the transaction log files are stored in the same directory as the server log file. However, you might want to make the transaction log files accessible to users who are not interested in or authorized to access the SOAP Gateway server log file. For example, the information in the transaction logs can be useful for IMS Connect and IMS system programmers and application developers who are trying to locate a bottleneck in transaction processing. The transaction logs can also be used to audit IMS transactions that are invoked by SOAP Gateway.

2. Create a space management plan for the transaction log directory. For the provider scenario, each transaction record uses approximately 300 bytes of disk space, and each successful transaction generates five transaction records in the active log file. For the callout scenario, the record size is larger because more transaction records are generated for each transaction. Use this information and your estimated or actual transaction workload statistics to determine how quickly the size of the transaction log files might grow in your production environment. The **-fileMaxSize** and **-fileMaxAge** parameters of the iogmgmt -tranLog command provide two options that you can use to control when the logger creates a new log file. You can create a scheduled task or cron job to move or delete old transaction log files at regular intervals.
3. Activate the transaction logger with the `iogmgmt -tranLog -on` command.

Related concepts:

"IDs for transaction correlation" on page 343
SOAP Gateway uses two different types of IDs to help track and correlate SOAP requests and responses: vertical IDs and horizontal IDs.

Related reference:

"-tranLog: Configure the SOAP Gateway transaction logger" on page 460
Use the -tranLog command to enable or disable the SOAP Gateway transaction logger for both web service provider and callout transactions.

Administrative tasks for SOAP Gateway web services

Use the SOAP Gateway management utility to help you to set up properties and create runtime code that SOAP Gateway uses to enable IMS applications as web services.

Setting up the connections and correlator file properties for web services

You must set up the connection bundle entry and correlator entry properties to ensure that incoming web service requests are correctly matched to the target IMS application. To configure these properties, use the SOAP Gateway management utility.

Connection bundle entries contain the connection and security properties that SOAP Gateway uses to communicate with IMS Connect and IMS. Define at least one connection bundle entry for each IMS system that SOAP Gateway communicates with. Any number of web services can share one connection bundle entry if they communicate with applications that run in the same IMS instance.

A correlator file specifies the transaction, runtime, and correlation properties that SOAP Gateway uses to match incoming web service requests to the appropriate back end IMS application. The unique identifier for a web service in SOAP Gateway is the combination of its service name and operation name in the correlator entry. Each unique web service must have a unique correlator entry.

1. Issue the SOAP Gateway management utility command `iogmgmt -conn -c -h` *`host_name`* `-d` *`datastore_name`* to create a connection bundle for the web service. A connection bundle entry for a web service provider must contain both the host name of the target IMS system and the name of the target datastore. You can also specify other connection and security properties in a connection bundle, such as a different port number for the web service.
2. Issue the SOAP Gateway management utility command `iogmgmt -corr -c -w` *`wsdl_file`* `-p` *`service_name`* `-i` *`operation_name`* `-n` *`connection_bundle_name`* to

create a correlator entry for the web service. You can also specify other interaction properties with the command.

3. Optional: If you modified the name of a connection bundle entry, issue the SOAP Gateway management utility iogmgmt -corr -u command for each correlator entry that uses the modified connection bundle entry to update the name. Correlator entries that continue to reference the obsolete connection bundle entry name can produce unexpected server behavior or runtime errors.

Deploying a web service to SOAP Gateway

A web service must be deployed to the SOAP Gateway server before it is available to client applications.

A web service in the provider scenario in SOAP Gateway has three components:

- A WSDL file that describes the web service to client applications. The WSDL file can import additional XSD files.
- An entry in a correlator XML file that defines the transaction properties between web service requests and an IMS application.
- A connection bundle entry that defines connection and security properties between SOAP Gateway and IMS Connect.

Deploying a web service makes these artifacts active in the runtime configuration. The service WSDL file and any imported XSD files are bundled into the web service Axis ARchive file (AAR file), stored in the master configuration, and made available in the runtime configuration. The web service is then available for client applications.

Tip: z/OS For z/OS systems, the correlator file and the WSDL file, when uploaded from a distributed platform, must be transferred in BINARY mode from your local workstation. Binary transfers provide a bit-by-bit copy that preserves the encoding on your system by instructing the FTP socket not to convert the encoding to the local system encoding (EBCDIC).

1. Deploy the web service with the `iogmgmt -deploy -w` *`wsdl_file`* `-r` *`correlator_file`* command. The WSDL file can be either the file name, if the WSDL file is already in the server WSDL directory, or the fully qualified path to the WSDL file. The correlator file can be either the file name, if the correlator XML file is already in the server XML directory, or the fully qualified path to the correlator XML file.

   Tip: For ease of maintenance and support, it is recommended that you store your web service WSDL, XSD, and correlator files in a location other than the SOAP Gateway WSDL and XML directories. Deploy a web service by specifying the fully qualified path to these files.
2. Verify that the web service is deployed. The SOAP Gateway management utility issues messages if it is unable to deploy the web service. You can verify that the service is deployed by issuing the `iogmgmt -view -correlatorfile ALL` command. Alternatively, use the SOAP Gateway administrative console. The correlator file for the newly deployed web service is displayed in the list of active correlator files in the runtime cache.

The web service is now deployed and can accept client application requests.

Related concepts:

"SOAP Gateway master configuration and runtime configuration" on page 25
The master configuration is the authoritative configuration of the SOAP Gateway server, and is stored in the file system. The active server configuration in memory is the runtime configuration.

Related reference:

"-deploy: Deploy a web service or callout application" on page 444
The -deploy command deploys a web service, callout application, or business event application to the active configuration of the SOAP Gateway server.

Changing deployed web services

After you deploy a web service, you can change interaction and connection properties with the SOAP Gateway management utility.

To protect the integrity of active web services and callout applications, some changes are made only to the master configuration of the server. The changes are automatically propagated from the master to the configuration to the runtime configuration the next time that the server starts. The runtime configuration does not immediately reflect changes related to the following tasks:

- Changes to an active connection bundle entry. A connection bundle entry is active if it is loaded in the runtime configuration, regardless of whether it is used by a web service or callout application.
- Change the connection bundle entry name or callout connection bundle entry name specified in an active correlator entry.

WSDL files can be updated with Rational Developer for System z or manually. The web service must be undeployed and then redeployed for the changes to take effect.

1. Issue the SOAP Gateway management utility command to update either a correlator entry or connection bundle entry.
   - For a correlator entry update, issue the `iogmgmt -corr -u` command with additional parameters to specify the updated properties.
   - For a connection bundle entry update, issue the `iogmgmt -conn -u` command with additional parameters to specify the updated properties.
2. Verify that the command was successful. The SOAP Gateway management utility generates an informational message to verify successful commands. Failed commands generate error or warning messages. Messages from the utility are prefixed with `IOGD`. You can also directly verify that the update is in effect with the `iogmgmt -view` command.

Related reference:

"-corr: Create or update a correlator entry" on page 439
Use the -corr command to create or update the transaction and runtime properties of a correlator entry.

"-prop: Set SOAP Gateway properties" on page 450
Use the -prop command to modify the SOAP Gateway server properties.

Undeploying a web service

Undeploying a web service removes the associated web service archive file and correlator XML file for the IMS application from both the runtime configuration and the master configuration.

To undeploy a web service:

1. Verify the WSDL file and correlator XML file for the web service. WSDL files can contain multiple services and operations that are accessed separately as web services. If more than one correlator is associated with the WSDL file (such as in the case of a multiple service WSDL file), issue additional undeploy commands for each additional correlator file.
2. Issue the SOAP Gateway management utility command `iogmgmt -undeploy -r correlator_file`. Do not specify an absolute path for the correlator file. Use only the file name.

Related concepts:

"SOAP Gateway master configuration and runtime configuration" on page 25
The master configuration is the authoritative configuration of the SOAP Gateway server, and is stored in the file system. The active server configuration in memory is the runtime configuration.

Related reference:

"-undeploy: Undeploy a web service or callout application" on page 461
Use the -undeploy command to undeploy a web service or callout application. Undeploying a service removes the XML file for the service from the runtime cache and master configuration.

Administrative tasks for SOAP Gateway callout

Use the SOAP Gateway management utility to set connection and correlator properties, and manage the callout threads that SOAP Gateway uses to make callout or business event requests from IMS applications to external web services or business event servers.

Deploying a callout application to SOAP Gateway

Deploy a callout application or business event emitter to SOAP Gateway with the SOAP Gateway management utility.

A callout application requires a valid correlator XML file with interaction properties for the application, a valid WSDL file that describes the web service to clients (or XSD file for calls to a business event monitoring server), and a valid connection bundle with connection properties for the application.

1. Ensure that the required web service artifacts are located in the SOAP Gateway installation directory. The correlator XML file and WSDL (or XSD) file can either be in the XML and WSDL directories, or elsewhere in the SOAP Gateway directory. If the files are already located in the XML and WSDL directories when you deploy the web service, you can provide the file name without the fully qualified path to the file.

   Tip: z/OS For z/OS systems, the correlator file and the WSDL file, when uploaded from a distributed platform, must be transferred in BINARY mode from your local workstation. Binary transfers provide a bit-by-bit copy that preserves the encoding on your system by instructing the FTP socket not to convert the encoding to the local system encoding (EBCDIC).

   Restriction: Nested XSD import statements are supported only by Rational Developer for System z Version 8.5.1 or later in its top-down support for COBOL data structure generation for synchronous callout. Sharing of XSD schema files among callout applications is not supported. See the -deploy command reference for details.

2. If you are not using the XML adapter function in IMS Connect, update your correlator file by using the SOAP Gateway management utility iogmgmt -corr command and setting the `-a` option (adapter type) to `No_Adapter`. By specifying the `No_Adapter` value, the adapterType entry in the correlator is set to blank. By default, this entry is set to `IBM XML Adapter`.
3. Issue the command to deploy the application to the server:
   - For a one-way or request-response callout application, or a business event application that emits events to WebSphere Business Events, issue the command `iogmgmt -deploy -w wsdl_file -r correlator_file.`
   - For a business event application that emits events to WebSphere Business Monitor, issue the command `iogmgmt -deploy -w xsd_file -r correlator_file.`

SOAP Gateway is configured to pass callout requests or business events to the target web service or business event monitor.

Related reference:

"-deploy: Deploy a web service or callout application" on page 444
The -deploy command deploys a web service, callout application, or business event application to the active configuration of the SOAP Gateway server.

Starting and stopping all callout threads

You can start and stop the callout threads by using the SOAP Gateway management utility.

The worker thread pool must be started before you can start callout threads.

The callout threads poll defined tpipes for new callout request messages and then add the messages to the work queue. If the callout threads are stopped, SOAP Gateway does not poll tpipes for new callout request messages.

The worker thread pool pulls messages from the work queue and open connections to external web services. If the worker thread pool is stopped, the work queue fills up with unprocessed messages and SOAP Gateway halts callout processing.

The server must have an active HTTP listening port before you can issue callout thread management commands.

1. Ensure that the worker thread pool is started with the `iogmgmt -view -workerthreads` command.
2. Start all callout threads with the `iogmgmt -callout -startall` command.
3. Optional: Verify that the callout threads are started with the `iogmgmt -view -calloutthreads` command.

The callout threads begin processing callout request messages.

Before you stop the worker thread pool, stop all callout threads with the `iogmgmt -callout -stopall` command. Stopping callout threads before you stop the worker thread pool ensures that no callout request messages are left unprocessed in the work queue.

Related concepts:

"Thread management for callout messages retrieval" on page 174
SOAP Gateway supports two options to determine how to manage the callout threads to send the requests to poll the hold queue for callout request messages:

one thread per tpipe, or one thread per connection bundle.

Starting the callout thread for a specific application

You must start a callout thread after a callout application is deployed to SOAP Gateway before the application can begin processing callout messages.

This task operates at the thread level. The behavior of this task depends on the thread policy that has been set. If the policy is to have one thread per tpipe, the start action affects only that particular tpipe. If the thread policy is set to one thread per connection bundle, the start action affects the callout thread for every tpipe defined in the connection bundle.

1. Ensure that the SOAP Gateway server has an active HTTP listening port. The SOAP Gateway management utility commands for thread management require an active HTTP listening port.
2. Ensure that the callout thread pool is started with the `iogmgmt -view -workerthreads` command. Callout threads poll messages from defined tpipes and send them to be picked up by worker threads in the thread pool. The worker threads then invoke external web services. If the thread pool is stopped while callout threads are running, the work queue fills up with unprocessed messages and callout processing halts.
3. Determine which callout thread must be started.
   - If your current thread policy is one thread per tpipe, you must know both the name of the connection bundle specified in the correlator XML file for the callout application and the name of each tpipe in the connection bundle that ais associated with the application.
   - If your current thread policy is one thread per connection bundle, you only must know the name of the connection bundle.
4. Start the callout thread to begin processing callout requests for the application with the `iogmgmt -callout -startone` command.
   - If your thread policy is one thread per tpipe, the command is `iogmgmt -callout -startone -c` *`connection_bundle_name`* `-p` *`tpipe_name`*. You must issue the command for each tpipe associated with the application to start all of the relevant callout threads.
   - If your thread policy is one thread per connection bundle, the command is `iogmgmt -callout -startone -c` *`connection_bundle_name`*.
5. Optional: Verify that the callout thread for the application is running with the `iogmgmt -view -calloutthreads` command. A callout thread might take some time to start, depending on available system resources.

The callout thread for the connection bundle (and tpipe name, if applicable) that you specified begins to process callout messages.

Related concepts:

"Thread management for callout messages retrieval" on page 174
SOAP Gateway supports two options to determine how to manage the callout threads to send the requests to poll the hold queue for callout request messages: one thread per tpipe, or one thread per connection bundle.

Related reference:

"-callout -startone: Start a specific callout thread" on page 431
The -callout -startone command starts a specific callout thread based on the provided connection bundle name and tpipe name.

"-callout -stopone: Stop a specific callout thread" on page 432
The -callout –stopone command stops a specific callout thread based on the

provided connection bundle name and tpipe name.

Stopping the thread pool

Stopping the thread pool stops all worker threads. When the worker threads are stopped, no messages in the callout work queue are processed.

To ensure that no in-flight messages are dropped and that the callout work queue does not overflow, when the thread pool is stopped, SOAP Gateway would wait for all in-flight messages to be processed before stopping the thread pool. Likewise, when you shut down the server, SOAP Gateway would stop the thread pool after all in-flight messages are processed.

To stop the thread pool gracefully, use the SOAP Gateway management utility iogmgmt -callout -stoppool command.

To force the thread pool to stop immediately without waiting for messages in the callout work queue to be processed, use the SOAP Gateway management utility iogmgmt -callout -stoppool -force command.

Verify that all worker threads are stopped with the iogmgmt -view -workerthreads command.

Related concepts:

"Thread management for callout messages retrieval" on page 174
SOAP Gateway supports two options to determine how to manage the callout threads to send the requests to poll the hold queue for callout request messages: one thread per tpipe, or one thread per connection bundle.

Creating a correlator file for a callout application

You can manually create a correlator file with the SOAP Gateway management utility if you do not have IBM Rational Developer for System z.

A correlator file provides the information to invoke the outbound web service and return the response message back to IMS. The information includes:

- The WSDL file name for the web service or the WebSphere Business Events server to be invoked, or the XSD file name for the WebSphere Business Monitor server to be invoked
- The timeout value for waiting for a response from the web service or business event server
- The IMS transaction code and the connection bundle name for returning the callout response

Restrictions:

- The SOAP Gateway management utility does not support the creation of correlator files that work with an XML adapter in the target IMS Connect. Use IBM Rational Developer for System z instead.
- The SOAP Gateway management utility does not support the creation of correlator XML files for the WebSphere Business Monitor scenario. Use IBM Rational Developer for System z instead.

To create a correlator file:

1. Determine the specific callout scenario your SOAP Gateway callout application supports. A SOAP Gateway callout application supports one of the following scenarios:

- An asynchronous call to a remote web service. A response may or may not be expected:
  - No response is expected from the target web service. This is also referred to as a "one-way" call. If the target server is expected to respond, it must invoke a separate SOAP Gateway web service.
  - For an asynchronous request-response application, SOAP Gateway returns the response to IMS Connect in a different transaction. In this case, you can optionally specify a separate connection bundle name, in addition to the normal callout connection bundle name, to handle the response message. If you specify only a callout connection bundle name, the response message is returned to the same host that issued the original callout request.
- A synchronous call to a remote web service. A response from the target server is expected on the same connection as the request message. In this case, you can optionally specify a separate connection bundle name, in addition to the normal callout connection bundle name, to handle the response message. If you specify only a callout connection bundle name, the response message is returned to the same host that issued the original callout request.
- A call to a business event monitoring server. SOAP Gateway can emit business event data to either WebSphere Business Events or WebSphere Business Monitor.
  - The WebSphere Business Events scenario is functionally the same as a one-way call to a remote web service. A WSDL file is required.
  - The WebSphere Business Monitor scenario uses the REST protocol and requires an XSD service definition file instead of a WSDL file. Additionally, the correlator XML file must be configured with the URI of the WebSphere Business Monitor server. Creation of correlator XML files for this scenario is only supported with IBM Rational Developer for System z.

2. Gather the information required to create a correlator for a callout application.
   a. Determine the WSDL or XSD file name for the application. The WSDL file contains the web service definition for a SOAP Gateway application. However, calls to WebSphere Business Monitor use the REST protocol and use an XSD file instead.
   b. Determine the service and operation names for the application. A WSDL file can contain multiple service and operation definitions, and so SOAP Gateway uses the combination of service and operation name as the unique identifier for the web service.
   c. Determine the callout connection bundle name. The callout connection bundle contains the properties that determine how SOAP Gateway handles message traffic that is sent from IMS to the target web service. Most importantly, the callout connection bundle specifies one or more tpipes that are used for inbound and outbound messages.
   d. Optional: For a request-response callout application, determine the connection bundle name. You can specify a non-callout connection bundle name (in addition to the required callout connection bundle name) in the correlator for a request-response callout application. This option allows you to configure how SOAP Gateway handles the response message in the same way that it handles incoming requests for web services.

e. Optional: Determine other correlation properties. A callout application correlator can contain other properties to override default interaction behaviors such as time out values. However, these properties are not required.

3. Create the correlator file by using Rational Developer for System z. If you are not using the XML adapter function, create the correlator file with the SOAP Gateway management utility `iogmgmt -corr -c` command.
   - For a one-way callout application correlator, or a synchronous request-response callout application correlator, issue the command `iogmgmt -corr -c -w wsdl_file -p operation_name -i service_name -d callout_connection_bundle_name`. You can specify multiple connection bundles for a synchronous callout correlator by separating the connection bundle names with commas: `iogmgmt -corr -c -w wsdl_file -p operation_name -i service_name -n connection_bundle_name -d callout_connection_bundle_name1,callout_connection_bundle_name2`.
   - For an asynchronous request-response callout application correlator, issue the command `iogmgmt -corr -c -w wsdl_file -p operation_name -i service_name -n connection_bundle_name -d callout_connection_bundle_name`.

   A new correlator XML file is created in the XML directory.
4. If you are not using the XML adapter function in IMS Connect, update the correlator and set the `-a` option (adapter type) to `No_Adapter`.

   ```
   iogmgmt -corr -u -r correlator file -p operation_name -i service_name  -a No_Adapter
   ```

   By default, this entry is set to `IBM XML Adapter`. By specifying `No_Adapter` for the **-a** option, the adapterType entry in the correlator is set to blank.
5. Optional: Save a copy of the correlator XML file outside of the XML directory. An undeploy operation deletes the associated correlator file for the web service from the XML directory. Saving a copy elsewhere preserves it when the associated web service is undeployed.

Related concepts:

"Correlator file" on page 22
The correlator file specifies transaction and runtime properties. This file also specifies the information that SOAP Gateway needs to match incoming requests to the appropriate backend IMS application and outgoing requests from an IMS application to a web service.

Related tasks:

Chapter 7, "Enabling an IMS application to emit a business event," on page 271
To enable an application to emit a business event, you must modify your IMS application, define an OTMA destination descriptor, generate the correlator file, the XML converter, and the data mapping XSD file, and configure SOAP Gateway for the business event server.

Related reference:

"-corr: Create or update a correlator entry" on page 439
Use the -corr command to create or update the transaction and runtime properties of a correlator entry.

Creating a connection bundle entry for callout applications

Create a connection bundle entry that describes the connection properties for accessing IMS by using the SOAP Gateway management utility. The connection bundle entries are stored in the `connbundle.xml` file.

You must create a new connection bundle entry or modify an existing connection bundle entry to specify the following properties:

- The connection properties for IMS Connect
- The names of the tpipes that hold the synchronous and asynchronous callout requests that are sent by your IMS application

If a response message is expected from the web service, you must either create an additional connection bundle entry or reuse an existing connection bundle entry to specify connection properties for sending the output response message. The connection bundle entry with the connection properties for the response message is specified separately from the connection bundle entry with the connection properties for the callout message. Each is specified with a separate entry in the callout correlator XML file. However, both sets of properties can be stored in the same connection bundle entry.

1. Gather the required information for the connection bundle entry. A connection bundle entry for a callout application must contain the following information:
   - Name for the connection bundle entry
   - Name of the callout tpipe for the application
   - IMS Connect host name
   - IMS Connect listening port number (default is 9999)
   - IMS Connect datastore name
2. Optional: To enable callout basic authentication, gather the following information:
   - Callout basic authentication user ID
   - Callout basic authentication password

   This option configures SOAP Gateway to perform basic authentication with the target web service as part of a callout request.
3. Optional: To enable server SSL authentication for the callout request, gather the following information:
   - Callout SSL truststore name
   - Callout SSL truststore password

   This option configures SOAP Gateway to confirm the identity of the server by verifying the information in the server truststore.
4. Optional: To enable client SSL authentication for the callout request, gather the following information:
   - Callout SSL keystore name
   - Callout SSL keystore password

   This option configures SOAP Gateway to send the server a client certificate that the server can use to confirm the identity of the SOAP Gateway server. Server SSL authentication must also be configured in the connection bundle entry to use client SSL authentication.
5. Issue the `iogmgmt -conn -c` command to create the connection bundle entry.

   Example 1. No security. The following example creates a connection bundle entry named `connbundle1` that connects to an IMS Connect host named ICONHOST1 on port 9998 and that uses a callout tpipe named tpipe1. Security is not enabled.

   ```
   iogmgmt -conn -c -n connbundle1 -h ICONHOST1 -p 9998 -d IMSSTOR1 -i tpipe1
   ```

   Example 2. Basic authentication. The following example creates a connection bundle entry named `connbundle2` that connects to the IMSSTOR2 data store on

an IMS Connect host named ICONHOST2 on port 9995. The callout tpipe is tpipe2. Basic authentication user ID and password is specified for basic authentication.

```
iogmgmt -conn -c -n connbundle2 -h ICONHOST2 -p 9995 -d IMSSTOR2 -i tpipe2
 -m basicAuthID  -b basicAuthPwd
```

Example 3. Client authentication and basic authentication. The following example creates a connection bundle entry named `connbundle3` that connects to the IMSSTOR3 data store on an IMS Connect host named ICONHOST3 on port 9992. The callout tpipe is tpipe3. In addition to basic authentication, target keystore and truststore information is provided for client authentication.

```
iogmgmt -conn -c -n connbundle3 -h ICONHOST3 -p 9992 -d IMSSTOR3 -i tpipe3
-l callout_target_ks_name -y callout_target_ks_pwd
-v callout_target_ts_name -q callout_target_ts_pwd
-m basicAuthID -b basicAuthPwd
```

A message indicates that the connection bundle entry is created.

Related concepts:

"Connection bundle properties" on page 20
The connection bundle specifies the connection and security properties for SOAP Gateway when it communicates with IMS Connect.

Related tasks:

"Configuring SSL and HTTPS support with Java keystore (JKS)" on page 148
To use HTTPS between a SOAP Gateway client and SOAP Gateway, or SSL between SOAP Gateway and its server (IMS Connect), you must create the keystore and truststore, and configure the SOAP Gateway server.

Related reference:

"-conn: Create, update, or delete a connection bundle" on page 435
Use the -conn command to create, update, or delete a connection bundle.

Updating SOAP Gateway callout properties

You can update SOAP Gateway callout properties by using the SOAP Gateway management utility.

Gracefully stop the thread pool before modifying the SOAP Gateway callout properties. Modifying the callout properties of a server with an active thread pool or callout threads can result in unexpected behavior.

The callout properties define how the SOAP Gateway server handles callout message processing and error conditions.

To update SOAP Gateway callout properties:

1. Verify that the thread pool is stopped with the `iogmgmt -view -calloutthreads` command.
2. Issue the `iogmgmt -callout -updateprop` command to modify callout properties.
3. Optional: Verify the update with the `iogmgmt -view -calloutproperties` command.
4. Restart the thread pool with the `iogmgmt -callout -startpool` command, and then restart all callout threads with the `iogmgmt -callout -startall` command.

Callout message processing resumes with the updated properties.

Related reference:

"Callout properties for thread management" on page 179
SOAP Gateway provides several properties for configuring and managing the threads.

"-callout -updateprop: Update SOAP Gateway callout properties" on page 434
The -callout –updateprop command updates the SOAP Gateway callout properties.

Undeploying a callout application

Use the SOAP Gateway management utility to undeploy a callout application or application that emits business event data.

Undeploying a callout application removes the application from the runtime cache and also deletes the associated WSDL or XSD file and correlator XML file from the master configuration of the server.

1. Gracefully stop callout message processing for the application with the `iogmgmt -callout -stopone` command.
   - If your current thread policy is one thread per tpipe, issue the command `iogmgmt -callout -stopone -c` *`connection_bundle_name`* `-p` *`tpipe_name`* once for each tpipe associated with the callout application.
   - If your current thread policy is one thread per connection bundle, issue the command `iogmgmt -callout -stopone -c` *`connection_bundle_name`*.
2. Undeploy the callout application with the `iogmgmt -undeploy -r` *`correlator_file`* command. The SOAP Gateway management utility removes the WSDL or XSD file specified in the correlator XML file that is being undeployed, as well as the correlator XML file itself. Do not specify an absolute path for the correlator file. Use only the file name.

The IMS callout application is removed from the runtime cache. The corresponding WSDL and the correlator XML files are removed from the WSDL and XML directories, respectively.

Related reference:

"-undeploy: Undeploy a web service or callout application" on page 461
Use the -undeploy command to undeploy a web service or callout application. Undeploying a service removes the XML file for the service from the runtime cache and master configuration.

Chapter 9. Tracking and monitoring SOAP Gateway transactions

Transaction requests routed through a SOAP Gateway server can include a unique ID that correlates the transactions. Depending on the usage scenario, the transactions can be correlated between the client application (provider scenario), external web service (callout scenario), SOAP Gateway, or IMS (provider scenario).

Inbound request messages (the provider scenario)

For inbound web service requests, SOAP Gateway uses two different types of IDs to help correlate SOAP requests and responses for the provider scenario: vertical IDs and horizontal IDs.

- Vertical IDs are generated by SOAP Gateway to correlate all of the SOAP Gateway message processing events that are associated with that request. Vertical IDs are not propagated to IMS Connect.
- Horizontal IDs (enabled by using the iogmgmt -tracking command) can be configured and are propagated to IMS Connect, so you can use them to trace a specific request through SOAP Gateway, IMS Connect, IMS, and back. SOAP Gateway can be configured to use the value of the WS-Addressing messageID element or a user-specified element in the incoming SOAP message header as the horizontal ID, or use the vertical ID as the horizontal ID. If the ID is propagated from the request message, SOAP Gateway does not validate the ID or guarantee that it is unique.

The horizontal ID is also propagated to IMS Connect in the IRM header. IMS Connect includes the ID with the information captured by the IMS Connect Event Recorder exit routine (HWSTECL0). This information can be consumed by the IBM IMS Connect Extensions for z/OS and equivalent tools.

The IMS log records for the transaction also include the horizontal ID. You can use IBM IMS Performance Analyzer for z/OS and IBM IMS Problem Investigator for z/OS, or equivalent tools, to inspect IMS log records.

After IMS completes the requested transaction, a response is sent via IMS Connect and SOAP Gateway to the client application. The response message also includes the ID from the request message, and processing events related to the response are also available in IMS, IMS Connect, and SOAP Gateway.

SOAP Gateway can also send the ID information to an IBM Tivoli Composite Application Manager for Transactions (ITCAM) data collection server.

Outbound request messages (the callout scenario)

For outbound web service requests, SOAP Gateway generates a vertical ID to help correlate the transactions. All the request and response processing messages and events include this generated tracking ID. Unlike the provider scenario, SOAP Gateway does not support generation of the horizontal ID to propagate the information to IMS Connect.

You can use the vertical ID to identify message processing events in the SOAP Gateway transaction log or in the transaction information sent to a remote IBM

Tivoli Composite Application Manager for Transactions (ITCAM) data collection server.

Transaction logging and monitoring

SOAP Gateway can generate a JSON log file that contains a record for each event that occurs during the request and response processing. To enable transaction logging and specify the log file location and attributes, use the iogmgmt -tranLog command. Both the provider and the callout requests are logged to the same file. To differentiate between the two scenarios, a`Scenario: "CONSUMER"` property is included in the event log for each callout message processing event.

SOAP Gateway can also send the event information to a remote IBM Tivoli Composite Application Manager for Transactions (ITCAM) data collection server. The log file includes the horizontal (provider only) and vertical IDs that are required to correlate the different events that are associated with a single request message. To send the transaction information to a remote IBM Tivoli Composite Application Manager for Transactions (ITCAM) data collection server for monitoring, use the iogmgmt -tranAgent command to enable the function and set the address of the target data collection server.

SOAP Gateway also provides a provider monitoring MBean that provides statistics about the SOAP Gateway server and deployed web services.

The following table describes the features and the related command by usage scenario.

Table 35. Supported transaction tracking and monitoring features and related commands

Task	Provider	Callout
Log transaction information to a SOAP Gateway transaction log (with vertical IDs only)	Issue `iogmgmt -tranLog -on`	
Send transaction information to a remote data collection server (with vertical IDs only)	Issue `iogmgmt -tranAgent -on -port` *`port_number`* `-address` *`address`*	
Generate a horizontal ID to pass to IMS for SOAP Gateway-to-IMS transaction tracking. **Note:** This ID will also be included in the transaction information sent to the SOAP Gateway transaction log or the remote data collection server.	Issue `iogmgmt -tracking -on`	Not supported.
Obtaining statistics about the SOAP Gateway server and deployed web services by using the SOAP Gateway monitoring MBean.	Issue `iogmgmt -mbeans -on -port` *`port_number`*	Not supported.

You can view the current configuration of the SOAP Gateway tracking and monitoring functions with the SOAP Gateway management utility iogmgmt -view -soapgatewayproperties command.

Related concepts:

"Transaction log format" on page 307
The SOAP Gateway transaction logger creates JSON files with a record for each message processing event.

Related tasks:

"Configuring the transaction log" on page 325
The SOAP Gateway transaction log records information about every inbound and outbound request and the associated response message to and from IMS.

Related reference:

"-tranLog: Configure the SOAP Gateway transaction logger" on page 460
Use the -tranLog command to enable or disable the SOAP Gateway transaction logger for both web service provider and callout transactions.

"-tranAgent: Configure the IBM Tivoli Composite Application Manager for Transactions (ITCAM) Transaction Tracking API (TTAPI)" on page 459
Use the -tranAgent command to enable or disable the IBM Tivoli Composite Application Manager for Transactions (ITCAM) Transaction Tracking API (TTAPI) and set the address for the target data collection server.

"-tracking: Configure SOAP Gateway-to-IMS transaction tracking IDs" on page 457
Use the -tracking command to enable, disable, and configure the horizontal tracking IDs for inbound SOAP messages (the provider scenario).

"-mbeans: Configure SOAP Gateway JMX monitoring" on page 447
Use the -mbeans command to switch the SOAP Gateway server JMX monitoring MBeans on or off and set the port number for JVM monitoring.

SOAP Gateway message processing events

Each request message sent to a SOAP Gateway web service provider (and the associated response message) generates several uniquely identified message processing events. Likewise, each callout request generates several message processing events, from receiving the IMS callout request to receiving the response from the external web service and sending the response back to IMS.

For an inbound request message, SOAP Gateway records the following event types, which are listed in the sequence of the processing flow. The numbers correspond to the numbers in the event flow diagram that follows.

- 1 Event type 0: The server received a request from the client.
- 2 Event type 10: The server validated the request message, including security validation if the server is configured to use WS-Security.
- 3 Event type 12: SOAP Gateway sent the request message to IMS Connect.
- 4 Event type 11: SOAP Gateway retrieved the response message from IMS Connect.
- 5 Event type 17: the SOAP response message was sent to the client application. The response can be a normal message response, a SOAP fault, an IMS Connect error response, an IMS error response, or an error from the target IMS application program.

The following diagram shows when the events are recorded during the inbound request processing.

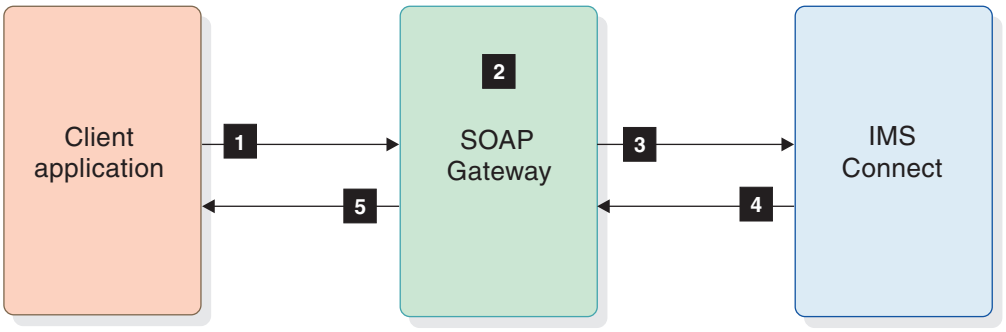


Figure 72. Inbound web service request processing event flow

For an outbound callout message, SOAP Gateway records the following event types, which are listed in the sequence of the processing flow. The numbers correspond to the numbers in the event flow diagram that follows.

- 1 Event type 0: SOAP Gateway issued a resume tpipe request to IMS Connect.
- 2 Event type 10: SOAP Gateway received the callout request.
- 3 Event type 8 and event type 9: SOAP Gateway validated the request and committed to processing. If an error occurs at this stage, an event type 9 is logged.
- 4 Event type 12: SOAP Gateway sent the request message to the external web service.
- 5 Event type 11: SOAP Gateway retrieved the response message from the external web service (for request-response mode callout messages)
- 6 Event type 13 or 17:
  - Event type 13: SOAP Gateway sent the response message to IMS Connect. An acknowledge from IMS Connect is expected (send-only-with-ack flag is set).
  - Event type 17: SOAP Gateway sent the response message to IMS Connect and the transaction ended. No acknowledgment from IMS Connect is expected.
- 7 Event type 8: SOAP Gateway received an acknowledgment from IMS Connect that the response is received (send-only-with-acknowledgment flag is set). If a NACK is received, and event type 9 is logged.

The following diagram shows when the events are recorded during the outbound request processing. Steps 5, 6, and 7 do not apply for asynchronous one-way invocation.

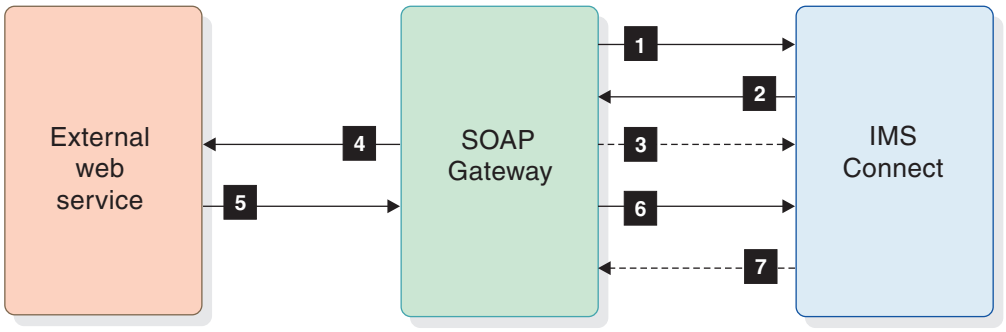


Figure 73. Outbound callout request processing event flow

Errors during message processing might cause a message to fail before all event types are recorded for the message. To determine why an error occurred, for the callout scenario, check the latest transaction logs for that callout request and examine the details in the log for any errors. For the provider scenario where the horizontal ID is propagated to IMS, search the SOAP Gateway server log, IMS Connect log records, or IMS log records for the horizontal ID associated with the failed message.

Related concepts:

"Provider request transaction log format by event type" on page 307
Use the value of the linkID field in the verticalID element of each message processing event to uniquely correlate every event related to a specific SOAP request message. The vertical link ID for a request is always unique, and the value is included in every event log entry.

"Callout request transaction log format by event type" on page 314
The SOAP Gateway transaction logger creates JSON files with a record for each callout message processing event.

IDs for transaction correlation

SOAP Gateway uses two different types of IDs to help track and correlate SOAP requests and responses: vertical IDs and horizontal IDs.

Vertical IDs

Vertical IDs are generated by SOAP Gateway and are not configurable. The vertical ID of a request is used to correlate all of the SOAP Gateway message processing events associated with that request using a single unique ID. Vertical IDs are always unique. Vertical IDs are included in the data written to the transaction log file or sent to a remote IBM Tivoli Composite Application Manager for Transactions (ITCAM) data collector, if either is enabled. Vertical IDs are not propagated to IMS Connect.

For callout request processing, SOAP Gateway generates vertical IDs only.

Horizontal IDs

Horizontal IDs can be propagated from different parts of the incoming SOAP request message, or they can be generated by SOAP Gateway. Horizontal IDs are for inbound request messages (the provider scenario) only.

You can configure how SOAP Gateway assigns horizontal IDs to request messages. Horizontal IDs are also included in the data written to the transaction log file or sent to a remote IBM Tivoli Composite Application Manager for Transactions (ITCAM) data collector, if either is enabled.

Horizontal IDs are propagated to IMS Connect, so you can use them to trace a specific request through SOAP Gateway, IMS Connect, IMS, and back.

The IMS Connect IRM header provides a field that SOAP Gateway uses to propagate a *tracking ID* to IMS Connect and IMS. SOAP Gateway provides three different methods to generate the horizontal ID, which is included in the tracking ID field:

- SOAP Gateway can get the value of the `messageID` element in the incoming SOAP message header, and use that value as the horizontal ID. If the `messageID` element is empty or contains more than 40 bytes of data, SOAP Gateway generates an ID for the message.
- SOAP Gateway can get the value of a user-specified element in the incoming SOAP message header, and use that value as the horizontal ID. If the specified element is empty or contains more than 40 bytes of data, SOAP Gateway generates an ID for the message.
- SOAP Gateway can generate a unique horizontal ID for every incoming SOAP message. In this case, the horizontal ID is the same as the vertical ID for the request.

You can activate and configure this feature with the SOAP Gateway management utility iogmgmt -tracking -on command.

For the provider scenario, SOAP Gateway uses the IRM header area to propagate tracking IDs to IMS Connect and IMS.

Related tasks:

"Configuring the transaction log" on page 325
The SOAP Gateway transaction log records information about every inbound and outbound request and the associated response message to and from IMS.

"Configuring the IBM Tivoli Composite Application Manager for Transactions (ITCAM) Transaction Tracking API (TTAPI)" on page 346
SOAP Gateway includes an implementation of the IBM Tivoli Composite Application Manager for Transactions (ITCAM) Transaction Tracking API (TTAPI). You can use the ITCAM TTAPI to send information about SOAP Gateway message traffic to a transaction collector for ITCAM.

Related reference:

"-tracking: Configure SOAP Gateway-to-IMS transaction tracking IDs" on page 457
Use the -tracking command to enable, disable, and configure the horizontal tracking IDs for inbound SOAP messages (the provider scenario).

Remote monitoring options for SOAP Gateway

SOAP Gateway provides two interfaces for remote monitoring.

The following table shows a comparison of the SOAP Gateway remote monitoring options:

Table 36. Comparison of SOAP Gateway server monitoring features

Feature	Purpose
Provider Monitoring MBean	The SOAPGatewayProviderMonitorMBean interface provides a remote JMX monitoring capability for SOAP Gateway server administrators. This interface can be used to monitor the workload statistics of web services deployed on the SOAP Gateway server.

Table 36. Comparison of SOAP Gateway server monitoring features (continued)

Feature	Purpose
IBM Tivoli Composite Application Manager for Transactions (ITCAM) Transaction Tracking API (TTAPI)	The IBM Tivoli Composite Application Manager for Transactions (ITCAM) Transaction Tracking API (TTAPI) implementation in SOAP Gateway connects to a remote IBM Tivoli Composite Application Manager for Transactions (ITCAM) server for aggregated performance monitoring and transaction tracking. **Restriction:** The ITCAM TTAPI does not record information for WebSphere Business Events emitters.

Configuring the SOAP Gateway monitoring MBean

The SOAP Gateway MBean interfaces provide statistics about the SOAP Gateway server and deployed web services (provider scenario).

If you want to run JConsole remotely from the SOAP Gateway server, you must install the Java Development Kit, version 7 or later, on the remote workstation. A remote workstation is required to use JConsole with a SOAP Gateway server running on z/OS.

The SOAP Gateway server provides a complete implementation of the JMX architecture (probe, agent, and remote management connector). You must provide a remote monitoring application to communicate with the interfaces provided by SOAP Gateway. The remote monitoring application can be a generic monitoring console such as JConsole, a JMX-compatible monitoring application. This example demonstrates how to configure the SOAP Gateway monitoring interface and get statistics with JConsole.

SOAP Gateway provides access to the standard Apache Tomcat 7.0 MBeans for server health-monitoring and statistics, and the SOAPGatewayProviderMonitorMBean interface that provides statistics about SOAP Gateway web services activity, connection bundles, and connections to IMS Connect.

Restriction: The SOAPGatewayProviderMonitorMBean interface does not include statistics for callout applications or WebSphere Business Events applications.

1. Issue the command `iogmgmt -mbeans -on -port` *xxxx* to the SOAP Gateway server to activate the MBeans interfaces. The *xxxx* value is the server listening port for MBean requests. Do not share the MBeans listening port number with the web service listening port, secure listening port, or server shutdown port. If the specified port is in use when the SOAP Gateway server starts, the MBeans interface is disabled.
2. Start or restart the server. The server must be restarted (or started, if it is not running) to activate the MBean interfaces.
3. Start JConsole either remotely (on a separate workstation) or locally (on the server). The JConsole application is located in *JDK_installation_directory*`/bin`.
   - Windows Linux z/OS Start JConsole on the remote workstation. Click **Remote Process** and enter the host name of the server and the port number that you selected in step 1. Click **Connect**.
   - Windows Linux Start JConsole on the SOAP Gateway server. Click **Local Process** and then select the **org.apache.catalina.startup.Bootstrap** process in the list. Click **Connect**. You can also use the **Remote Process** option with the local IP address and JMX port number to connect to a local server.

JConsole connects to the SOAP Gateway server. The JConsole window displays basic JVM information including heap memory usage, thread count, loaded classes, and CPU utilization.

4. Switch to the **MBeans** tab. The JConsole window shows a list of the active MBeans on the SOAP Gateway server.
5. In the navigation tree, expand the folder **com.ibm.ims.soap.server**, then the node **SOAPGatewayProviderMonitorMBean**, and then click **Operations**.

The JConsole window displays a list of operations. You can get various statistics from the server by clicking the button for an operation. A pop-up window opens with the returned value from the server. Some operations with complex argument or return types are not available in JConsole.

Related reference:

"-mbeans: Configure SOAP Gateway JMX monitoring" on page 447
Use the -mbeans command to switch the SOAP Gateway server JMX monitoring MBeans on or off and set the port number for JVM monitoring.

Related information:

Using JConsole
See more information about JConsole at docs.oracle.com.

Apache Tomcat 7 MBean Names
See a list of MBeans that are available in the Tomcat servlet container instrumentation at the Apache Software Foundation.

Java API specification for the SOAP Gateway monitoring MBean

This API specification includes reference information about the Java Management Extensions (JMX) monitoring MBean for the SOAP Gateway server.

You can use this functionality with a user-supplied Java application or with a monitoring client that supports MBeans such as JConsole.

Configuring the IBM Tivoli Composite Application Manager for Transactions (ITCAM) Transaction Tracking API (TTAPI)

SOAP Gateway includes an implementation of the IBM Tivoli Composite Application Manager for Transactions (ITCAM) Transaction Tracking API (TTAPI). You can use the ITCAM TTAPI to send information about SOAP Gateway message traffic to a transaction collector for ITCAM.

You must have a transaction collector to receive the data from the SOAP Gateway server.

The ITCAM TTAPI collects information about every inbound web service request to SOAP Gateway and every outbound callout request from SOAP Gateway, and the associated response messages. This information is sent to the transaction collector and no data is stored on the local file system.

The same information collected by the ITCAM TTAPI can also be collected by the SOAP Gateway transaction logger. The transaction logger writes the information to a local log file. See "Configuring the transaction log" on page 325.

Restriction: The ITCAM TTAPI does not record information for WebSphere Business Events emitters.

1. Start the remote transaction collector. Note the host name (or IP address) and port number of the transaction collector.
2. Start the ITCAM TTAPI with the `iogmgmt -tranAgent -on -address` *`server_addres`* `-port` *`server_port`* SOAP Gateway management utility command. The server address (or host name) and port number are specified for the ITCAM TTAPI are those of the remote transaction collector.

The ITCAM TTAPI starts and immediately begins sending information to the transaction collector.

Related tasks:

"Configuring the transaction log" on page 325
The SOAP Gateway transaction log records information about every inbound and outbound request and the associated response message to and from IMS.

Chapter 10. Troubleshooting

Potential problems could occur in the installation verification program (IVP) for SOAP Gateway or during run time. Tips and techniques are provided for troubleshooting the IVP, callout requests, and runtime errors.

Diagnosing Installation Verification Program errors

Running the IVP for SOAP Gateway is usually trouble free. However, if you do experience an error, this topic lists the possible errors and recommended solutions for these problems.

If the IVP does not complete successfully, it might be due to one of the following errors:

- A page not found error when accessing the SOAP Gateway IVP web client from the web browser. Possible causes for this error are:
  - Possible cause of this error:
    - The SOAP Gateway server may not be started.
    - The host name and port number you specified in the URL of the IVP web client for SOAP Gateway is not correct.
  - User action:
    - Use the SOAP Gateway management utility to stop and then start the server.
    - Use the `netstat -o` command to check if the port is reserved on z/OS. Then use the SOAP Gateway management utility to change the port number.
- You receive the error message, `com.ibm.ims.soap.xmlparser.XMLFileException: IOGS029E: Connection bundle name [imssoapivp] not found.`
  - Possible cause of this error:
    - The connection bundle with the name “imssoapivp” has not been created.
    - SOAP Gateway was not restarted after the connection bundle **imssoapivp** was created.
  - User action:
    - Follow the instructions in “Verifying the installation of SOAP Gateway” to update the connection bundle named “imssoapivp” for the correct host name, data store name, and port number.
    - Ensure that you restart SOAP Gateway after you create the connection bundle and before you run the IVP.
- You receive the error message, `com.ibm.ims.soap.server.IMSSOAPException: IOGC003E: Failed to send and receive messages from IMS Connect. Hostname` *`hostname`*`, port` *`port`*`. [SocketException: Connection reset]`
  - Possible cause of this error:
    - The User Message Exit HWSSOAP1 has not been properly installed with IMS Connect to handle the input message from SOAP Gateway. If this error happens, you will also see the following message on the z/OS console for the associated IMS Connect: `HWSP1445E UNKNOWN EXIT IDENTIFIER SPECIFIED IN MESSAGE PREFIX; MSGID=*HWSOA1*/ HWSSOA1, M=SDRC.`

- User action: Follow the instructions in "Setting up the user exit routine for IMS Connect" to ensure that the User Message Exit HWSSOAP1 installed properly.

- An error message returned when you invoke the SOAP Gateway IVP web client.
  - Possible cause of this error:
    - Invalid data was specified when configuring the connection bundle properties for the host IMS environment. For example:
      - The IMS host name is misspelled and is not sufficiently qualified (for TCP/IP communication).

        Attention: In some environments the IP address might be required.
      - An incorrect IMS Port Number was specified for the target IMS Connect (for TCP/IP communication).
      - IMS Datastore name is invalid for the target IMS or is misspelled. Datastore name must be in all uppercase characters.
    - IMS is not running.
    - IMS Connect is not running.
    - The IMS Connect port is not active. Use the IMS Connect command **VIEWHWS** to determine if the port is active. Use the IMS Connect command **OPENPORT** to activate an IMS Connect port.
    - The target IMS data store is not active. Use the IMS Connect command **VIEWHWS** to determine if the data store is active. Use the IMS Connect command **OPENDS** to activate a IMS data store.
    - TCP/IP failure. Always ensure a successful ping to your IMS environment prior to running the IVP.
  - User action: Check the SOAP Gateway messages and codes for the error message to perform the appropriate user action.

Related tasks:

"Configuring IMS Connect for SOAP Gateway" on page 101
You must configure IMS Connect to allow SOAP Gateway to access IMS transactions.

"Verifying the installation of SOAP Gateway" on page 102
To verify the installation of SOAP Gateway, use the SOAP Gateway Installation Verification Program (IVP).

Configuring for diagnostic error messages from Rational Developer for System z

For the Enterprise Service Tools (EST) in Rational Developer for System z to issue diagnostic error messages, add the SFEKLMOD module to the STEPLIB in the IMS Connect startup JCL.

The code generator in EST issues diagnostic IRZ messages. If the load module is unavailable and an error is encountered during the code generation process, the following message is issued:

```
IRZ9999S Failed to retrieve the text of a Language Environment runtime
message. Check that the Language Environment runtime message module for
facility IRZ is installed in DFHRPL or STEPLIB.
```

This error indicates that the SFEKLMOD module is not configured in the STEPLIB in the IMS Connect startup JCL. After the module is added, specific IRZ messages are issued with more detailed diagnostic information. The IRZ messages are described in the Rational Developer for System z information center.

The *Rational Developer for System z Host Configuration Guide* provides host configuration information.

Related information:

IRZ messages (in Rational Developer for System z V9 information center)
For more information about IRZ messages, see the Rational Developer for System z V9 information center.

Rational Developer for System z Host Configuration Guide (in Rational Developer for System z library)
For more information about diagnostic IRZ error messages for Rational Developer for System z, see Chapter 6 in the Rational Developer for System z Host Configuration Guide.

Keystore import errors

If you receive an IO exception when you import a certificate to a keystore or truststore, the JDK you use might not be compatible with the version that is included with SOAP Gateway.

Explanation

If the exception is:

```
java.io.IOException: DerInputStream.getLength(): lengthTag=127, too big
```

The certificate is not created in the exact format that the JDK you use to import the certification expects.

User action

Use the ikeyman tool that is included in *IBMJava*/jre/bin/ directory to generate and import the certificate.

Diagnosing runtime errors

Informational, error, and unrecoverable (fatal) messages are logged to assist diagnosis of runtime problems during server startup and request and response processing.

SOAP Gateway logs the messages in the following locations:

- Windows For Windows only: SOAP Gateway Server console window
- z/OS For z/OS only: the z/OS syslog
- z/OS Linux Windows The SOAP Gateway server log file, `imssoap.log`.

The log file rolls over on a daily basis. An `imssoap.log.`*`timestamp`* file is created each day for the previous day.

Messages are logged in the `imssoap.log` file based on the trace level setting. The default trace level is ERROR, and both error and fatal (unrecoverable) messages are logged.

You cannot turn off the server log file on platforms other than the z/OS. On the z/OS platform, by default, WTO logging is enabled and error and fatal messages are sent to the z/OS syslog. If you do not need additional messages, and do not

want a separate server log file that contains the same information, set the trace level is set to 0 to disable the log file. If the SOAP Gateway server is not running, fatal messages are sent to the spool.

SOAP Gateway also provides a snapshot of the callout thread pool cache to assist troubleshooting. When you use the SOAP Gateway management utility to request to print the callout thread pool cache, a `threadPool.cache.`*`timestamp`* file is generated.

The log file and thread pool cache snapshot file are stored in the *`install_dir`*`/imsbase/logs` directory. After installation, the location for the log file and for the thread pool cache snapshot file can be configured separately by using the SOAP Gateway management utility.

EZD messages and AT-TLS return codes

You might encounter the EZD message in syslog or in TCP/IP joblog that report any errors that occur on an AT-TLS connection

The EZD1286I message is issued to syslog for AT-TLS connection issues when the trace level 2 (Error) is set. EZD1287I is issued to the TCP/IP job log to report any errors that occur on an AT-TLS connection when the trace level 1 (Error) is set. These messages include the event that AT-TLS was processing and a return code. Return codes 5001 - 5999 describe AT-TLS errors that can be corrected by the user. Return codes 6001 - 6999 describe internal AT-TLS errors.

See the AT-TLS return codes information in the *z/OS V1R11 Communications Server IP Diagnosis Guide*.

Related reference:

Refer to the z/OS V1R13.0 Communications Server IP Diagnosis Guide for AT-TLS return codes.
For more detail about AT-TLS return codes and other troubleshooting information, see the z/OS V1R13.0 Communications Server IP Diagnosis Guide.

Setting the trace level for z/OS Communications Server AT-TLS feature

Use the IBM Configuration Assistant for z/OS Communications Server V1R13 to specify the trace level to help troubleshoot potential issues.

The following steps are based on the V1R11 of the IBM Configuration Assistant for z/OS Communications Server V1R13. You must have configured AT-TLS for server authentication.

1. In the Configuration Assistant, go to the AT-TLS perspective by clicking **Perspective** > **AT-TLS**.
2. Click the TCP/IP stack of your z/OS image.
3. Click the connectivity rule for SOAP Gateway, and click **Modify**.
4. Click the traffic descriptor to modify, and then click **Settings**.
5. In the **Tracing** tab, click Log only the selected trace levels, and select the appropriate tracing levels, such as Level 1, level 2, and level 4.
6. Click **OK** multiple times until you are back to the TCP/IP stack information in the AT-TLS perspective.
7. Click **Apply Changes**.

View the configuration file by right-clicking your image, selecting **Install Configuration File**, selecting the stack for your z/OS image, and clicking **Show Configuration File**.

The generated configuration file shows the new value for the Trace parameter of the TTLSConnectionAction policy statement. The value is the sum of the levels that you selected.

```
TTLSConnectionAction              cAct1~IMS_SOAP_traffic_desc
{
  HandshakeRole                   Server
  TTLSConnectionAdvancedParmsRef  cAdv1~IMS_SOAP_traffic_desc
  CtraceClearText                 Off
  Trace                           7
}
```

Related reference:

Refer to the z/OS V1R13.0 Communications Server IP Diagnosis Guide for AT-TLS return codes.
For more detail about AT-TLS return codes and other troubleshooting information, see the z/OS V1R13.0 Communications Server IP Diagnosis Guide.

Setting up for WS-Security tracing

To help debug WS-Security related issues, update the file `logging.properties` in the *install_dir*`/java/jre/lib` directory and turn on the FileHandler to log messages to a file.

The file `logging.properties` contains properties that let you specify the logging level, the maximum size of the log file, and the number of log files allowed. Logging is handled by handler classes. By default, only ConsoleHandler is turned on, which displays messages to the console, not to a file.

To set up for logging to one or more log files for tracing:

1. Open the file `logging.properties` in *install_dir*`/java/jre/lib`.
2. Comment out the following line for handlers.

   ```
   #handlers= java.util.logging.ConsoleHandler
   ```

3. Uncomment the following line for handlers, so both FileHandler and ConsoleHandler are turned on.

   ```
   handlers= java.util.logging.FileHandler, java.util.logging.ConsoleHandler
   ```

4. Append the following two lines at the end of the file to log all levels of messages.

   ```
   com.ibm.ws.wssecurity.level = ALL
   com.ibm.ws.policyset.level = ALL
   ```

5. Optionally, you can specify the maximum size per log file, and the maximum number of log files. For example:
   - Change the value for java.util.logging.FileHandler.limit to 20000000 would allow up to 20 MB per log file.

     ```
     java.util.logging.FileHandler.limit = 20000000
     ```

   - Change the java.util.loggin.FileHandler.count to 10 would allow up to 10 log files.

     ```
     java.util.logging.FileHandler.count = 10
     ```

6. Optionally, you can specify the location of the log file by modifying this line:

   ```
   # default file output is in user's home directory.
   java.util.logging.FileHandler.pattern = %h/java%u.log
   ```

CWWSS messages for WS-Security related errors

CWWWS messages indicate issues related to WS-Security.

CWWSS5205E

The CWWSS5205E message indicates that the timestamp has expired.

```
CWWSS5205E: The time stamp in the message has expired.
```

Check to see if you need to adjust the WS-Security timeout values. For SAML tokens, timestamp values can be set by using an assertion named Conditions, which has two attributes, NotBefore and NotOnOrAfter. For both SAML and username tokens, you can use the web service utility (wsu) to specify the message creation time and expiration time. See the topic on setting the timeout value for WS-Security elements for more information.

CWWSS5502E

The CWWSS5502E message indicates that a user name token was sent instead of a SAML token:

```
CWWSS5502E: The target element: wsse:UsernameToken was not expected.
```

Modify the client application to send the correct type of security token to the SOAP Gateway server.

CWWSS5509E

The CWWSS5509E message indicates that a token is missing.

```
CWWSS5509E: A security token whose type is [http://docs.oasis-open.org/wss/
oasis-wss-saml-token-profile-1.1#SAMLV1.1] is required.
```

Modify the client application to send the correct type of security token to the SOAP Gateway server.

CWWSS5514E

The CWWSS5514E message might follow the CWWSS6901E message if the security token already expires.

```
CWWSS5514E: An exception while processing WS-Security message.
```

Check to see if you need to adjust the WS-Security timeout values. If you use the assertion named Conditions, check the values for the NotBefore and NotOnOrAfter attributes. For both SAML and username tokens, you can use the web service utility (wsu) to specify the message creation time and expiration time. See the topic on setting the timeout value for WS-Security elements for more information.

CWWSS6521E

The CWWSS6521E message indicates that the specified custom authentication module class file or JAR file is not the correct file or is not present.

CWWSS6521E: The Login failed because of an exception:
javax.security.auth.login.LoginException: No LoginModules configured for
system.wss.consume.saml*xx*

Check that the SAML token module is configured correctly in the wsjaas.conf file in the *install_dir*/imssoap/WEB-INF directory. Store your compiled custom module .class file in the *install_dir*/imssoap/WEB-INF/classes directory. For JAR files that contain classes files, store them in the *install_dir*/imssoap/WEB-INF/lib directory.

CWWSS6901E

When a request to access a web service is received, an error occurs in the SOAP Gateway server log: Nov 6, 2009 6:26:49 PM
com.ibm.ws.wssecurity.util.LoggerTraceImpl log SEVERE: CWWSS6901E: The
Application Server cannot load the configuration file for the security
token service.

This error might occur even when web services security (WS-Security) is not enabled for a web service.

This error message indicates that some metadata that is required for an embedded security code is not available. This metadata is not used by SOAP Gateway, so this error can be safely ignored. No action is needed.

Related tasks:

"Plugging in a custom authentication module" on page 170
Append your custom authentication module to the default authentication module in the corresponding UsernameToken or SAML entries in the wsjaas.conf file in the SOAP Gateway installation.

"Setting the timeout value for WS-Security enabled messages" on page 166
You can specify timestamp validation information for WS-Security tokens so that the SOAP Gateway sever can decide if the data has become stale and the message needs to be discarded.

General runtime errors

You might get an runtime error during the server startup process or when a request tries to access a web service that is deployed on SOAP Gateway.

This section describes some general runtime errors you might encounter and the actions to take.

Endpoint initialization error

When you start SOAP Gateway, you receive the error: Error initializing endpoint java.net.BindException: The socket name is already in use.: *port_number*.

This error appears in the SOAP Gateway console window (for Windows only) or in the log file.

Explanation

SOAP Gateway failed to start because the port number is already in use.

Possible causes

- Another instance of SOAP Gateway has been started.

- If another instance of SOAP Gateway is not already started, another program on the machine is using the port number.

User action

- Verify that another instance of SOAP Gateway is not already started by looking at the SOAP Gateway log or the Administrative Console. The Administrative Console can only be started if SOAP Gateway is up and running.
- If several programs share the same port number, you must change the port number value for SOAP Gateway. Use the SOAP Gateway management utility `iogmgmt -prop -u -p` *`port_number`* command to update the port number.
- z/OS Check the dedicated shutdown port number in addition to the server listening port number. Each instance of the SOAP Gateway server must have its own unique shutdown port. Use the `iogmgmt -view -sgp` command to check the port number and the`iogmgmt -prop -u -d` *`port`* command to change it if necessary.

Connection errors with SOAP requests

You receive a connection error when the client application submits a SOAP request to SOAP Gateway.

Explanation

The client is unable to communicate with SOAP Gateway.

Possible causes

- SOAP Gateway has not been started.
- The client application is using the wrong host name and port number for the URL that is used to access SOAP Gateway. This error can occur when you change the host name and port number of SOAP Gateway.

User action

- Verify that SOAP Gateway is started by viewing the SOAP Gateway log or the Administrative Console. Because you can view the Administrative Console only when SOAP Gateway is started, this action is a way to quickly test if SOAP Gateway is up and running.
- Ensure that your client application is using the correct host name and port number for the URL that is used to access SOAP Gateway.

Keystore errors during startup

You receive a keystore or keystore password error when the server starts up, which prevents the SOAP Gateway from running.

Explanation

The server is unable to initialize the HTTPS port because one of the properties is incorrect.

Possible causes

Possible cause of this error:

- The HTTPS keystore name is incorrect or the keystore specified does not exist.
- The HTTPS keystore password is incorrect, meaning it cannot open the specified keystore file.

- The HTTPS port number is not valid.

User action

- Ensure that the Keystore specified for HTTPS is valid and exists at the specified location. Ensure that the keystore name includes the full path to the keystore file.
- Ensure that the keystore password specified is correct and can be used to open the keystore file.
- Ensure that the HTTPS port specified is valid for the machine in which the SOAP Gateway is running. Try a different port if the error persists.
- If HTTPS is not needed, turn off server authentication or client authentication by using the SOAP Gateway management utility iogmgmt -prop -u -serverauth false or iogmgmt -prop -u -clientauth false command.
- Restart the server.

Diagnosing issues with callout and business event requests

During the callout process and the response return process, if an error occurs, it could occur in the callout request or response processing in IMS Connect, SOAP Gateway, or the web service.

You can print the thread pool log cache to trace the threads for troubleshooting issues related to callout message processing.

Tip: Business event requests are basically the same as asynchronous callout requests. The same troubleshooting techniques for diagnosing asynchronous callout requests apply to business event requests.

The following scenarios describe where the errors could occur and how these errors are reported:

- IMS Connect fails to process the callout request message.
  - If IMS Connect cannot process the callout request message because, for example, the message is corrupted. IMS Connect displays a HWSP1510E message in the SOAP Gateway console window to indicate that the message was rerouted to the dead letter queue (HWS$DLQ).
  - If IMS Connect fails to call the specified adapter to process the message because, for example, the adapter name is invalid, IMS Connect issues a HWSA0340E message, and the message is rerouted to the dead letter queue.
  - If the XML converter is not found or a converter is not specified, IMS Connect reports an error, and the message is rerouted to the dead letter queue.
  - If the XML converter fails to process the message, IMS Connect reports an error, and the message is rerouted to the dead letter queue.
- SOAP Gateway fails to process the callout request message.

  This problem might occur because of the following reasons:
  - The message is missing a web service identifier (WSID) or the WSID value is not valid.
  - The service name and operation name are missing or not valid.
  - The data is not in XML format.
  - The callout web service information is not configured.

In this scenario, the message is rerouted to the dead letter queue (HWS$DLQ), and an error message is logged in the SOAP Gateway error log and the SOAP Gateway console window.

- The web service returns a SOAP fault message to SOAP Gateway.

  SOAP Gateway logs the web service fault message in its error log.

  Check the web service or business event server logs or documentation for troubleshooting information. For example, for WebSphere Business Monitor, refer to the WebSphere Business Monitor troubleshooting information for the following error scenarios:

  – The event does not show in WebSphere Business Monitor Dashboard. See the Dashboard troubleshooting information.
  – The event is reported to have failed. See the topic on examining a failed event.
  – You cannot deploy the monitor model. See the lifecycle troubleshooting information.
- IMS Connect is not available when SOAP Gateway sends it the response from the web service.

  SOAP Gateway logs the error message in its error log.

In addition, if a response message is expected, and a response mode transaction is invoked to process the response message, the output of the response mode transaction is routed to the IOG$RESP tpipe if you need to troubleshoot or trace callout response messages.

Error status code returned to IMS during synchronous callout requests processing

When SOAP Gateway encounters errors during synchronous callout request processing, the status code that are returned are in the range of 1000 to 2000.

When SOAP Gateway returns a status code when processing a callout request, the message has the following format:

`SOAP Gateway Error: status code: error response.`

The status codes and their error messages are listed in the following table.

Table 37. Status codes from SOAP Gateway and their messages

Status code	Message
1001	No callout response was returned from the web service.
1002	Error occurred when SOAP Gateway invoked the web service.
1003	Error occurred when SOAP Gateway sent the callout response.
1004	Error occurred when SOAP Gateway received the callout message (regular NAK - discard).
1005	Error occurred when SOAP Gateway received the callout message (NAK with reroute - continue).
1006	Error occurred when SOAP Gateway parsed the callout message (regular NAK - discard).
1007	Error occurred when SOAP Gateway parsed the callout message (NAK with reroute - continue).
1008	Error occurred when SOAP Gateway sent a special NAK.

Table 37. Status codes from SOAP Gateway and their messages (continued)

Status code	Message
1009	`Network error occurred when invoking web service.` This status code is followed by an IOGS0077E error message about a network error that occurred during the invocation of the external web service.

Troubleshooting callout message processing and the thread pool

You can print the status of callout threads in the thread pool cache to help troubleshooting issues related to callout message processing.

To print the status of threads in the thread pool cache:

1. Issue the following SOAP Gateway management utility command:

   ```
   iogmgmt -view -workerthreads
   ```

   A log of the thread processing status is generated.
2. Open the generated file and examine the thread information.

For distributed platforms, if the server log file location is not explicitly specified, the files are stored in the *install_dir*/imsbase/logs. See the topic on configuring the log file location as a post-installation configuration step.

To change the cache buffer size, set the queue throttle length property by using the iogmgmt -callout -updateprop command. You are prompted to specify, for each property, a new value.

Tip:

- The number of messages to be cached is determined by the threadPoolCacheCapacity property, and can be configured by using the SOAP Gateway management utility.
- In general, a larger buffer size is better for debugging problems.

Related tasks:

"Configuring the SOAP Gateway log file location" on page 95
By default, the log files are written to the *install_dir*/imsbase/logs directory. To change the log file location, use the SOAP Gateway management utility.

Troubleshooting performance issues

If request processing time seems to slow down significantly or the server runs out of memory and crashes, you might need to adjust the thread management-related properties or the Java memory heap size.

If you encounter errors such as:

```
JVMDUMP039I Processing dump event "systhrow", detail "java/lang/OutOfMemoryError"
at 2013/03/08 23:53:40 - please wait.
```

or

```
WARNING: Encountered a failure in the fireAlarm method java.lang.OutOfMemoryError:
Failed to create a thread: retVal -1073741830
```

the server does not have enough native memory to create threads, and the server could eventually crash.

For the callout scenario, you might need to adjust the values for the numberOfWorkerThreadsInPool and queueThrottleLength callout property values to increase callout message processing concurrency.

The maxThread property in the `server.xml` file is for inbound requests to SOAP Gateway and has no impact on callout threads. However, its value can be reduced to 50 or 75 from the default value to allow for more memory for native thread creation.

The other properties in the `server.xml` file that might affect performance include acceptCount, connectionTimeout, maxKeepAliveRequests, and acceptorThreadCount. SOAP Gateway management utility does not support configuration of these properties, and care must be taken if you choose to manually modify their values. For z/OS systems, keep in mind that this file is in UTF-8 encoding. If you download the file, make changes on a local workstation, and upload it back to the z/OS system, ensure that the file is transferred in binary mode. For more information about these properties, refer to Apache Tomcat documentation.

The Java heap size setting might need to be adjusted to allow for more threads to be created. You can start by setting the initial heap size to 50% of the maximum heap size, monitor the thread activities, and adjust the value accordingly.

Related concepts:

"Thread management and configuration considerations" on page 180
Depending on the work load in your environment, you can tune SOAP Gateway to maximize performance and throughput of the callout processing by configuring the SOAP Gateway callout properties.

Related reference:

"Callout properties for thread management" on page 179
SOAP Gateway provides several properties for configuring and managing the threads.

Related information:

Memory management and heap sizing for IBM SDK for z/OS, Java Technology Edition, Version 7

Memory management and heap sizing for IBM SDK for Windows, Java Technology Edition, Version 7

Memory management and heap sizing for IBM SDK for Linux, Java Technology Edition, Version 7

Apache Tomcat 7 HTTP Connector documentation

Messages for SOAP Gateway

SOAP Gateway returns messages with the prefixes IOGC, IOGM, IOGS, and IOGX.

IOG messages

IOG messages are related to the startup and shutdown status of the SOAP server.

IOG0001I *server_start_informational_message***.**

Explanation: SOAP Gateway sends this information message to the console to signal that a server start has been issued to the SOAP Gateway server. The *server_start_informational_message*can be one of the following messages to reflect the startup status:

- The server starts up in *startup_time* nanoseconds.
- The SOAP Gateway server is now up and running.

startup_time is the number of nanoseconds it takes to start up the server.

User response: No action is required.

IOG0002I *server_stop_informational_message***.**

Explanation: SOAP Gateway sends this information message to the console to signal that a server stop has been issued to the SOAP Gateway server. The *server_stop_informational_message* can be one of the following messages to reflect the shutdown status:

- Stopping all callout threads on SOAP Gateway.
- Stopping the thread pool on SOAP Gateway.
- The SOAP Gateway server is stopped.

User response: No action is required.

IOG0003E *server_start_error_message***.**

Explanation: SOAP Gateway sends this error message to the console to signal that an error occurred during the SOAP Gateway server startup process. The [*server_start_error_message*] can be one of the following messages to reflect the startup status:

- Catalina.start: *Error_thrown*.
- The SOAP Gateway server cannot be started. *Error_thrown.*

Error_thrown is the error thrown by the Java Runtime Environment (JRE) that explains the cause of the failure.

User response: Use the Java error information to identify the cause.

IOG0005E *server_stop_error_message***.**

Explanation: SOAP Gateway sends this error message to the console to signal that an error occurred during the SOAP Gateway server shutdown process. The *server_stop_error_message* can be one of the following messages to reflect the startup status:

- Catalina.stop: *Error_thrown*.
- The SOAP Gateway server cannot be stopped. *Error_thrown*.

Error_thrown is the error thrown by the Java Runtime Environment (JRE) that explains the cause of the failure.

User response: Use the Java error information to identify the cause.

IOG00000E SOAP Gateway has detected a Java stack trace useful for technical support. *Error_details.*

Explanation: Possible error details are:

- The stack trace is written to the log.
- The stack trace cannot be written to the log because the logfile appender is set to OFF. Turn on the logfile appender and try to recreate the problem so the stack trace could be written to the log.
- The stack trace is: *stack_trace_details*

User response: If the stack trace is written to the log, examine the `imssoap.log` file in the *`install_dir`*`/imsbase/logs` directory.

If the stack trace is not written to the log, use the iogmgmt -prop command to set the trace level to 2 (ERROR) or above:

```
iogmgmt -prop -u -l 5
```

IOG00003E The SOAP Gateway server cannot be started.

Explanation: The SOAP Gateway server cannot be started.

User response: Check the `imssoap.log` file in the *install_dir*`/imsbase/logs` directory. If the messages are logged to the z/OS syslog only (that is, the log file is disabled), review the syslog, and consider setting the trace level to 2 or above. Use the iogmgmt -prop command to set the trace level:

```
iogmgmt -prop -u -l 5
```

In addition, check that the SOAP Gateway installation directory has not been manually altered or moved.

IOG00005E The SOAP Gateway server cannot be correctly stopped.

Explanation: SOAP Gateway tries to stop the server in 60 seconds.

User response: Check the `imssoap.log` file in the *install_dir*`/imsbase/logs` directory. If the messages are logged to the z/OS syslog only (that is, the log file is disabled), review the syslog, and consider setting the trace level to 2 or above. Use the iogmgmt -prop command to set the trace level:

```
iogmgmt -prop -u -l 5
```

In addition, check that the SOAP Gateway installation directory has not been manually altered or moved.

If using the iogmgmt -stop command does not stop the server, you can use the iogmgmt -stop -force command to stop the server immediately and discard all work in progress. This command works only on Linux on System z, and only when the server was started by using the iogmgmt -start command.

IOG00007E *Exception_type*. **The (***path_to_the_file***) file is empty, invalid, in wrong encoding, or not found.**

Explanation: The exception type is most likely an I/O Exception. The file that is reported cannot be accessed or found.

User response: Verify that the file does exist, is valid, and is in UTF-8 encoding.

IOG00008E *Exception_type*. **SOAP Gateway is unable to read the port value in the (***path_to_the_file***) file.**

Explanation: The exception type is most likely an I/O Exception. The server port information cannot be determined because the server properties file is either not valid or in incorrect encoding.

User response: Verify that the file is valid, contains the port information, and is in UTF-8 encoding. Restart the server.

IOG00009E The SOAP Gateway server is already in use on *port_list*. **Check that the reported port or ports are not reserved for other applications or used by other instances of the SOAP Gateway server.**

Explanation: The server could not start because one of the configured ports (*port_list*) is already in use.

User response: Resolve the port conflict by changing the configured ports for the SOAP Gateway server with the `iogmgmt -prop -u -p` command (for the server listening port) or the `iogmgmt -prop -u -d` command (for the dedicated server shutdown port on z/OS) or for the conflicting application.

IOG00010E The Java directory cannot be updated based on the current setting: JAVA_HOME=*directory*. **Error=***details*.

Explanation: The properties file containing the location of the IBM SDK for Java Technology was not found. This error indicates that the internal library files used by the SOAP Gateway have been deleted, renamed, or moved.

User response: Reinstall SOAP Gateway.

IOG00011E The SOAP Gateway installation home directory is not found based on the current setting: IMSSOAP_HOME=*directory*.

Explanation: The properties file containing the location of the SOAP Gateway home directory was not found. This error occurs when the directory *install_dir*/imssoap does not exist.

User response: Reinstall SOAP Gateway.

IOG00012E The SOAP Gateway server directory is not found based on the current setting: IMSSOAP_DIR=*directory*.

Explanation: The SOAP Gateway server directory does not exist. This error occurs only if the file system is modified after installation to remove the server root installation directory.

User response: Reinstall SOAP Gateway.

IOG00013E Java *installed_SDK_version* is not supported. Use the iogmgmt command update the Java path to point to the supported version *required_SDK_version*.

Explanation: The installed version of Java is not supported by this version of SOAP Gateway.

User response: Use the `iogmgmt -prop -u -java -h` *file_path* command to select the directory of a compatible IBM SDK for Java Technology.

- Windows Linux The IBM SDK for Java Technologies is installed with IBM Installation Manager for SOAP Gateway.
- z/OS The IBM SDK for Java Technologies is installed as part of the IMS Enterprise Suite base services component.

IOG00014E SOAP Gateway was unable to detect the version of Java specified in *directory*. Use the iogmgmt command to update Java to the supported version *required_SDK_version*

Explanation: The installed version of the IBM SDK for Java Technology is not supported by this version of SOAP Gateway because the version details could not be read.

User response: Use the `iogmgmt -prop -u -java -h` *file_path* command to select the directory of a compatible IBM SDK for the server.

- Windows Linux The IBM SDK is installed with IBM Installation Manager for SOAP Gateway.
- z/OS The IBM SDK is installed as part of the IMS Enterprise Suite base services.

IOG00015E SOAP Gateway has detected that Java has not been configured. Use the iogmgmt command to specify the location for the supported version *required_SDK_version*.

Explanation: No IBM SDK for Java Technology home directory is set. A supported version of the IBM SDK is required to start SOAP Gateway.

User response: Use the `iogmgmt -prop -u -java -h` *file_path* command to select the directory of a compatible IBM SDK for the server.

- Windows Linux The IBM SDK is installed with IBM Installation Manager for SOAP Gateway.
- z/OS The IBM SDK is installed as part of the IMS Enterprise Suite base services.

IOG00016E SOAP Gateway requires the *required_SDK_version* distribution of Java. Use the iogmgmt command to update the location for the supported distribution.

Explanation: The currently configured distribution of Java is not supported. SOAP Gateway requires the indicated version of the IBM SDK for Java Technology.

User response: Use the `iogmgmt -prop -u -java -h` *file_path* command to select the directory of a compatible SDK for the server.

- Windows Linux The IBM SDK is installed with IBM Installation Manager for SOAP Gateway.

- z/OS The IBM SDK is installed as part of the IMS Enterprise Suite base services.

IOG00017E The path to Java can not contain spaces. The current JAVA_HOME is set to *directory*.

Explanation: z/OS Linux
The path to the Java home directory cannot contain spaces. Spaces are only allowed on Windows systems.

User response: Use the `iogmgmt -prop -u -java -h file_path` command to select the directory of a compatible IBM SDK for Java Technology for the server that does not contain spaces.

IOG00018E The command failed because the specified *directory_name* directory does not exist.

Explanation: The specified Java home directory does not exist or was not reachable.

User response: Update the Java home directory with the `iogmgmt -prop -u -java -h` command, or create a file system link to the target directory.

IOG00019E SOAP Gateway must be configured with the Java Development Kit (JDK). Configuring SOAP Gateway with the Java Runtime Environment (JRE) only is not supported.

Explanation: The SOAP Gateway server requires the complete IBM SDK for Java Technology.

User response: Install the IBM SDK for Java Technology appropriate for the platform where you are installing the server. Update the Java home directory for the server with the `iogmgmt -prop -u -java -h` command.

IOG00020E SOAP Gateway could not load the file *file_name*. The file might not have the correct permission, or the file does not exist.

Explanation: An internal SOAP Gateway configuration or library file is missing or corrupted.

User response: Reinstall SOAP Gateway.

IOG00021E The SOAP Gateway Windows service IMSSOAPGateway failed to install because the executable was not found. Reinstall SOAP Gateway using Installation Manager.

Explanation: The server executable is not found during the attempt to install the SOAP Gateway Windows service. The executable might have been renamed, deleted, or moved.

User response: Reinstall SOAP Gateway using the IBM Installation Manager.

IOG00022E The SOAP Gateway Windows service IMSSOAPGateway could not be removed. The specified service does not exist.

Explanation: The SOAP Gateway Windows service could not be removed because it is not installed on the system.

User response: No user action is needed.

IOG00023E The SOAP Gateway Windows service IMSSOAPGateway failed to start because the executable was not found. Reinstall SOAP Gateway using Installation Manager.

Explanation: The server executable is not found during the attempt to start the SOAP Gateway Windows service. The executable might have been renamed, deleted, or moved.

User response: Reinstall SOAP Gateway using the IBM Installation Manager.

IOG00024E The SOAP Gateway Windows service IMSSOAPGateway failed to install. Check that you have sufficient privileges.

Explanation: The Windows service could not be installed.

User response: Check if you have sufficient system administrative right to install Windows services.

IOG00025E The SOAP Gateway Windows service IMSSOAPGateway failed to install. IMSSOAPGateway already exists as an installed service.

Explanation: The SOAP Gateway Windows service is already installed and could not be reinstalled.

User response: No user action is needed.

IOG00026E The SOAP Gateway Windows service IMSSOAPGateway failed to start. IMSSOAPGateway is not installed on the system.

Explanation: The SOAP Gateway Windows service could not be started because it is not yet installed.

User response: See "Installing SOAP Gateway as a Windows service" on page 94 for more information about how to install SOAP Gateway as a Windows service.

IOG00027E The SOAP Gateway Windows service IMSSOAPGateway failed to stop. IMSSOAPGateway is not installed on the system.

Explanation: The SOAP Gateway Windows service could not be stopped because it has not been installed.

User response: No user action is required.

IOG00028E The SOAP Gateway Windows service IMSSOAPGateway failed to start. The SOAP Gateway server is already running as a terminal application.

Explanation: The SOAP Gateway Windows service could not be started because SOAP Gateway is already running as a terminal application.

User response: Stop the SOAP Gateway instance that is running in terminal mode before starting the SOAP Gateway Windows service.

IOG00029E The SOAP Gateway imsbase directory is not found based on the current setting: SG_BASE_DIR=*imsbase_install_dir*. The imsbase directory is either not installed, deleted, or altered. Correct the directory name, or reinstall SOAP Gateway.

Explanation: You are trying to issue a SOAP Gateway management utility command, but the required imsbase directory is not found. Either the directory is deleted, moved, or renamed, or it is never installed. Missing to install the imsbase component could happen on the z/OS platform when the provided AEWTSGIN JCL sample job was not modified correctly and the imsbase component was omitted.

User response: Correct the directory name or restore the imsbase directory to the correct location. If the imsbase component is never installed, you can install the missing component without reinstalling the others. However, ensure that you have the path information for all three components correctly defined in AEWIOGCF in order for SOAP Gateway to run properly.

IOG00030E The SOAP Gateway imssoap directory is not found based on the current setting: SG_SOAP_DIR=*imssoap_install_dir*. The imssoap directory is either not installed, deleted, or altered. Correct the directory name, or reinstall SOAP Gateway.

Explanation: You are trying to issue a SOAP Gateway management utility command, but the required imssoap directory is not found. Either the directory is deleted, moved, or renamed, or it is never installed. Missing to install the imssoap component could happen on the z/OS platform when the provided AEWTSGIN JCL sample job was not modified correctly and the imsbase component was omitted.

User response: Correct the directory name or restore the imssoap directory to the correct location. If the imssoap component is never installed, you can modify the AEWTSGIN job to install the missing component without reinstalling the others. However, ensure that you have the path information for all three components correctly defined in AEWIOGCF in order for SOAP Gateway to run properly.

IOG00031E The -migrate command requires one argument. Specify either the correlator keyword for correlator migration, or the source SOAP Gateway installation location for server and web services migration.

Explanation: The iogmgmt -migrate command was issued without a required argument.

User response: Reissue this command with either the correlator keyword or the source SOAP Gateway installation location. Specify the source SOAP Gateway installation location to migrate server properties, web services, and correlator files from an older release. If only correlator files need to be migrated, specify the correlator keyword to migrate correlator files from version 1.0 to version 2.0 of the correlator XML schema. If the correlator files that are generated by older versions of Rational Developer for System z are of version 1.0, the files must be migrated to version 2.0.

Related reference:

"-migrate: Migrate and upgrade SOAP Gateway" on page 448
The -migrate command upgrades SOAP Gateway artifacts and settings to the latest version and generates a migration log.

IOG30001I The SOAP Gateway server is now up and running. Elapsed time is *startup_time* nano seconds

Explanation: *startup_time* is the number of nanoseconds it takes to start up the server.

User response: No action is required.

IOG30002I Stopping all resume tpipe threads on SOAP Gateway.

Explanation: SOAP Gateway is stopping all resume tpipe threads. When a thread receives the stop command, it issues its own informational message. The informational message contains the unique thread ID. For the one-thread-per-tpipe thread policy, the thread ID contains the connection bundle and the tpipe name. For one-thread-per-connection-bundle thread policy, the thread ID contains the connection bundle name.

User response: No action is required.

IOG30003I Stopping the thread pool on SOAP Gateway.

Explanation: SOAP Gateway is stopping the thread pool. When the worker threads receive the stop command, each thread issues its own informational message.

User response: No action is required.

IOG30004I The SOAP Gateway server is stopped.

Explanation: SOAP Gateway is stopped successfully.

User response: No action is required.

IOG30005I The SOAP Gateway server is now up and running.

Explanation: This message is logged when SOAP Gateway is started successfully by using the SOAP Gateway management utility iogmgmt -start command.

User response: For more information that is logged during the server startup, view the SOAP Gateway log.

IOG30006I Checking on server status.

Explanation: This message is informational only.

User response: No action is required.

IOG30008I The Java directory has been updated to *directory*.

Explanation: The IBM SDK for Java Technology home directory for the server was successfully updated.

User response: No action is required.

IOG30009I The SOAP Gateway Windows service IMSSOAPGateway was successfully removed.

Explanation: This is an informational message logged when the SOAP Gateway Windows service is removed.

User response: No action is required.

IOG30010I The SOAP Gateway Windows service IMSSOAPGateway was successfully installed.

Explanation: This is an informational message logged when the SOAP Gateway Windows service is installed on the system.

User response: No action is required.

IOG30011I The SOAP Gateway Windows service IMSSOAPGateway could not be stopped. IMSSOAPGateway is not started.

Explanation: This is an informational message when the command to stop the SOAP Gateway Windows service failed because the service has not been started or is not running.

User response: No action is required.

IOG30012I The SOAP Gateway Windows service IMSSOAPGateway could not be started. IMSSOAPGateway is already running.

Explanation: This is an informational message when the command to start the SOAP Gateway Windows service failed because the service is already started or running.

User response: No action is required.

IOG30013I The SOAP Gateway Windows service IMSSOAPGateway is now up and running.

Explanation: This is an informational message to indicate that the SOAP Gateway Windows service has successfully started.

User response: No action is required.

IOG30014I The SOAP Gateway Windows service IMSSOAPGateway is stopped.

Explanation: This is an informational message to indicate that the SOAP Gateway Windows service has successfully stopped

User response: No action is required.

IOG30015I The SOAP Gateway Windows service executable was not found. The IMSSOAPGateway service was removed manually.

Explanation: The IMSSOAPGateway service was removed from the registry. However, the executable was not found during this operation most likely because the executable is renamed, deleted, or missing.

User response: Reinstall SOAP Gateway using the Installation Manager.

IOG30016I Graceful shutdown of the server is in progress. All incoming messages are blocked.

Explanation: The graceful shutdown of the server has been initiated. Any subsequent new requests will be rejected by the server and receive an IOGS0125E message.

User response: This message is for informational purpose only.

IOG30017I All callout threads and the thread pool are stopped.

Explanation: This message is for informational purpose only.

User response: No action is needed.

IOG30018I The server is processing in-flight messages. The server will wait for a maximum of 5 minutes for the processing before it forces the shutdown.

Explanation: This message is for informational purpose only. It is issued at most 3 times in the log.

User response: No action is needed.

IOG30019I In-flight statistics right after the graceful shutdown command was issued: *list_of_statistics*.

Explanation: This message provides the overall statistics for the messages that the server processed or is processing right after the graceful shutdown command was issued. The following numbers are reported:

- Number of provider requests received
- Number of provider responses processed
- Number of provider requests in error
- Number of provider requests rejected

If there are consumer (callout) or business event requests, the statistics are combined and the following numbers are reported:

- Number of consumer requests received
- Number of consumer responses processed
- Number of consumer requests in error

The reported numbers include only requests that have passed the required authentication and authorization, if web service or callout security is configured.

User response: No action is needed.

IOG30020I In-flight statistics right before the server was stopped: *list_of_statistics*.

Explanation: This message provides the overall statistics for the messages that the server processed right before the server was stopped. The following numbers are reported:

- Number of provider requests received
- Number of provider responses processed
- Number of provider requests in error
- Number of provider requests rejected

If there are consumer (callout) or business event requests, the statistics are combined and the following numbers are reported:

- Number of consumer requests received
- Number of consumer responses processed
- Number of consumer requests in error

The reported numbers include only requests that have passed the required authentication and authorization, if web service or callout security is configured.

User response: No action is needed.

IOG30022I Number of callout in-flight messages in the worker queue: *number_of_messages*.

Explanation: This message provides the number of in-flight messages that were in the internal work queue for callout at the time of graceful shutdown processing.

User response: No action is needed.

IOG30023I The server is shutting down by force after the 5-minute maximum wait time.

Explanation: This message is a notification that the server is shutting down after the 5-minute wait period, regardless of the status of in-flight messages processing.

User response: No action is needed.

IOG30024I The SOAP Gateway Windows service IMSSOAPGateway is up and running.

Explanation: This is an informational message to indicate that the SOAP Gateway Windows service is up and running when you check the status of the service.

User response: No action is needed.

IOG60001W SOAP Gateway supports only on or off for the IBM System z Application Assist Processor (zAAP) setting. The JAVA_IFA=*invalid_value* setting is not supported and is reset to off. Use the iogmgmt command to change the setting.

Explanation: The only valid values for `iogmgmt -prop -u -java -i value` are `on` and `off`. This value sets zAAP processing on or off for the server. The IFA setting was reverted to off because the specified value is invalid.

System action: The server continues to run with the Java IFA set to off.

User response: Reissue the command with a valid IFA setting and restart the server.

IOGC messages

IOGC messages are messages that result from errors from IMS or IMS Connect.

IOGC001E Failed to connect to IMS Connect. Hostname[*hostname*], port [*portnumber*]. [*error_message*].

Explanation: SOAP Gateway was unable to connect to the host and port combination. The *error_message* indicates the reason for the failure to connect.

User response: Examine the error message to determine the reason for the failure to connect to the host. Possible errors include:

UnknownHostException: *hostname*
The hostname that you specified in the connection bundle is invalid. You might have to use the fully qualified path for host name or the IP address.

ConnectException: Connection refused
Possible reasons:

- The port number is invalid. Ensure that you are using a valid port number for the IMS Connect indicated by hostname.
- The specified port is stopped. You can determine whether the port is stopped by using the IMS Connect command VIEWHWS. If the port is stopped, its status is NOT ACTIVE. Use the IMS Connect command OPENPORT *portnumber* to start the port.
- IMS Connect on the specified host is not running. Start IMS Connect on the host system.
- TCP/IP was restarted without canceling and restarting IMS Connect or issuing the command STOPPORT followed by OPENPORT on the host.

SocketException: Network is unreachable: connect
Possible reasons:

- The system with the specified host name is unreachable on the TCP/IP network. Ensure that the host system is accessible from the TCP/IP network by issuing the ping command to the specified host system from the system on which SOAP Gateway is running. Start TCP/IP on the host if it is not started.
- TCP/IP was restarted, but the status of the port that is used by the application was NOT ACTIVE. To fix the problem, you can do one of the following:
  - Use the IMS Connect command OPENPORT *portnumber* to start the port.
  - Restart IMS Connect.

IOGC002E Socket timeout value [*socket_timeout*] is invalid. [*error_message*].

Explanation: The value *socket_timeout* that was specified for the socket timeout property in the correlator file is not valid. *error_message* denotes the reason for the failure.

User response: One of possible errors is `IllegalArgumentException: timeout can't be negative`, which indicates that a negative value has been provide for the socket timeout value. Ensure a positive numerical value was given for the socket timeout in the correlator file.

IOGC003E SOAP Gateway cannot send or receive messages from IMS Connect. Hostname [*hostname*], port [*portnumber*]. [*error_message*].

Explanation: SOAP Gateway was unable to successfully complete a send or receive interaction with the target IMS Connect. *error_message* indicates the reason for the failure to complete the interaction.

User response: Possible values for *error_message* and their associated user actions are:

EOFException

A possible reason for this exception is that IMS Connect has closed the connection because an error occurred. See the z/OS console for associated IMS Connect error messages. IMS Connect error messages begin with the characters “HWS”. For more diagnostic information on the other return code and reason code values, and IMS Connect error messages, see the *IMS Messages and Codes* information.

SocketException: Connection reset by peer: socket write error

Possible reasons for the exception are:

- The underlying socket connection that is used for the interaction is no longer connected to IMS Connect. This can happen if IMS Connect is restarted during the interaction.
- TCP/IP on the host is shutting down.

SocketException: Connection reset

A possible reason for this exception is that IMS Connect has closed the connection because of an error occurred. See the z/OS console for associated IMS Connect error messages. IMS Connect error messages begin with the characters " HWS." For more diagnostic information on the other return code and reason code values, and IMS Connect error messages, see the *IMS Messages and Codes* information.

Another possible reason is that the User Message Exit HWSSOAP1 has not been properly installed with IMS Connect to handle the input message from SOAP Gateway. See the following message on the z/OS console for associated IMS Connect:

```
HWSP1445E UNKNOWN EXIT NAME SPECIFIED IN MESSAGE PREFIX; MSGID=*HWSOA1*/
HWSSOA1 , M=SDRC
```

Check the SOAP Gateway log for IMS Connect outages encountered and the timestamp information. The connection pool status is logged when the trace level is set to DEBUG.

IOGC004E Failed to set up SSL connection. [*error_message*].

Explanation: SOAP Gateway was unable to create an SSL connection with IMS Connect. *error_message* indicates the reason for the failure to set up the connection.

This message is also generated when SSL properties are configured in the SOAP Gateway server, but not all of the properties required for a valid SSL connection. These properties include:

- SSL keystore name
- SSL keystore password
- SSL truststore name
- SSL truststore password
- SSL encryption type

If a value is configured for any one of these properties, this error message is generated when SOAP Gateway attempts to send a message to IMS Connect. The target IMS Connect will generate a HWSP1445E error message with encrypted data in the msgid field:

```
HWSP1445E UNKNOWN EXIT NAME SPECIFIED IN MESSAGE PREFIX; MSGID=[unreadable characters], M=SDRC
```

IMS Connect interprets encrypted traffic on a non-SSL port as an OTMA message with an invalid exit name and issues the HWSP1445E message.

User response: Verify that the SSL information and connection information for IMS Connect are specified correctly. Verify that SSL is enabled on IMS Connect. Ensure that IMS Connect is running and that the host name and port number are correct. If you are connecting to a non-SSL port on the target IMS Connect, ensure that none of the SOAP Gateway SSL properties is configured with a value in the connection bundle for the web service.

If you use AT-TLS for the security between SOAP Gateway and IMS Connect, remove the SSL information in the connection bundle. Use the SOAP Gateway management utility iogmgmt -conn command to update the connection bundle.

IOGC005E SOAP Gateway internal error. [*error_message***]**

Explanation: An internal error has occurred in SOAP Gateway.

User response: Contact IBM Software Support. Specify the error message when reporting the problem.

IOGC006E IMS Connect XML Adapter returns an error. Return code: [*returncode***].**

Explanation: IMS Connect XML Adapter returned an error. The *error_message* variable provides a brief description of the return code.

User response: Examine the error message to determine the reason for the failure. Possible errors are:

-
  ```
  Return code: [108]. Inbound Error: The specified XML Converter Driver was
  not found or there was a load failure.
  ```

  Possible cause of this error:
  - The XML Converter Driver has not been compiled and link edit into load library. Ensure you have upload the XML Converter Driver program generated by Rational Developer for System z to your mainframe system. Make sure the driver program has been compiled and link edit into a data set or load library that is accessible by IMS Connect.
  - You have specified a wrong XML Converter Driver name in the correlator file for the web service. Correct the driver name property value in the correlator file for the web service.
  - The XML Converter Driver is bad and causes a load failure.
-
  ```
  Return code: [208]. Outbound Error: The specified XML converter driver [converter_name]
  was not found or cannot be loaded.
  ```

  Possible cause of this error:
  - The XML converter driver has not been compiled and link edit into load library. Ensure you have upload the XML converter driver program generated by Rational Developer for System z to your mainframe system. Make sure the driver program has been compiled and link edit into a data set or load library that is accessible by IMS Connect.
  - You have specified a wrong XML converter driver name in the correlator file for the web service. Correct the driver name property value in the correlator file for the web service.
  - The XML converter driver has errors and causes a load failure.
  - If a custom fault message is configure through Rational Developer for System z when you generated the converters, check for the HWSA0380E message in the IMS Connect system console for the name of the converter that caused the error. Because SOAP Gateway stores the name of the converter for processing the requests rather than the name of the converter for the fault message, the *converter_name* reported in this message might not be the converter that caused the problem.
- If you receive other return code values, contact IBM Software Support.

IOGC007E IMS Connect had an internal error. Return code: [*returncode*], Reason code: [*reasoncode*].

Explanation: An internal error occurred in IMS Connect.

User response: Check the Return and reason codes for IMS Connect exits documentation for the problem description. IMS Connect return codes reported by this message are decimal values. Refer to the decimal value column in the IMS Connect messages and codes documentation when looking up the problem description.

IOGC008E IMS returned an error: [*DFS_message*]

Explanation: IMS returned the IMS message *DFS_message* instead of the transaction output message.

User response: See the message explanation in the *IMS Messages and Codes* information. Some of the possible messages are:

- DFS065 `hh:mm:ss TRAN/LTERM STOPPED`

  The IMS transaction that the client wants to invoke is stopped. The transaction might have been stopped by the IMS /STOP command. Ensure that the IMS transaction is started. Use the IMS command /START to start the transaction if needed.
- DFS1292E `SECURITY VIOLATION`

  IMS OTMA has rejected the IMS transaction request because it failed the security check. Ensure that the user ID and the optional group name value in the connection bundle properties valid.
- DFS064 `hh:mm:ss DESTINATION CAN NOT BE FOUND OR CREATED`

  The transaction code value in the input request could not be recognized as a valid IMS transaction. Ensure that the transaction code property value in the correlator properties is valid.

IOGC009E The value [*lterm*] for the LTERM property exceeds 8 characters.

Explanation: The value of the LTERM property exceeds eight characters.

User response: Correct the length of the LTERM property value in the correlator properties for the web service to be less than or equal to eight characters.

IOGC010E The value [*password*] of the Password property exceeds 8 characters.

Explanation: The value of the password property exceeds eight characters.

User response: Correct the length of the password property value in the connection bundle properties for the web service to be less than or equals to eight.

IOGC011E The value [*user ID*] of the User ID property exceeds 8 characters.

Explanation: The value of the user name property exceeds eight characters.

User response: Correct the length of the user name property value in the connection bundle properties for the web service to be less than or equal to eight characters.

IOGC012E The value [*groupname*] of the Group name property exceeds 8 characters.

Explanation: The value of the group name property exceeds eight characters.

User response: Correct the length of the group name property value in the connection bundle properties of the web service to be less than or equal to eight characters.

IOGC013E The value [*datastore*] of the Datastore name property exceeds 8 characters.

Explanation: The value of the datastore name property exceeds eight characters.

User response: Correct the length of the datastore name property value in the correlator properties of the web service to be less than or equal to eight characters.

IOGC014E The value [*driver*] of the XML converter driver name property exceeds 8 characters.

Explanation: The value of the XML converter driver name property exceeds eight characters.

User response: The length of the XML converter driver name property value located in the correlator properties of the web service should be less than or equal to eight characters.

IOGC016E The value [*adapter_name*] of the adapter name property is invalid.

Explanation: The adapter name is not valid.

User response: SOAP Gateway supports two adapter types: "IBM XML Adapter" and "No Adapter". If you are handling the XML to bytes data conversion in your application instead of using the IMS Connect XML adapter function, set the adapter type property in the correlator file to `No Adapter` by using the SOAP Gateway management utility iogmgmt -corr command. On the z/OS or Linux on System z platforms, specify the adapter type with an underscore (`No_Adapter`).

IOGC020E The Execution timeout property value [*executiontimeout*] is invalid. The execution timeout value must either be -1 or in the range from 0 to 3600000 milliseconds.

Explanation: The execution timeout property value [*executiontimeout*] is invalid.

User response: Change the execution timeout property to between -1 to 3 600 000 in the correlator properties of the web service.

IOGC024E The Sync Level property value [*sync_level*] is not valid.

Explanation: The identified sync level is not supported by SOAP Gateway. SOAP Gateway supports a sync level of NONE for the web service provider scenario, and commit mode 0 and a sync level of CONFIRM for the web service consumer scenario. If the sync level is set to SYNCPT, this error is thrown.

User response: Correct the sync level in IMS Connect.

IOGC025E Error processing output message. Outbound codepage is not supported. *[Error_message]*

Explanation: The encoding of the output message from IMS Connect is not correct and cannot be processed.

User response: Ensure the code page setting in Rational Developer for System z is correct, and regenerate the XML converter driver(s).

IOGC026E The adapter error message is not formatted correctly. The content cannot be retrieved.

Explanation: SOAP Gateway cannot process the error message from the XML adapter function in IMS Connect.

User response: Contact IBM Software Support and provide the IMS Connect trace information.

IOGC027E The DFS message is not formatted correctly. The content cannot be retrieved.

Explanation: SOAP Gateway cannot process the message from IMS.

User response: Contact IBM Software Support and provide the IMS Connect trace information.

IOGC028E The IMS Connect RSM message is formatted incorrectly. Unable to retrieve the content.

Explanation: An internal error has occurred in IMS Connect.

User response: Contact IBM Software Support.

IOGC029E The IMS Connect message is not formatted correctly.

Explanation: An internal error has occurred in IMS Connect.

User response: Contact IBM Software Support.

IOGC030E The IMS Connect XML Adapter message is not formatted correctly.

Explanation: An internal error has occurred in IMS Connect.

User response: Contact IBM Software Support.

IOGC031E The XML converter driver reports an error. [*error_message***].**

Explanation: The Rational Developer for System z converter driver program returns an error.

User response: Review the error message. For more diagnostic information on the error message, see the Rational Developer for System z documentation.

IOGC034E Socket Timeout has occurred for this interaction. The Socket Timeout value specified was [*sockettimeout***] milliseconds. [***error_message***]**

Explanation: The time for SOAP Gateway to receive a response from IMS Connect is greater than the time specified for the socket timeout.

The *error_message* explains the reason for the error. A common error is `[SocketTimeoutException: Read timed out]`, which indicates that a timeout occurred when SOAP Gateway was trying to read a message from IMS Connect from the TCP/IP socket.

User response: Review the error message. Ensure that the value of the socket timeout is sufficient for SOAP Gateway to receive a response from IMS Connect. If it is not, increase the socket timeout value in the correlator properties for the web service. If the value of the socket timeout given is sufficient, it is possible that network problems are causing delays. Contact your network administrator.

IOGC035E IMS Connect default timeout has occurred for this interaction.

Explanation: The IMS Connect timeout value that is specified in the IMS Connect configuration member has been expired. The IMS Connect timeout value has been used because the execution timeout property for this interaction was not specified in the correlator file or has been set to zero.

User response: If the IMS Connect TIMEOUT value is not what you expected, modify the timeout value in the IMS Connect configuration member. For more information abut changing the timeout value for IMS Connect, see *IMS Version 13: Communications and Connections* information.

If you expect to use an execution timeout value instead of the IMS Connect timeout value, ensure that have a valid execution timeout specified in the correlator properties for the web service.

Related reference:

HWSCFGxx IMS Connect configuration member (IMS Version 13)
For information about IMS Connect configuration member, see IMS V13 System Definition information.

IOGC036E Execution timeout has occurred for this interaction. The timeout value used was [*exeuctiontimeout***] milliseconds.**

Explanation: The time it took for IMS Connect to send a message to IMS and receive the response was greater than the execution timeout value that was rounded to an appropriate execution timeout interval. SOAP Gateway converts the value of the execution timeout property in the correlator file to a value that IMS Connect can use.

User response: If the rounded execution timeout value is not what you expected, see the following table of conversion rules and change the execution timeout value in the correlator file as appropriate.

Table 38. Execution timeout value conversion rules

Range of execution timeout values in the correlator file	Conversion rule
1 – 250	If the value is not divisible by 10, it is converted to the next greater increment of 10.
251 – 1 000	If the value is not divisible by 50, it is converted to the next greater increment of 50.

Table 38. Execution timeout value conversion rules (continued)

Range of execution timeout values in the correlator file	Conversion rule
1 001 – 60 000	If the value is not divisible by 1 000, it is converted to the next greater increment of 1 000. Values that are exactly between increments of 1 000 are converted to the next greater increment of 1 000.
60 001 – 3 600 000	The value is converted to the nearest increment of 60 000. Values that are exactly between increments of 60 000 are converted to the next greater increment of 60 000.

IOGC037E The specified XML Converter Driver name cannot be empty or all blanks.

Explanation: The value of the driver name property cannot be empty or all blanks.

User response: Correct and specify the driver name property value in the correlator properties of the web service. Ensure the value is not empty or all blanks.

IOGC038E The specified adapter name cannot be empty or all blanks

Explanation: Most likely a custom XML conversion solution is used, but the adapter type is set to `IBM XML Adapter`. When the adapter type is set to `IBM XML Adapter`, a valid adapter name is expected.

User response: If you are using own XML conversion solution, specify `No Adapter` for the adapter type by using the SOAP Gateway management utility iogmgmt -corr command. On the z/OS or Linux for System z platforms, specify the adapter type with an underscore (`No_Adapter`).

Related reference:

"-corr: Create or update a correlator entry" on page 439
Use the -corr command to create or update the transaction and runtime properties of a correlator entry.

IOGC040E Bad output message.

Explanation: SOAP Gateway cannot process the output message.

This error can occur because the output message is encoded in a code page that is not supported by SOAP Gateway. An output message is built in the following ways:

- The output XML message is built and encoded by your IMS application. If the output message is built this way then this error occurs when the IMS application uses an code page that is not supported by SOAP Gateway to built the output XML message.
- The output message is built and then transformed into XML format by the IMS Connect XML Adapter and the Rational Developer for System z converters. If the output message is built this way, then the error occurs because the converter driver programs are created with an inbound and outbound code page value that is not supported by SOAP Gateway.

User response: If the output XML message is built by your IMS application, correct your application and ensure that the output message is encoded in a code page value that is supported by SOAP Gateway.

If the output XML message is built and transformed by the IMS Connect XML Adapter and the Rational Developer for System z converter driver program, correct the Rational Developer for System z converter driver program by regenerating the converter driver program with the code page values that is supported by SOAP Gateway.

IOGC041E IMS Connect returns an error. Return code: [*returncode*]. Reason code: [*reasoncode*]. [*error_message*]

Explanation: IMS Connect returned an error. *error_message* provides a brief description of the *reasoncode*.

User response: Review the error message to determine the reason for the failure. Possible causes of the error include:

- For `Return code: [8]. Reason code:[40]. [Security violation. User ID [`*userid*`], Group ID[`*groupid*`]]`, IMS Connect rejected the IMS transaction request because it failed the security check.
- For `Return code: [4]. Reason code:[72]. [IMS Datastore [`*datastore*`] not found]`, the specified IMS data store name cannot be found. Look at the IMS Connect configuration file and ensure that the IMS data store name is

specified and configured. Examine the IMS data store name property value in the correlator properties of the web service to ensure that it is specified and that it is valid. Datastore names must be in all uppercase characters.

- For `Return code: [4]. Reason code:[74]. IMS Datastore [datastore] in stop or close process`, the IMS with the data store name *datastore* is in stop and shut down process. Invoke the IMS transaction request again after IMS has been restarted.
- For `Return code: [8]. Reason code:[14]. Message not processed by an adapter.`, the message has not been processed by an adapter. For example, an invalid adapter name is specified or the adapter cannot be found. Possible causes for this error include:
  - An invalid adapter type is specified. Ensure that IBM XML Adapter is specified in the adapter type property of the correlator file of the web service.
  - The IBM XML adapter cannot be found. Ensure that XML Adapter has been configured correctly with IMS Connect.
  - The IBM XML Adapter or the XML converter driver has returns an error.
  - The IBM XML Adapter has return a zero length message after processing the XML message. This error can occur because the XML message cannot be processed by XML converter driver.

In addition, see the z/OS console for associated IMS Connect error messages. IMS Connect error messages begin with the characters "HWS". For more diagnostic information on the other return code and reason code values, and IMS Connect error messages, see the *IMS Messages and Codes* information.

IOGC042E The value [*trancode*] of the IMS Transaction code property exceeds 8 characters.

Explanation: The value of the transaction code property exceeds eight characters.

User response: Correct the length of the transaction code property value in the correlator properties of the web service to be less than or equals to eight characters.

IOGC044E The value [*reroute_name*] of the reroute name property exceeds 8 characters.

Explanation: The reroute name cannot exceed eight characters.

User response: Correct the reroute name.

IOGC045E The reroute name property value *reoute_name* is not valid

Explanation: Valid characters are a-z, A-Z, 0-9, @, #, and $.

User response: Correct the reroute name.

IOGC051E IMS Connect return an error. IMS Datastore is not available.

Explanation: This error message is returned from IMS Connect. The transaction failed because the IMS data store is not available at this moment.

User response: Report the error to your IMS system administrator to ensure the IMS data store you are using is in active status.

IOGD messages

IOGD messages are generated by the SOAP Gateway management utility.

After an IOGD message is generated, the utility always returns control to the command prompt, terminal, or console, depending on platform. As a result, these messages do not contain specific system action details.

IOGD0001E The *command_name* command failed because it does not include all of the required parameters. The *command_name* command requires *required_parameters*.

Explanation: One or more of the required parameters was not included when the command was issued.

User response: Reissue the command with all the required parameters.

IOGD0002E *option_name* **is not a valid -callout command option.**

Explanation: The -callout command requires one of the following command options:

- -startall
- -startone
- -stopall
- -stopone
- -stoppool
- -updateprop

User response: Select a -callout command option from the list and retry the command.

Related reference:

Chapter 11, "SOAP Gateway management utility reference," on page 429
The SOAP Gateway management utility provides a command line interface to manage the SOAP Gateway server runtime, configure server properties, and work with web service artifacts.

IOGD0003E *command_name* **is not a valid command.**

Explanation: The command failed because it is not a supported command.

User response: See the SOAP Gateway management utility command reference information for a list of valid commands.

Related reference:

Chapter 11, "SOAP Gateway management utility reference," on page 429
The SOAP Gateway management utility provides a command line interface to manage the SOAP Gateway server runtime, configure server properties, and work with web service artifacts.

IOGD0004E The *command_name* **command failed because the connection bundle name is invalid. A valid connection bundle name must be provided as the** *argument_order* **positional pair of arguments for an** *command_name* **command, is 1 to 20 characters in length and contains no blank characters.**

Explanation: There are multiple possible causes for this message:

- A connection bundle name was not specified with the *command_name* command.
- The connection bundle name that was specified was not in the correct *argument_order* position in the list of parameters specifed with the command. This error might have caused another parameter value to be parsed, incorrectly, as the connection bundle name.
- The connection bundle name that was specified is invalid because it is not 1 - 20 characters long, or because it contains blanks.

User response: Determine the cause of the error and reissue the command with a valid connection bundle name.

IOGD0005E The *command_name* **command failed because the file specified, (***file_name***), is not a valid** *file_type* **file. Provide a valid file name with the** *parameter_name* **parameter.**

Explanation: The specified file is not a valid instance of the specified type.

User response: Ensure that the file name is correct and reissue the command.

IOGD0006E An I/O exception occurred with the (*file_name***) file. Either the file could not be read or found, or the file encoding is not UTF-8. Message=** *error_details*

Explanation: A web service artifact is corrupted or missing.

User response: Delete the file and recreate it.

IOGD0008E The *command_name* command failed because the *parameter_name* identifier parameter is invalid. The *command_name* command requires a *parameter_name* identifier parameter and its associated *parameter_value* value.

Explanation: The specified parameter is not valid with the specified command.

User response: Reissue the command with the indicated parameter and a valid value.

IOGD0009E The *command_name* command failed because a parameter was missing.

Explanation: Because your current thread policy is one thread per tpipe, a valid tpipe name must be specified with the -p parameter.

User response: Reissue the command with a valid tpipe name for the thread.

IOGD0010E The command to update callout properties failed because the (*parameter_name*) is not valid for this command.

Explanation: The `-callout -updateprop` command only supports the parameters `-1` to `-10`.

User response: Reissue the command with a valid parameter.

Related reference:

"-callout -updateprop: Update SOAP Gateway callout properties" on page 434
The -callout –updateprop command updates the SOAP Gateway callout properties.

IOGD0011E An invalid value was specified for the Callout Resume Tpipe Poll Interval property.

Explanation: The poll interval is specified in milliseconds and must be 1 - 86400000.

User response: Reissue the command with a valid interval value.

IOGD0012I The specified *property_name* property value is blank.
The property value was reset to its default value: *property_default*.

Explanation: This message is informational.

User response: No action is required.

IOGD0013E An invalid value was specified for the Callout Stop on Resume Tpipe Error property.

Explanation: The valid values are true and false.

User response: Reissue the command with a valid value.

IOGD0014E An invalid value was specified for the Callout One Thread per Tpipe property.

Explanation: The valid values are true and false.

User response: Reissue the command with a valid value.

IOGD0015E An invalid value for Callout Number of Worker Threads in Pool property.

Explanation: The valid values are 1 - 32.

User response: Reissue the command with a valid value.

IOGD0016E An invalid value was specified for the Callout Queue Throttle Length property.

Explanation: The valid values are 1- 64.

User response: Reissue the command with a valid value.

IOGD0017E An invalid value was specified for the Callout Check Worker Health Interval property.

Explanation: The valid values are 1 - 86400000 milliseconds.

User response: Reissue the command with a valid value.

IOGD0018E An invalid value was specified for the Callout Thread Pool Cache Capacity property. The valid values are 1 - 2000. Reissue the command with a valid value.

Explanation: The valid values are 1 - 2000.

User response: Use the -callout -updateprop command to specify a valid value.

IOGD0019E An invalid value was specified for the Callout Discard Pending Messages on Error property.

Explanation: The valid values are true and false.

User response: Reissue the command with a valid value.

IOGD0020E An invalid value was specified for the Callout Discard Pending Messages on Shutdown property.

Explanation: The valid values are true and false.

User response: Reissue the command with a valid value.

IOGD0022E An invalid value was specified for the callout thread pool cache directory.

Explanation: The directory must be a subdirectory of the SOAP Gateway installation directory.

User response: Reissue the command with a valid directory.

IOGD0023E The *command_name* command failed because (*option_name*) is not a valid option for this command.

Explanation: Some management utility commands have a specific set of valid command options, which are listed in the command reference information.

User response: See the command reference information and reissue the command with a valid command option.

Related reference:

Chapter 11, “SOAP Gateway management utility reference,” on page 429
The SOAP Gateway management utility provides a command line interface to manage the SOAP Gateway server runtime, configure server properties, and work with web service artifacts.

IOGD0024E The *command_name* command failed because the *parameter_name* parameter and a valid value are required for this command.

Explanation: The specified command was not issued with all required parameters.

User response: See the reference information for the command and reissue it with all the required parameters.

Related reference:

Chapter 11, “SOAP Gateway management utility reference,” on page 429
The SOAP Gateway management utility provides a command line interface to manage the SOAP Gateway server runtime, configure server properties, and work with web service artifacts.

IOGD0025E The *incorrect_command_name* command is not valid. The valid command is *correct_command_name* and it is case sensitive.

Explanation: The specified command was not issued with the correct cases.

User response: Reissue the suggested correct command in the exact cases.

IOGD0026E The *command_name* command failed because the specified *file_type* file (*file_path*) does not exist or is not accessible to the command.

Explanation: The specified file was not found, or is not accessible to the SOAP Gateway management utility because of security restrictions.

User response: Check the file name, file path, and security permissions, and then reissue the command.

IOGD0027E Correlator validation failed due to an invalid field name (*field_name*).

Explanation: The specified correlator XML file failed validation with the correlator schema. This error occurs if the file name is incorrect, if the correlator file is based on an obsolete schema, or if the file is corrupted.

User response: Check the file name and extension to ensure that the file is a valid correlator file. If the file is valid, ensure that the software product or utility that is used to generate the correlator meets the minimum requirement for the version of SOAP Gateway that you have.

IOGD0029I The undeploy command successfully removed the service from the SOAP Gateway master configuration along with its WSDL file, *wsdl_file_name*, and associated correlator XML file, *correlator_file_name*.

Explanation: This message is informational.

User response: No action is required.

IOGD0031E The *command_name* command failed because the specified XML adapter type (*adapter_type_name*) is invalid. Provide a valid adapter type. The default type for an adapter generated with Rational Developer for System z is `IBM XML Adapter`. Specify `No adapter` if the target IMS Connect is not using an XML adapter.

Explanation: The default value for an IMS Connect instance with an XML adapter is `IBM XML Adapter`. If IMS Connect is not using an adapter, specify the name `No adapter`.

User response: Create a new correlator file with IBM Rational Developer for System z that references a valid XML adapter type, replace the existing correlator file, and reissue the command. If the target IMS Connect does not use an XML adapter, update the existing correlator file with the adapter type "No adapter".

IOGD0033E The *command_name* command failed because the specified callout connection bundle name (*callout_connbundle_name*) is invalid.

Explanation: A callout connection bundle must have a 1 - 20 character name that does not contain blanks

User response: Reissue the command with a valid callout connection bundle name.

IOGD0034E The *command_name* command failed because the specified execution timeout value (*timeout_value*) is invalid.

Explanation: The valid range is -1 to 3600000 milliseconds. The default is 0, which instructs IMS Connect to use its own timeout value. Specifying -1 instructs IMS Connect to wait forever.

User response: Reissue the command with a valid timeout value.

IOGD0035E The *command_name* command failed because the specified correlator service name (*correlator_service_name*) is invalid.

Explanation: The specified correlator service name was not found in the WSDL file of the web service.

User response: Ensure that the specified service name exists in the WSDL file of the target web service and reissue the command.

IOGD0036E The *command_name* command failed because the specified lterm name (*lterm_name*) is invalid.

Explanation: An lterm name must be 1 - 8 characters and can contain only alphanumeric characters and the following special characters: @ # $

User response: Reissue the command with a valid lterm name. The lterm name is an optional property and can be omitted from the command.

IOGD0037E The *command_name* command failed because the specified socket timeout value (*socket_timeout*) is invalid.

Explanation: The valid range for the socket timeout value is 0 to 3660000 milliseconds. The socket timeout value is specified in milliseconds. It must be greater than the execution timeout value. The execution timeout value is either the default value that is used by the target IMS Connect, or the override value that is specified in the correlator file.

User response: Reissue the command with a valid value, or omit this parameter to use the default value of 0 milliseconds.

IOGD0038E The *command_name* command failed because the *command_option* option specified a blank *artifact_name*.

Explanation: The specified command option requires the name of an artifact, depending on the option. For example, the -connectionbundleentry option requires a connection bundle name.

User response: Reissue the command with a valid artifact name.

IOGD0039E The *command_name* command failed because the specified connection bundle name (*connection_bundle_name*) is invalid.

Explanation: The specified connection bundle name is invalid. A valid name is 1 - 20 characters and does not contain blanks.

User response: Reissue the command with a valid connection bundle name.

IOGD0040E The *command_name* command failed because the specified callout WS timeout value (*web_service_timeout_value*) is invalid.

Explanation: The specified callout web service timeout value is not in the valid range. Valid values are 0 - 2147483647 milliseconds. The default is 7500 milliseconds.

User response: Reissue the command with a valid web service timeout value or omit this parameter to use the default value.

IOGD0041E The *command_name* command failed because the specified operation name (*correlator_operation_name*) is invalid or was not found in the specified WSDL file.

Explanation: Each correlator entry must have a unique service and operation name from the WSDL file of the target web service.

User response: Ensure that the operation name exists in the WSDL file and reissue the command.

IOGD0042E The *command_name* command failed because the specified IMS transaction code value (*trancode_value*) is invalid.

Explanation: The transaction code must be 1 - 8 alphanumeric characters and must match the transaction code of the application in the target IMS host. If no transaction code is specified, SOAP Gateway does not append a transaction code to the message when sending it to IMS. This parameter is optional.

User response: Reissue the command with a valid transaction code.

IOGD0043E The *command_name* command failed because the IMS Connect XML converter name (*converter_name*) is invalid.

Explanation: A converter name must be specified if the XML adapter function is used. A converter name is 1 - 8 alphanumeric characters and does not contain blanks.

User response: Check the name of the XML converter in IMS Connect and reissue the command.

IOGD0044E The *command_name* command failed because *parameter_name* is not a valid parameter.

Explanation: The specified parameter was not recognized by the command parser.

User response: See the SOAP Gateway management utility information for the command. Reissue the command with valid parameters.

Related reference:

Chapter 11, “SOAP Gateway management utility reference,” on page 429
The SOAP Gateway management utility provides a command line interface to manage the SOAP Gateway server runtime, configure server properties, and work with web service artifacts.

IOGD0045E The *command_name* command failed because *parameter_name* is not a valid argument for this command.

Explanation: The specified parameter is not valid for this command.

User response: See the SOAP Gateway management utility information and reissue the command with valid parameters.

Related reference:

Chapter 11, “SOAP Gateway management utility reference,” on page 429
The SOAP Gateway management utility provides a command line interface to manage the SOAP Gateway server runtime, configure server properties, and work with web service artifacts.

IOGD0047E The *command_name* command failed because the *parameter_name* parameter value cannot be blank.

Explanation: A value is required for the specified parameter.

User response: Reissue the command with a valid value for the parameter.

IOGD0048E The *command_name* command failed because the specified connection bundle name (*connection_bundle_name*) does not exist in the master configuration of the target SOAP Gateway.

Explanation: The specified connection bundle was not found. The connection bundle must exist in the SOAP Gateway server file system in the *install_dir*/imssoap/xml directory.

User response: Confirm that the connection bundle exists and is correctly named, and then reissue the command.

IOGD0049E The *command_name* command failed because the specified IMS datastore name (*datastore_name*) is invalid.

Explanation: The data store name value is configured in the IMS Connect configuration member of the IMS.PROCLIB data set (HWSCFGxx), and the value given in SOAP Gateway must match. The value is case-sensitive and must be uppercase. The value must be 1 - 8 characters and can contain only alphanumeric characters and the following special characters: # $ @

User response: Ensure that the data store name is correct and reissue the command.

IOGD0050E The *command_name* command failed because the specified SSL encryption type (*encryption_type*) is invalid.

Explanation: The specified SSL encryption type is not one of the following valid values: "strong", "weak", or "none". This value determines the level of SSL encryption that is used to secure messages sent through SOAP Gateway. If the encryption type is "none" then no encryption is used but the SSL handshake is still performed.

User response: Specify "strong", "weak", or "none" for the encryption type and reissue the command.

IOGD0051E The *command_name* command failed because the specified group name (*RACF_group_name*) is invalid.

Explanation: The specified RACF group name is invalid. A RACF group name must be 1 - 8 characters and can contain only alphanumeric characters and the following special characters: # $ @

The specified RACF group name must match the name that is configured in RACF for the target IMS Connect and IMS system.

User response: Ensure that the RACF group name is correct and reissue the command.

IOGD0052E The *command_name* command failed because the specified keystore or truststore name (*store_name*) is invalid.

Explanation: A keystore or truststore name must be 6 - 20 characters and the file type must be `.ks`

User response: Correct the keystore or truststore name and reissue the command.

IOGD0053E The *command_name* command failed because the specified password (*specified_password*) is invalid.

Explanation: The password must be 6 - 20 characters.

User response: Reissue the command with a valid password.

IOGD0054E The *command_name* command failed because the specified port number (*port_number*) is invalid.

Explanation: The port number must be in the range of 1 - 65535. The listening port is specified with the PORTID= parameter of the HWSCFGxx configuration member of the IMS.PROCLIB data set for the target IMS Connect. A single IMS Connect can have up to 50 listening ports, but only one port can be specified for each connection bundle in SOAP Gateway.

User response: Obtain the listening port number from the IMS Connect configuration settings and reissue the command.

IOGD0055E The *command_name* command failed because the specified RACF password (*password*) is invalid.

Explanation: The RACF password must be 1 - 8 characters and might be case sensitive depending on the RACF settings for the target IMS Connect.

User response: Confirm the RACF password for the target IMS Connect and reissue the command.

IOGD0056I The *command_name* command successfully updated the SOAP Gateway master configuration. The parameters submitted with the command were: *parameter_list*.

Explanation: The SOAP Gateway server file system was updated with the listed properties.

User response: No action is required.

IOGD0057E The *command_name* command failed because the specified RACF user ID (*user_ID*) is invalid.

Explanation: The user ID must be 1 - 8 characters.

User response: Specify a valid SAF (security authorization facility) user ID and reissue the command.

IOGD0059E The *command_name* command failed because a specified callout tpipe name (*tpipe_name*) is invalid. Provide a valid tpipe name or a comma separated list of valid tpipe names with the *parameter_name* parameter.

Explanation: The *command_name* command failed for one of the following reasons:

- A tpipe name was longer than 8 characters.
- A tpipe name contained invalid characters.
- A list of tpipe names was not separated by commas or contained blanks between tpipe names.

A tpipe name is 1 - 8 characters, and can contain only alphanumeric characters and the following special characters: # $ @

A tpipe name cannot contain blanks.

User response: Specify a valid tpipe name, or a list of valid tpipe names separated with commas, and reissue the command.

IOGD0060E The *command_name* **command failed because the specified connection bundle name (***connection_bundle_name***) already exists in the SOAP Gateway master configuration. Specify a unique connection bundle name with the** *parameter_name* **parameter.**

Explanation: A connection bundle name can only be used once in a single SOAP Gateway instance.

User response: Reissue the command with a unique connection bundle name.

IOGD0061E The *command_name* **command failed because the** *file_type* **file (***file_name***) does not exist or could not be deleted.**

Explanation: The specified file was not deleted because the file was not found or because security permissions on the file stopped the delete operation.

User response: Ensure that the file name and fully qualified path are correct, and that the SOAP Gateway management utility has the required security permissions to delete the file, and then reissue the command.

IOGD0062E The *command_name* **command failed because the** *command_name* **command requires the "-u" (update) function parameter. No value was provided.**

Explanation: The command must be immediately followed by `-u`

User response: Reissue the command with the `-u` option.

IOGD0063I The (*connection_bundle_name***)** *connection_bundle_type* **connection bundle has been deleted.**

Explanation: This message is informational.

User response: No action is required.

IOGD0064E The *command_name* **command failed because the specified connection bundle named (***connection_bundle_name***) does not exist. Verify the connection bundle name.**

Explanation: The specified connection bundle was not found.

User response: Ensure that the connection bundle name is correct and reissue the command.

IOGD0071E Invalid WSDL file name (*file_name***).**

Explanation: A WSDL file name is required.

User response: Ensure that the file name is correct, specify the fully qualified path to the file, and reissue the command.

IOGD0072E Invalid correlator file name (*file_name***).**

Explanation: A correlator XML file name is required.

User response: Ensure that the file name is correct, specify the fully qualified path to the file, and reissue the command.

IOGD0076E The *command_name* command failed because the specified WS-Security type, (*security_type*) is invalid. Provide a valid WS-Security type value: *security_options*.

Explanation: The valid token types for the web service provider scenario are:

- `SAML11Token`
- `SAML11SignedTokenTrustAny`
- `SAML11SignedTokenTrustOne`
- `SAML20Token`
- `SAML20SignedTokenTrustAny`
- `SAML20SignedTokenTrustOne`
- `UserNameToken`

The valid token types for the synchronous callout scenario are:

- `SAML11Token`
- `SAML20Token`

User response: Reissue the command with a valid token type.

IOGD0077E The *command_name* command failed because the *command_name* command requires a task option.

Explanation: The valid task options are "-c" (create) and "-u" (update).

User response: Reissue the command with a valid task option.

IOGD0078I The *command_name* command succeeded, but the value (*parameter_value*) for the *parameter_name* parameter will be ignored. Web service consumer security properties are ignored for web service providers.

Explanation: A callout service property was specified for a web service provider and was ignored.

User response: No action is required.

IOGD0079E You cannot enable *requested_option*, since *current_option* is already enabled. You must first disable *current_option* before you can enable *requested_option*.

Explanation: The specified option can only be enabled if the conflicting option is already disabled.

User response: Stop the SOAP Gateway server, disable the current option, enable the new option, and then restart the SOAP Gateway server.

IOGD0081I *option_value* cannot be disabled since it is already disabled. Processing of any optional parameters specified with this command will continue.

Explanation: This message is informational.

User response: No action is required.

IOGD0082I The *command_name* command did not complete normally because the *option_name* option requires *option_value* as a value. The invalid value (*specified_value*) was replaced with the default value: *default_value*.

Explanation: This message is informational.

User response: No action is required.

IOGD0083E A truststore name and password are not required for server authentication.

Explanation: Truststore information is used only for client authentication.

User response: Reissue the command without a truststore name and password.

IOGD0084E The *command_name* command to enable client authentication failed because an SSL port number, keystore name, keystore password, truststore name, and truststore password must be specified.

Explanation: All of the listed parameters are required to enable client authentication.

User response: Reissue the command with all the required parameters.

Related reference:

"-prop: Set SOAP Gateway properties" on page 450
Use the -prop command to modify the SOAP Gateway server properties.

IOGD0085I The *property_name* property value was set to (*new_property_value*).

Explanation: The indicated property was successfully updated.

User response: You must restart the server for the change to take effect.

IOGD0086E The *command_name* command failed because the *command_name* command requires a task option. No value was provided.

Explanation: The command requires one of the following task options:

-c Create

-u Update

-v View

-d Delete

User response: Reissue the command with a valid task option.

IOGD0089E The *command_name* command failed because the *command_name* command does not support *invalid_task_option* as a task option.

Explanation: The valid task options are "-c" (create) and "-u" (update).

User response: Reissue the command with a valid task option.

IOGD0090I Processing of the *command_name* command for resume tpipe thread *thread_name* is complete.

Explanation: This message is informational.

User response: No action is required.

IOGD0091I Processing of the *command_name* command for connection bundle *connection_bundle_name* is complete.

Explanation: This message is informational.

User response: No action is required.

IOGD0093E The *command_name* command failed because the specified *command_name* command task option (*task_option*) is invalid.

Explanation: The command requires one of the following task options:

-c Create

-u Update

-v View

-d Delete

User response: Reissue the command with a valid task option.

IOGD0094E To enable server authentication for your SOAP Gateway server, an SSL port number, key store name, and key store password must be specified.

Explanation: A required parameter for the command was not specified.

User response: Reissue the command with all the required parameters.

IOGD0095I *option_name* **was successfully enabled in the SOAP Gateway server master configuration. The changes will take effect the next time that the SOAP Gateway server starts.**

Explanation: This message is informational.

User response: No action is required.

IOGD0096E The command to update the SOAP Gateway server properties failed because the SOAP Gateway management utility does not have the required security permissions.

Explanation: The SOAP Gateway management utility does not have read and write permission on the server XML configuration file.

User response: Ensure that the SOAP Gateway management utility has permission to read and modify the `server.xml` file.

IOGD0097E This VM does not support the Latin-1 character set.

Explanation: This character set could not be decoded.

User response: Contact IBM software support.

IOGD0098E The command to restore the SOAP Gateway server properties to their default values failed because the SOAP Gateway management utility does not have the required security permissions.

Explanation: The SOAP Gateway management utility requires read and write permission on the server XML configuration file.

User response: Ensure that the SOAP Gateway management utility has permission to read and modify the `server/conf/server.xml` file.

IOGD0100E The command failed because the *parameter_name* **parameter is invalid.**

Explanation: The specified parameter is not supported by this command.

User response: See the reference information for the command for a list of valid parameters. Reissue the command using only valid parameters.

IOGD0101E The *command_name* **command failed because not all the required parameters were specified. The** *command_name* **command requires: correlator file name, operation name, and service name.**

Explanation: The indicated parameters are used to identify the artifact to update.

User response: Reissue the command with all the required parameters.

IOGD0102E The *command_name* **command failed because the specified** *option_name* **option value,** *option_value***, is invalid. Provide a valid** *option_name* **value:** *option_value_choices***.**

Explanation: The specified parameter has a fixed list of valid values.

User response: Reissue the command with a valid value for the indicated parameter.

Related reference:

Chapter 11, “SOAP Gateway management utility reference,” on page 429
The SOAP Gateway management utility provides a command line interface to manage the SOAP Gateway server runtime, configure server properties, and work with web service artifacts.

IOGD0104I **The deploy command successfully deployed the** *service_type* **web service to the runtime and master configurations:**
web service definition:
wsdl_file
Correlator XML:
correlator_file
Schema XML files:
schema_file_name

Explanation: This message is informational.

User response: No action is required.

IOGD0105E **File name (***file_name***) must have the** *extension_type* **extension. Provide a valid** *file_type* **file name with an** *extension_type* **extension, such as** *example_file_name***.**

Explanation: The command failed because the specified file name did not have the correct extension.

User response: Specify a file name with a valid extension. Reissue the command.

IOGD0106E **The command failed because the specified** *file_type* **file,** *file_name***, does not exist. Specify a valid** *file_type* **file.**

Explanation: A valid *file_type* file is required for the command.

User response: Ensure that the specified file exists in the target file system of the target SOAP Gateway, and reissue the command with the fully qualified path to the file.

IOGD0107E **The SSL** *truststore_or_keystore* **password was not updated because the new password is invalid. Provide a valid password, 6 - 20 characters long.**

Explanation: Truststore and keystore passwords must be 6 - 20 characters.

User response: Reissue the command with a valid password.

IOGD0111E **The command to update callout properties failed because no properties were specified with the command.**

Explanation: At least one callout property must be specified when a command to update properties is issued.

User response: Reissue the command with at least one property to update.

IOGD0112I **The changes made by the** *command_name* **command were successfully updated in the affected deployed applications currently active in the SOAP Gateway runtime configuration.**

Explanation: Correlator properties were updated successfully.

User response: No action is required.

IOGD0113I **The** *command_name* **command successfully changed the SOAP Gateway master configuration.**

Explanation: The SOAP Gateway server file system was updated. The changes will be reflected in the runtime configuration of the server after the next time that the SOAP Gateway server starts.

User response: No action is required.

IOGD0114E The command to update the *current_connection_bundle_name* **connection bundle name failed because the new connection bundle name (***new_connection_bundle_name***) is already in use.**

Explanation: Connection bundle names must be unique within a SOAP Gateway server.

User response: Reissue the command with a unique connection bundle name.

IOGD0116E Unable to obtain the callout thread status because of the following underlying error: *error_details*

Explanation: The command failed either because SOAP Gateway has not started or because the server is experiencing an unrelated error that is blocking access to the callout threads.

User response: Ensure that the SOAP Gateway server is started and reissue the command. If this error recurs, examine the error details for more information about the problem.

IOGD0117E Unable to obtain the callout thread information.

Explanation: Either the SOAP Gateway has not started or the server is experiencing an unrelated error that is blocking access to the callout threads.

User response: Ensure that the SOAP Gateway server is started and reissue the command.

IOGD0122E The *file_type* **file specified with the** *command_name* **command, (***file_name***), could not be parsed.**

Explanation: To prevent server errors, WSDL files and correlator files are checked for validity against the current schema versions for the file type before being sent to the server. This error occurs when the validity check fails.

User response: Ensure that the specified file name is correct and that the you are using a supported method to generate the file.

IOGD0123I The *property_name* **SOAP Gateway property has been set to** *new_value***.**

Explanation: This message is informational.

User response: No action is required.

IOGD0124I The *property_name* **SOAP Gateway property provided as an argument to The** *command_name* **command was blank. The** *property_name* **property has been set to its default value,** *property_value***.**

Explanation: This message is informational.

User response: No action is required.

IOGD0125E The *command_name* **command failed because the value specified for the** *property_name* **property (***property_value***) is invalid. Provide a valid value for the** *parameter_name* **parameter. A valid value is** *example_value***.**

Explanation: Only the listed value, range, or list of examples are valid.

User response: Reissue the command with a valid value.

Related reference:

Chapter 11, “SOAP Gateway management utility reference,” on page 429
The SOAP Gateway management utility provides a command line interface to manage the SOAP Gateway server runtime, configure server properties, and work with web service artifacts.

IOGD0128I As a result of the incomplete update of the SOAP Gateway master configuration, no attempt was made to update the SOAP Gateway runtime configuration.

Explanation: An update to the SOAP Gateway server was partially successful in the master configuration. To maintain system integrity, the automatic update to the runtime configuration was cancelled. Instead, the change will be reflected in the runtime configuration after the next time the SOAP Gateway server starts.

User response: No action is required.

IOGD0129E The *command_name* command failed because the correlator file is not version 3.0. Migration of the correlator file is mandatory.

Explanation: Only version 3.0 correlator files are valid.

User response: Either re-create the correlator file or migrate it with the migration tool. Then reissue the command.

Related tasks:

"Migrating correlator files to schema version 3.0" on page 302
IMS Enterprise Suite Version 3.1 SOAP Gateway requires correlator schema version 3.0. To migrate an existing correlator file from older versions to version 3.0, use the SOAP Gateway management utility iogmgmt -migrate correlator command.

IOGD0130E The *command_name* command failed because the specified *property_name* value (*property_value*) is invalid.

Explanation: The valid values are true and false.

User response: Reissue the command with a valid value.

IOGD0131E The *command_name* command failed because the specified service name *service_name* and operation name *operation_name* cannot be located in the correlator file.

Explanation: The specified correlator file does not contain an entry that corresponds to the specified service and operation name. The specified correlator file was not updated.

User response: Check the service name and operation name of the correlator entry you want to update. You can verify the names in both the correlator XML file and in the WSDL file for the web service. The service name and operation names must be the same in both the correlator XML file and the WSDL file for the web service.

IOGD0136E The web service was not deployed because there is not enough disk space.
There is *free_disk_space* MB available space.
***required_disk_space* MB is required.**

Explanation: No web services can be deployed until additional disk space is available.

User response: Increase the amount of available disk space and then deploy the web service again.

IOGD0137E The web service was not deployed because there is insufficient authority to access the client proxy directory and create files. Provide read, write, and execute access to the resource: *resource_path*.

Explanation: The SOAP Gateway management utility requires read, write, and execute permissions on the specified path.

User response: Add the required permissions and then deploy the web service again.

IOGD0138E The *command_name* command failed because both a keystore name and a keystore password are required.

Explanation: When you specify a keystore name, you must also specify a keystore password that SOAP Gateway uses to access the keystore.

User response: Reissue the command with the keystore password.

IOGD0139E The *command_name* command failed because both a truststore name and a truststore password are required.

Explanation: When you specify a truststore name, you must also specify a truststore password that SOAP Gateway uses to access the truststore.

User response: Reissue the command with a truststore password.

IOGD0141E The *command_name* command failed because the specified command argument, *parameter_name*, is invalid.

Explanation: This command supports a fixed set of parameters, which are listed in the command reference information.

User response: See "-prop: Set SOAP Gateway properties" on page 450 for a list of parameters valid with this command.

Reissue the command with a valid parameter.

IOGD0142E The *command_name* command failed because the specified *file_type* file (*file_name*) failed validation with the *file_type* schema file. Provide a valid *file_type* file.

Explanation: This error can occur because the file was not migrated when you upgraded SOAP Gateway or because it was generated using an incompatible version of IBM Rational Developer for System z.

User response: Re-create the file using a compatible utility or development tool.

Related concepts:

"Software requirements" on page 43
IMS Enterprise Suite Version 3.1 SOAP Gateway requires IMS Version 13, Version 12, or Version 11.

IOGD0143E The *command_name* command failed because the specified socket time out value *socket_timeout* must be greater than the execution time out value *execution_timeout*.

Explanation: The SOAP Gateway management utility verifies that the socket timeout value is greater than the execution timeout value before changing the server configuration.

User response: Reissue the command with a valid socket timeout value that is greater than the execution timeout value.

IOGD0144E The *command_name* command failed because the specified command argument, (*argument_name*), is only valid for (*type*) callout connection bundle.

Explanation: The indicated command argument is not valid for this type of connection bundle.

User response: Reissue the command with a valid argument.

IOGD0145E The *command_name* command failed because both the callout basic authentication name and callout basic authentication password are required.

Explanation: When you specify a callout basic authentication user ID in a connection bundle, you must also specify a basic authentication password.

User response: Reissue the command with a basic authentication password.

IOGD0150W A correlator file was updated with a new connection bundle name or callout connection bundle name in the master configuration.

Explanation: Connection bundle names are locked in the runtime configuration and can be changed only in the master configuration. The old connection bundle name will be replaced with the new connection bundle name the next time the SOAP Gateway server starts.

User response: Restart the SOAP Gateway server.

IOGD0200E The iogmgmt view command failed because two conflicting parameters were specified, -connectionbundle (or its shortcut -cb) and -connectionbundleentry (or its shortcuts -ce or -n). Only one of these parameters can be specified in an iogmgmt view command.

Explanation: A `-view` command can be issued to view either connection bundles or a connection bundle entry, but not both.

User response: Reissue the command with only one of the parameters.

IOGD0201E The iogmgmt command failed because a valid command parameter was not specified.

Explanation: A SOAP Gateway management utility command requires a valid command parameter.

User response: See the SOAP Gateway management utility command reference information for a list of valid command parameters.

Related reference:

Chapter 11, "SOAP Gateway management utility reference," on page 429
The SOAP Gateway management utility provides a command line interface to manage the SOAP Gateway server runtime, configure server properties, and work with web service artifacts.

IOGD0202E The iogmgmt command failed because multiple command parameters were specified.

Explanation: A SOAP Gateway management utility command requires one and only one command parameter, such as `-view`.

User response: Reissue the command with one command parameter.

IOGD0203E The *command_type* view command failed because the property *property_name* was not found in the *properties_file_name* file.

Explanation: The indicated properties file exists but does not contain the requested property value. This error indicates that the properties file is in an invalid state.

User response: Reinstall SOAP Gateway.

IOGD0204E The -service command keyword is supported only for SOAP Gateway servers on the Windows operating system.

Explanation: This message is generated when the -service argument is used on a non-Windows platform.

User response: Reinstall SOAP Gateway.

IOGD0205E The command *command_name* failed because the option *option_1* was specified but it is mutually exclusive with option *option_2* and all other options in this group: *list*

Explanation: The SOAP Gateway management utility rejected the command because two of the specified options (*option_1* and *option_2* in the message text) are mutually exclusive.

User response: Reissue the command with valid options.

IOGD0206E The command *command_name* failed because of a missing argument for parameter *parameter_name*. Parameter usage: *usage_information*

Explanation: The SOAP Gateway management utility rejected the command because the value given for the indicated parameter is invalid. In the message text, *usage_information* gives the valid values for the parameter.

User response: Reissue the command with a valid value for the parameter.

IOGD0207E The command *command_name* failed because it did not include all the required parameters. The command requires one or more of the parameters: *parameter_list*

Explanation: The SOAP Gateway management utility rejected the command because one or more required parameters were omitted. In the message text, *parameter_list* shows the required parameters for the command.

User response: Reissue the command with all of the required parameters.

IOGD0208E The command *command_name* failed because *parameter_name* is not a supported parameter.

Explanation: The SOAP Gateway management utility rejected the command because an invalid parameter was specified.

User response: Reissue the command with valid parameters.

IOGD0209E The resource bundle cannot be found, or a resource is missing from a resource bundle.

Explanation: The required property files cannot be found. You might run into this error when developing your client application, but the SOAP Gateway property files are not in your classpath during compilation. If the imsbase installation component is tampered, and files in this component is removed or renamed,

User response: Ensure that the *SOAP_Gateway_install_directory*`\imsbase\conf` path is in your client application classpath.

IOGD0210E The *command* command failed because the specified port number (*port_number*) is in use. Please specify an available port in the range of 1-65535.

Explanation: The specified port number is already in use.

User response: Reissue the command to change the port number to a different number that is not already in use.

IOGD0211E The *command* command failed because the specified port number (*port_number*) is invalid. The port number must be in the range of 1 - 65535.

Explanation: The specified port number is not valid.

User response: Reissue the command to change the port number to a valid number that is not already in use.

IOGD0300I List of correlator(s) in the (`master|runtime`) configuration:

Explanation: This message is informational.

User response: No action is required.

IOGD0301I List of correlator entries from correlator file (*correlator_file*) in the (`master|runtime`) configuration:

Explanation: This message is informational.

User response: No action is required.

IOGD0302I Correlator entry for service (*service_name*), operation (*operation_name*) from correlator file (*correlator_file*) in the (`master|runtime`) configuration:

Explanation: This message is informational.

User response: No action is required.

IOGD0303E Correlator entry for service (*service_name*), operation (*operation_name*) from correlator file (*correlator_file*) cannot be found in the (`master|runtime`) configuration.

Explanation: No correlator entry matching the indicated parameters exists in the indicated server configuration.

User response: Ensure that the service name, operation name, and correlator file are correct and then reissue the command.

IOGD0304W No correlator entries were found in the specified correlator file (*correlator_file*) in the (master|runtime) configuration.

Explanation: The specified correlator XML file exists, but does not contain correlator entries.

User response: Add a correlator entry to the correlator XML file by using the `iogmgmt -corr -u` command.

IOGD0305W There are no correlator files in the (master|runtime) configuration.

Explanation: The server's XML directory or runtime cache does not contain any correlator files. No web services or callout applications are deployed.

User response: Deploy a web service or callout application with the `iogmgmt -deploy` command.

IOGD0400E The *command_name* command failed because the specified hostname (*host_name*) is invalid.

Explanation: A hostname must be 1- 256 characters and contain only alphanumeric characters and the special characters '.' and '-'. The name must match the TCP/IP hostname of the target IMS Connect.

User response: Reissue the command with a valid host name.

IOGD0401E The *command_name* command failed because the specified callout basic authentication user ID (*user_ID*) is invalid.

Explanation: A basic authentication user ID must be 1 - 255 characters.

User response: Reissue the command with a valid ID.

IOGD0402E The *command_name* command failed because the specified callout basic authentication password (*password*) is invalid.

Explanation: A basic authentication password must be 1 - 255 characters.

User response: Reissue the command with a valid password.

IOGD0403E The *command_name* command failed because the required connection bundle name parameter (-n) was not specified or was specified in the wrong position. The *out_of_order_parameter_name* parameter was specified in the position where the -n parameter is required.

Explanation: The `-n` parameter and an associated 1 - 20 character connection bundle entry name without blanks are required as the first pair of arguments specified with the *command_name* command.

User response: Reissue the command with the `-n` parameter and a valid connection bundle name in the correct position.

IOGD0500E The update command failed because the *unsupported_property* property is not supported by this correlator type (*current_correlator_type*). The *unsupported_property* property is only supported by the following type of correlator: *other_correlator_type*.

Explanation: The correlator you are updating does not support the property you are trying to update.

User response: Reissue the command without the unsupported property.

IOGD0501I The changes made with the update command were successfully applied to the following correlator file: *file_name*

Explanation: This message is informational.

User response: No action is required.

IOGD0502W The update command successfully changed the master configuration of the service with the following correlator name:
(*correlator_name***)**
However, the following error blocked the update to the runtime configuration:
error_details

Explanation: This is normal if the runtime server is not up and running. The changes made to the service will take effect after the server is started.

User response: Start the SOAP Gateway server for the changes to be made in the runtime configuration.

IOGD0504W The update correlator command successfully updated *correlator_name* **with a new connection bundle name or callout connection bundle name in the master configuration with the following parameters:**
parameter_list

Explanation: Connection bundle properties and references are protected in the active server configuration to ensure stability. The old connection bundle name will be replaced with the new connection bundle name the next time the SOAP Gateway server starts.

User response: Restart the SOAP Gateway server.

IOGD0505W The update correlator command successfully updated *correlator_name* **with the following parameters:**
parameter_list
However, the connection bundle name was only updated in the master configuration.

Explanation: Connection bundle names are protected in the active server configuration to ensure stability. The old connection bundle name will be replaced with the new connection bundle name the next time the SOAP Gateway server starts. The other properties were updated successfully.

User response: Restart the SOAP Gateway server.

IOGD0506E The *command_name* **command failed because the specified correlator file name (***file_name***) already exists. Correlator file names must be unique within a SOAP Gateway instance.**

Explanation: Correlator file names are used to identify individual correlators and so must be unique.

User response: Reissue the command with a unique correlator file name.

IOGD0600I SOAP Gateway is starting the callout thread pool.

Explanation: This message is informational.

User response: No action is required.

IOGD0601I SOAP Gateway is stopping the callout thread pool.

Explanation: This message is informational.

User response: No action is required.

IOGD0602I Callout thread status:
thread_status

Explanation: This message is informational.

User response: No action is required.

IOGD0603I The callout property values in the current SOAP Gateway master configuration are:
callout_properties

Explanation: This message is informational.

User response: No action is required.

IOGD0604I The SOAP Gateway property values in the current SOAP Gateway master configuration are: *server_properties*

Explanation: This message is informational.

User response: No action is required.

IOGD0605I The status of the thread pool cache was written to: *file_location*

Explanation: This message is informational.

User response: No action is required.

IOGD0606I The connection bundle XML file contains the following connection bundle entries: *connbundle_list*

Explanation: This message is informational.

User response: No action is required.

IOGD0607I The requested connection bundle entry *connbundle_name* **contains the following property values:** *property_list*

Explanation: This message is informational.

User response: No action is required.

IOGD0650W The trace level can only be set to OFF for z/OS servers.

Explanation: On other platforms, a command to set the trace level to OFF (trace level 0) is ignored.

User response: Reissue the command with a supported trace level.

IOGD0651E The *command_name* **command failed because the specified property value (***property_value***) was not found in (***file_name***).**

Explanation: The specified property does not exist in the expected file.

User response: Reinstall SOAP Gateway.

IOGD0652W The value given for (*property_name***) was already set to (***property_value***). The file (***configuration_file***) was not updated.**

Explanation: This property value cannot be changed unless the current value is disabled first.

User response: Disable the current value and then enable the new value.

IOGD0653W The command to set *property_name* **to** *property_value* **did not update the runtime configuration because the utility was unable to connect to the server.**

Explanation: Either the server is stopped or an unrelated error blocked the update to the runtime configuration.

User response: Start or restart the server.

IOGD0655W This *command_name* **command is deprecated. See the SOAP Gateway management utility command reference information for a list of valid commands.**

Explanation: The command identified in this message is deprecated.

User response: See the SOAP Gateway management utility command reference for the supported parameters. If the command is iogmgmt -callout -updateprop, the discard pending messages on error (-8) option is deprecated, and replaced by the graceful shutdown command and forced shutdown command.

IOGD0656W This *command_name* **command is deprecated. Use the -force option for immediate shutdown of the thread pool. To stop the server immediately, use the -force option on non-z/OS platforms, and the CANCEL command on z/OS. See the SOAP Gateway management utility command reference information for a list of valid commands.**

Explanation: The iogmgmt -callout –updateprop -9 *discard_pending_messages_on_shutdown* command is deprecated. By default:

- The iogmgmt -callout -stoppool command would stop all worker threads before stopping the thread pool. To stop the thread pool immediately, use the iogmgmt -callout -stoppool -force command.
- The iogmgmt -stop command for the distributed platforms, or the STOP AEWIOGPR command on z/OS, would attempt to process all in-flight messages before shutting down. To discard in-flight messages and shut down the server immediately, use the iogmgmt -stop -force command for the distributed platforms, or CANCAL AEWIOGPR on z/OS.

User response: See "-callout -stoppool: Stop the thread pool" on page 433 and "-stop: Stop the SOAP Gateway server" on page 456 for more information.

IOGD0700W The *command_name* **command successfully changed the SOAP Gateway master configuration, but could not update the runtime configuration because the server connection failed with the following error:**
(*error_details***)**

Explanation: Your service will be available the next time the SOAP Gateway server starts.

User response: Restart the server. Examine the error details to determine whether the underlying error requires corrective action.

IOGD0701W The *command_name* **command successfully changed the SOAP Gateway master configuration, but could not update the runtime configuration because of the following error:**
(*error_details***).**

Explanation: An error was passed from another component to the SOAP Gateway management utility during the update to the runtime configuration.

User response: Restart the SOAP Gateway server to propagate changes from the master configuration to the runtime configuration. Examine the error details to determine whether the underlying error requires corrective action.

IOGD0702E The deploy command failed because the *file_type* **file (***file_name***) could not be copied into the (***file_path***) directory.**
(*error_details***)**

Explanation: An underlying error occurred that requires corrective action.

User response: Examine the error details to determine what corrective action is required.

IOGD0703E The deploy command failed because the service file (*file_name***) could not be created.**
(*error_details***)**

Explanation: An underlying error occurred that requires corrective action.

User response: Examine the error details to determine what corrective action is required.

IOGD0704E The deploy command failed because it failed to set the (*file_name***) file with the** *permission_type* **permission.**

Explanation: The SOAP Gateway management utility does not have the required permission type on the specified file.

User response: Add the required permission and then deploy the web service.

IOGD0705E The deploy command failed with the following reason: *error_details*.

Explanation: An underlying error occurred during deployment.

User response: Examine the error details for more information.

IOGD0706E The deploy command failed because the specified schema location (*schema_path*) in the WSDL file is not supported.

Explanation: The SOAP Gateway management utility could not deploy the web service because the specified WSDL file contains an unsupported XSD import location in the `schemalocation` attribute of the `xsd:import` element. Absolute paths and paths starting with (.) are not supported.

User response: Recreate the WSDL file with a valid schema import location and then deploy the web service by reissuing the command.

IOGD0707E The deploy command failed because the service (*service_name*) is already deployed. (*error_details*)

Explanation: The web service name already exists in the server runtime configuration.

User response: Undeploy the existing web service and then deploy the new web service.

IOGD0708E The deploy command failed because the specified schema file (*schema_file_name*) can not be opened for reading.

Explanation: An XSD import statement in the WSDL file for the service being deployed could not be resolved. This error occurs either because the target XSD file did not exist or because the import statement was not properly formatted.

User response: Ensure that the target XSD file exists and that the import statement in the WSDL is properly formatted. XSD import statements cannot contain absolute paths or paths starting with a period. The target XSD, and all XSDs imported by the target XSD, must be in the same directory as the WSDL when the web service is deployed.

IOGD0709E The deploy command failed because the specified truststore type (*truststore_type*) is not supported. The supported truststore types are JCEKS, JKS and PKCS12.

Explanation: The specified truststore type is not supported.

User response: Reissue the iogmgmt -deploy command with a valid truststore type.

IOGD0710E The deploy command failed because the binding file (*binding_file*) could not be updated with the specified truststore type (*truststore_type*), truststore password, and truststore path (*truststore_path*).

Explanation: The binding file cannot be parsed or updated. The file might not exist, or might be invalid or in incorrect encoding.

User response: Check that the server binding file at the reported location exists, and was not manually altered. Check that the file is in correct encoding, and ensure that the user issuing this command has the permission to update the binding file in the installed SOAP Gateway instance. Reissue the iogmgmt -deploy command after the problems are addressed.

IOGD0711E The deploy command failed because one of the specified options (*option*) is not valid with the specified security token type (*token_type*). The specified token type requires the following parameters: (*parameter_list*).

Explanation: The identified option that was specified with the deployment is not valid. If the token type specified is not correct, an IOGD0709E message was issued. If the password is not valid, an IOGD0053E message was issued. If the truststore path is not valid, and IOGD0105E message was issued.

User response: Check for the occurrence of the related messages to identify which parameter was incorrectly specified. Reissue the iogmgmt -deploy command with the correct parameter values.

"IOGD0053E" on page 383
The *command_name* command failed because the specified password (*specified_password*) is invalid.

"IOGD0105E" on page 388
File name (*file_name*) must have the *extension_type* extension. Provide a valid *file_type* file name with an *extension_type* extension, such as *example_file_name*.

"IOGD0709E" on page 398
The deploy command failed because the specified truststore type (*truststore_type*) is not supported. The supported truststore types are JCEKS, JKS and PKCS12.

IOGD0712W The deploy command ignored the following options: (*options_list***) The specified token type (***type***) does not use those parameters.**

Explanation: The deploy command was processed, but some parameters were ignored because the specified security token type does not use them.

User response: Ensure that the security token type was correctly specified with the iogmgmt -deploy command.

IOGD0750I The undeploy command successfully undeployed the service with correlator (*correlator_file_name***, *artifact_list*) from the SOAP Gateway master and runtime configurations.**

Explanation: This message is informational.

User response: No action is required.

IOGD0751W The undeploy command successfully removed the service associated with *correlator_file_name* **from the SOAP Gateway master configuration. However, it encountered the following error(s):**
additional_error(s)

Explanation: The SOAP Gateway management utility successfully undeployed the web service, but some manual cleanup of files may be required. Any of the following errors may appear in place of *additional_error* in the message text:

File(s) (*file_name_list***) cannot be removed and need to be removed manually.**
One or more web service files could not be deleted and must be removed from the server file system manually.

Unable to connect to the runtime server (*error_details***).**
The runtime configuration could not be updated, but will be the next time the server starts.

Unable to remove from the runtime configuration because of (*error_details***). See the runtime log for additional error details.**
An error occurred when the SOAP Gateway management utility attempted to undeploy the web service from the runtime cache. Check the server log for more details.

Unable to remove from the runtime configuration because the service name information cannot be determined from the correlator file.
The specified correlator file did not contain a unique web service identifier that could be matched to a deployed web service in the runtime configuration. Restarting the server will undeploy the service when runtime cache is updated from the master configuration.

User response: If the server is stopped, no action is required. If the server is started, restart the server.

IOGD0752E The undeploy command is unable to remove the service (*service_name***) from the SOAP Gateway server. The correlator file does not contain the associated WSDL and service information.**

Explanation: The correlator file does not contain the specified WSDL and service information.

User response: Restart the server.

IOGD0753E The undeploy command failed because the path of the specified correlator file (*correlator_name*) is not in the SOAP Gateway XML directory.

Explanation: The correlator file must be in the XML directory. Correlator files outside the XML directory are not used by the server and are not associated with any deployed web services.

User response: Confirm the name of the correlator XML file and reissue the command.

IOGD0758E Execution of the batch command from a batch file is not supported.

Explanation: The batch file can contain any number of SOAP Gateway management utility commands, except the batch command itself.

User response: Remove the batch command from the batch file.

IOGD0759E The *management_utility_command* command failed during the batch call with the following error: *error_details*.

Explanation: The message is issued when the batch command encounters an issue running the *management_utility_command*, followed by the actual error that is generated by the *management_utility_command*.

User response: `AbatchFail.timestamp.txt` file is generated with all the commands that failed to run. Examine the errors, correct the problems, and rerun the iogmgmt -batch command with the corrected file.

IOGD0760E The batch command stopped because the specified batch file *path_to_file_name* could not be opened or found.

Explanation: The specified file is likely corrupted, in an incorrect encoding, or non-existent.

User response: Check that the path to the batch file, including the file name, is correct, and the file is in the correct encoding.

IOGD0811I The *command_name* command has successfully updated the SOAP Gateway environment configuration. The changes are in effect for all Java instances and will be applied to the server runtime the next time that the SOAP Gateway server starts.

Explanation: The Java Runtime Environment configuration for the server has been modified.

User response: No action is required.

IOGD0812I The *JRE_property* value for *property_name* is: *value*.

Explanation: This message displays the configuration details for the Java Runtime Environment currently in use by the server. It is issued in response to the `iogmgmt -view -java` command. If the command is issued with the `-i` or `-h` parameters, only the IFA setting or Java home directory setting are displayed. If the command is issued with the `-a` parameter, all Java settings are displayed.

User response: No action is required.

IOGD0813I The *command_name* command successfully changed the SOAP Gateway environment configuration. The SOAP Gateway server runtime is updated.

Explanation: The indicated command succeeded, and the changes are now active for the server.

User response: No action is required.

IOGD0814E The *command_name* command failed because an invalid file path (*path*) was specified. Ensure that the file path is correct. Reissue the command with a valid fully qualified path.

Explanation: The specified file path does not exist, or is not accessible to the utility.

User response: Ensure that the specified path exists and that the SOAP Gateway management utility has read and write permissions for the target directory.

IOGD0815E The *command_name* command failed because the specified ID type *id_type* does not accept one of the specified properties *invalid_property*. Reissue the command with the *correct_type* ID type.

Explanation: The indicated parameter is not valid with the specified ID type.

User response: Reissue the iogmgmt -tracking command with a different ID type, or reissue the command without the invalid parameter.

IOGD0816W The *command_name* command was ignored because the server configuration is already set to the specified values.

Explanation: The indicated command did not specify any parameter values that are different from the current server configuration.

User response: Use the iogmgmt -view -sgp command to verify that the server is configured correctly.

IOGIM messages

IOGIM messages are related to the use of IBM Installation Manager to install SOAP Gateway.

IOGIM001E *Package_name* requires package(s) *package_list* to be selected to complete the *installation* | *uninstallation*.

Explanation: All three parts of SOAP Gateway must be selected to be installed or uninstalled at the same time. This separation of parts allows you to install each part to a different location for installation flexibility and ease of maintenance. However, all three parts must be installed or uninstalled at the same time.

User response: Return to the previous step to select all three parts.

IOGIM002E The *package_name* package must be installed into the *package_directory_name* directory.

Explanation: You can specify your custom path to install the *package_name* package, but the last directory must be named *package_directory_name*. Directories that do not exist will be created.

User response: Return to the previous step, and name the last directory *package_directory_name*. The *package_directory_name* is either `imsserver`, `imsbase`, or `imssoap`, depending on the associated package that causes this error.

IOGS messages

IOGS messages are returned from the SOAP server.

IOGS001E Invalid SOAP URN [*urnname*] in SOAP message.

Explanation: The SOAP action URN in the client request does not match the SOAP action URNs of any of the deployed web services. This could happen because the web service has not been successfully deployed to the SOAP Gateway. This could also happen if the correct WSDL file is not used for proxy code generation.

User response: The following actions can be taken to address the problem:

1. Ensure that the web Service has been successfully deployed to the SOAP Gateway.
2. Ensure that the correct WSDL file is used for proxy code generation. The SOAPAction element in the WSDL file must match the URN in the client request message and the name of the correlator file for the web service.

IOGS023E The IMS Connect Port Number property value [*value*] is invalid.

Explanation: The value of the IMS Connect port number property is invalid. For example, the value is not an integer.

User response: Examine the IMS Connect port number property value in the connection bundle properties of the web service and ensure that it is a valid integer value.

IOGS024E The Socket Timeout property value [*value*] is invalid.

Explanation: The value of the Socket Timeout property is invalid. For example, the value is not an integer.

User response: Examine the Socket Timeout property value in the correlator properties of the web service and ensure that it is a valid integer value.

IOGS025E Error processing the output message from IMS. A null value was returned.

Explanation: SOAP Gateway received an empty message from IMS.

User response: Look at the SOAP Gateway and IMS traces to determine the problem. Ensure that IMS returns an output message.

IOGS026E The Execution Timeout property value [*value*] is invalid.

Explanation: The value of the Execution Timeout property is invalid. For example, it is not an integer.

User response: Examine the Execution Timeout property value in the correlator file of the web service and ensure that it is a valid integer value.

IOGS027E Error processing output message. [*error_message*].

Explanation: An error occurred when processing the IMS™ output message. Possible reasons are as follows:

- The output message that was returned from the IMS application is not a valid XML document.
- The XML document that was returned from the IMS application does not match the schema definition of the output message that is defined in the WSDL document of the web service.

User response: Examine the output message from the IMS application and verify the following:

- Ensure that the output message is a valid XML document. For example, make sure that the output data is enclosed in the appropriate matching XML tags.
- Ensure the output XML message is encoded with the code page supported by SOAP Gateway.
- Ensure that the output XML message matches the schema definition of the output message in the WSDL file. If it does not match, you can either correct the IMS application or correct the WSDL file. If you choose to correct the WSDL file, be sure to deploy the corrected WSDL file to SOAP Gateway.

IOGS028E SOAP Gateway cannot find the SG_HOME_DIR environment variable. Property represents the XML directory path under SG_HOME_DIR with a trailing file separator. If no SG_HOME_DIR system property is detected at run time, the default is used: *install_dir***/imssoap**

Explanation: The SOAP Gateway home directory is not found. The variables might have been manually modified.

User response: Set the environment variable SG_INSTALL_DIR to where the installation directory is.

IOGS029E Connection bundle name [*connbundle*] not found in connbundle file.

Explanation: The value of the Connection Bundle name property specified in the correlator properties of the web service is not found.

User response: Change the connection bundle name in the correlator properties for the web Service to match one that is properly defined in the connection bundle properties for SOAP Gateway.

IOGS030E Correlator file [*correlator_filename*] for SOAP URN [*urn*] cannot be found.

Explanation: The correlator file for the SOAP request is not found. Possible reasons are:

- The correlator file has not been created successfully when the web service is deployed.
- The correlator file has been removed from SOAP Gateway.

User response: Create the correlator file again for the web service by using the SOAP Gateway management utility.

IOGS031E A value must be specified for the keystore password.

Explanation: The keystore password is not specified when a keystore name is provided in the connection bundle.

User response: Use the SOAP Gateway management utility -conn command to update this information in the connection bundle.

IOGS032E A value must be specified for the truststore password.

Explanation: The truststore password is not specified when a truststore name is provided in the connection bundle.

User response: Use the SOAP Gateway management utility -conn command to update this information in the connection bundle.

IOGS033E A value must be specified for the keystore name.

Explanation: The keystore name is not specified when a keystore password is provided in the connection bundle.

User response: Use the SOAP Gateway management utility -conn command to update this information in the connection bundle.

IOGS034E A value must be specified for the truststore name.

Explanation: The truststore name is not specified when a truststore password is provided in the connection bundle.

User response: Use the SOAP Gateway management utility -conn command to update this information in the connection bundle.

IOGS035E The encryption type is not valid. Valid values are strong, weak, or none.

Explanation: A valid encryption type must be specified for SSL communications with IMS Connect.

User response: Use the SOAP Gateway management utility -conn command to update this information in the connection bundle.

IOGS036E A value must be specified for the truststore name and password, or keystore name and password.

Explanation: An encryption type is specified, but the truststore and keystore information is missing.

User response: Use the SOAP Gateway management utility-conn command to update this information in the connection bundle.

IOGS037E The keystore password *password* is not valid. It must be 6 to 20 characters in length.

Explanation: The password might have been incorrectly modified.

User response: Use the SOAP Gateway management utility-conn command to correct this information in the connection bundle.

IOGS038E The truststore password *password* is not valid. It must be 6 to 20 characters in length.

Explanation: The password information might have been incorrectly modified.

User response: Use the SOAP Gateway management utility-conn command to correct this information in the connection bundle.

IOGS041E The callout request message cannot be processed. [*error_message*]

Explanation: SOAP Gateway cannot process the callout request message. The *error_message* further explains the cause of the error.

User response:

Error message	Cause	User response
install_dir/imssoap/xml/correlator_file.xml (The system cannot find the file specified).	The correlator file for the callout request cannot be found.	Ensure that the correlator file for the callout web service is deployed correctly in the correlator file directory.
IOGX019E: The XML document cannot be parsed. [Error: URI=null Line=1: cvc-complex-type.2.4.a: Invalid content was found starting with element "IMS:Namespace". One of "{"http://www.ibm.com/IMS/Callout":WSID}" is expected	The callout message is not in the appropriate format. For example, the WSID tag in the callout service data is not in the appropriate XML format.	Ensure that the callout request message conforms to the callout request message schema. Correct the callout request message in your IMS application accordingly.

Related concepts:

"Preparing callout messages" on page 236
If you are not using the IMS Connect XML adapter function to convert the data between bytes and XML, you must ensure that the callout message from your IMS application is in a valid XML format.

IOGS042E SOAP Gateway encountered an error while creating the callout request message. The *error_message* **further explains the cause of the error.**

Explanation: SOAP Gateway encountered an error while creating the callout request message. The *error_message* further explains the cause of the error.

User response:

Eror message	Cause	User response
Error processing WSDL document: javax.xml.rpc.ServiceException: Cannot find service: *{service_namespace}service_name*	The web service operation name associated with the service, port, and namespace value cannot be found in the WSDL file.	Ensure that the valid service, port, namespace and operation name of the callout web service is specified in the service data for the callout request message. If the IMS Connect XML adapter is used, make sure the correct values are specified when you generate the XML converters by using Rational Developer for System z. Otherwise, correct the values in the callout request message in your IMS application.
Error processing WSDL document: java.io.FileNotFoundException: C:\Program Files\IBM\IMS Enterprise Suite V*x.x*\SOAP Gateway\server\webapps\imssoap\wsdl*wsdl_file.xml* (The system cannot find the file specified).	The WSDL file for the callout web service cannot be found.	Ensure that the WSDL file *wsdl_file.xml* for the callout web service is deployed correctly in the wsdl directory.

IOGS043E SOAP Gateway cannot invoke the web service. [*error_message***].**

Explanation: The callout web service cannot be invoked. The *error_message* further explains the cause of the error.

User response:

Eror message	Cause	User response
java.lang.IllegalArgumentException: invalid QName local part	The callout web service cannot find the QName of the local part for the input message. This error might be caused by missing payload data.	Make sure that the payload data is specified in the callout request message.

IOGS044E Invalid service data in callout request message. [*error_message***]**

Explanation: The service data in the callout request message is not valid. The *error_message* further explains the cause of the error.

User response:

Error message	Cause	User response
A WSID value is required.	No web service identifier (WSID) value is specified. A WSID value must be specified in the service data prefix of the callout request message.	Specify a WSID value in the service data prefix of the callout request message. If the IMS Connect XML adapter is used, ensure the correct values are specified when you generate the XML converters by using Rational Developer for System z. Otherwise, correct the values in the callout request message in your IMS application.
An operation name value is required.	No operation name value is specified. An operation name value must be specified in the service data prefix of the callout request message.	Specify an operation name value in the service data prefix of the callout request message. If the IMS Connect XML Adapter is used, ensure that the correct values are specified when you generate the XML converters by using Rational Developer for System z. Otherwise, correct the values in the callout request message in your IMS application.
A port name value is required.	No port name value is specified. A port name value must be specified in the service data prefix of the callout request message.	Specify a port name value in the service data prefix of the callout request message. If the IMS Connect XML adapter is used, ensure that the correct values are specified when you generate the XML converters by using Rational Developer for System z. Otherwise, correct the values in the callout request message in your IMS application.
A service name value is required.	No service name value is specified. A service name value must be specified in the service data prefix of the callout request message.	Specify a service name value in the service data prefix of the callout request message. If the IMS Connect XML adapter is used, ensure that the correct values are specified when you generate the XML converters by using Rational Developer for System z. Otherwise, correct the values in the callout request message in your IMS application.

Error message	Cause	User response
A namespace value is required.	No namespace value is specified. A namespace value must be specified in the service data prefix of the callout request message.	Specify a namespace value in the service data prefix of the callout request message. If the IMS Connect XML adapter is used, ensure that the correct values are specified when you generate the XML converters by using Rational Developer for System z. Otherwise, correct the values in the callout request message in your IMS application.

Related concepts:

"Preparing callout messages" on page 236
If you are not using the IMS Connect XML adapter function to convert the data between bytes and XML, you must ensure that the callout message from your IMS application is in a valid XML format.

IOGS045E WSDL file name is required in callout correlator file [*correlator_file.xml***].**

Explanation: The WSDL file name for the callout web service is not specified in the correlator file.

User response: Specify the WSDL file name for the callout web service in the correlator file *correlator_file.xml*. Use the SOAP Gateway management utility -callout -corr command.

IOGS061E SOAP Gateway is unable to send outbound message to server *serverURL***.** *error_message*

Explanation: SOAP Gateway gets an error when sending out the outbound event message to the server. The *error_message* further explains the cause of the error.

User response: Either the remote server at *serverURL* is not up and running, the *serverURL* is not specified correctly, or the user name and password for basic authentication failed. Ensure that the remote server is up and running, the callout location URI is specified correctly, or the user name and password information is correct.

- The following error message indicates that the HTTP protocol is missing in the callout location URI correlator property:

  ```
  com.ibm.ims.soap.server.IMSSOAPException: IOGS061E: SOAP Gateway is unable to send outbound message
  to server [/rest/bpm/events].
  ```

- The following error message indicates that the server that is specified in the callout location URI value is either incorrect, or the server is down.

  ```
  com.ibm.ims.soap.server.IMSSOAPException: IOGS061E: SOAP Gateway is unable to send outbound message
  to server [http://localhost:9081/rest/bpm/events].
  ```

- When basic authentication fails, the server might return the following error:

  ```
  ERROR: com.ibm.ims.soap.server.IMSSOAPException: IOGS061E: SOAP Gateway is unable to send outbound
  message to server [https://localhost:9443/rest/bpm/events]. [java.io.IOException: Server returned
  HTTP response code: 401 for URL: https://localhost:9443/rest/bpm/events]
  ```

IOGS501E SOAP Gateway could not find the web service to invoke. Target web service is [*service_name***].**

Explanation: SOAP Gateway could not find the web service to invoke. The web service *service_name* is not deployed to SOAP Gateway.

User response: Ensure the web service *service_name* is deployed to SOAP Gateway. Check the SOAP Gateway server console to make sure that the web service is deployed.

Also, verify that the client is sending the correct name of the web service. The web service name is the name attribute value of the port element in the WSDL file.

IOGS502E SOAP Gateway cannot find the operation [*operation_name*].

Explanation: SOAP Gateway could not find the operation *operation_name* for the web service. Possible reasons are:

- The web service with the associated operation has not been successfully deployed to the SOAP Gateway.
- The client is not built correctly to invoke the web service. For example, an incorrect WSDL file is used to build the client application.

User response: Ensure that the web service deployed to the SOAP Gateway has the operation defined. The value can be verified by looking at the WSDL file. Verify the client is sending the correct operation name.

IOGS503E The envelope namespace of the SOAP message is invalid: [*namespace_name*]

Explanation: The envelope namespace *namespace_name* of the SOAP message is not valid.

User response: Ensure that the client is sending a valid SOAP message to SOAP Gateway.

IOGS504E The envelope tag of the SOAP message is invalid: [*tag_name*]

Explanation: The envelope tag name of the SOAP message is not valid.

User response: Ensure that the client is sending a valid SOAP message to SOAP Gateway.

IOGS505E The SOAP message does not contain a body element.

Explanation: The body part is not found in the SOAP message.

User response: Ensure that the client is sending a valid SOAP message that contains the body element with the input data to SOAP Gateway.

IOGS0000E SOAP Gateway has detected a Java stack trace useful for technical support. *Error_details*.

Explanation: Possible error details are:

- The stack trace is written to the log.
- The stack trace cannot be written to the log because the logfile appender is set to OFF. Turn on the logfile appender and try to recreate the problem so the stack trace could be written to the log.
- The stack trace is: *stack_trace_details*

User response: If the stack trace is written to the log, examine the `imssoap.log` file in the *`install_dir`*`/imsbase/logs` directory.

If the stack trace is not written to the log, use the iogmgmt -prop command to set the trace level to 2 (ERROR) or above:

```
iogmgmt -prop -u -l 5
```

IOGS0077E An error occurred during the invocation of the external web service: [*error_message*].

Explanation: The external web service is invoked, but an error occurs. The cause might be a network error, TCP/IP issue, or other problems.

User response: Check the error text that is returned to the waiting IMS application that issued the callout request. Correct the reported issue.

IOGS0078E Client authentication is required for WS-Security SAML confirmation method.

Explanation: Client authentication is required for callout WS-Security SAML token support.

User response: Configure your system to support client (mutual) authentication between the SOAP Gateway client and the external web service server.

IOGS0079E An Axis fault occurred. The WS-Security SAML token cannot be generated. *Error_details*.

Explanation: The JVM system is in an unhealthy state, such as out of memory, and is unable to instantiate the classes to generate the SAML token.

User response: Check the health of the JVM system. Use the SOAP Gateway monitoring mbean to obtain JVM health information.

Related tasks:

"Configuring the SOAP Gateway monitoring MBean" on page 345
The SOAP Gateway MBean interfaces provide statistics about the SOAP Gateway server and deployed web services (provider scenario).

IOGS0080E The WS-Security feature is for synchronous callout applications only. Asynchronous callout or business event scenarios are not supported.

Explanation: WS-Security is not supported for asynchronous callout or business event scenario. The user information required for WS-Security is extracted from the correlation token that is generated for synchronous callout applications only.

User response: Either modify the application to be a synchronous callout application, or redeploy the asynchronous callout application or business event application without specifying a SAML token type.

IOGS0081I SOAP Gateway received a positive acknowledgement (ACK) from IMS for the following callout application: service *service_name*, **operation** *operation_name*, **and target namespace** *namespace_name*.

Explanation: A positive acknowledgement was received for a callout application that is configured to use the send-only with acknowledgement protocol.

In the message text:

service_name
The service name of the callout application.

operation_name
The operation name of the callout application.

namespace_name
The target namespace of the callout application.

User response: No action is required.

IOGS0082E SOAP Gateway received a negative acknowledgement (NACK) from IMS for the following callout application: service*service_name*, **operation** *operation_name*, **target namespace** *namespace_name*. **The reason for the NACK response was:** *error_text*.

Explanation: A negative acknowledgement was received for a callout application that is configured to use the send-only with acknowledgement protocol.

In the message text:

service_name
The service name of the callout application.

operation_name
The operation name of the callout application.

namespace_name
The target namespace of the callout application.

error_text
The error from IMS that caused the NACK response.

User response: This error indicates that the callout response message did not reach the IMS application that issued the original callout message. Typically, this error is caused by a communication problem or error in IMS Connect. Examine the error text to determine the underlying cause of the problem.

IOGS0083E SOAP Gateway encountered an error while trying to send a callout response message for the following callout application: service *service_name*, **operation** *operation_name*, **and target namespace** *namespace_name*. **The reason for the error was:** *error_text*.

Explanation: SOAP Gateway failed to send a response to a callout request for an application that is configured to use the send-only with acknowledgement protocol.

In the message text:

service_name
The service name of the callout application.

operation_name
The operation name of the callout application.

namespace_name
The target namespace of the callout application.

error_text
Information about the underlying error.

User response: This error indicates that the callout response message was not sent to the IMS application that issued the original callout request message. Typically, this error is caused by an internal error in SOAP Gateway. Examine the error text to determine the underlying cause of the problem.

IOGS0102E The resource bundle cannot be found, or a resource is missing from a resource bundle.

Explanation: Either the resource bundle file is accidentally manually deleted, or SOAP Gateway was not correctly installed.

User response: Reinstall SOAP Gateway. If the problem continues, contact IBM Software Support.

IOGS0103E SOAP Gateway was not able to detect the operating system. SOAP Gateway must shut down. Ensure that the server is installed and run on a supported operating system: z/OS, Linux on System z, Windows, and AIX.

Explanation: This SOAP Gateway server instance is not installed on a supported platform.

User response: Install SOAP Gateway on a supported platform.

IOGS0104E An I/O exception occurred with the (*full_path_to_file***) file. Either the file could not be read or found, or the file encoding is not UTF-8. Message=** *error_message*.

Explanation: The named file cannot be read or found, or the file encoding is incorrect. The error messages might be:

- SOAP Gateway was unable to load file into memory and must read from the file system.
- SOAP Gateway is unable to configure the logging facility.

User response: Check that the identified file exists and is in UTF-8 encoding.

IOGS0105E SOAP Gateway was unable to configure *SG_INSTALL_DIR*. **The path set for** *SG_INSTALL_DIR* **might not be a directory, or was not configured. For z/OS,** *SG_INSTALL_DIR* **is configured in the configuration member. For distributed platforms, it is configured in iogstart.sh or iogstart.bat in the bin\ directory. SOAP Gateway must shut down.**

Explanation: The SOAP Gateway *SG_INSTALL_DIR* directory path variable is not properly configured.

User response: Check the `iogstart.sh` or `iogstart.bat` file in the `bin\` directory on distributed platforms. They might have been manually modified incorrectly after the initial installation. For z/OS, check the configuration member.

IOGS0106E A null path name argument occurred. Path name = *path_name*; Value = *argument_value*. SOAP Gateway must shut down. Please contact IBM Software Support.

Explanation: The value for the path name argument is null and SOAP Gateway cannot run. Most likely the `iogstart.sh` or `iogstart.bat` file for the distributed platform was incorrectly modified.

User response: Contact IBM Software Support and send the following files:

- The `catalina.sh` or `catalina.bat` file in the *`install_dir`*`/imsserver/server/bin` directory
- Any log files

IOGS0107E The message context is null. The service request cannot be processed.

Explanation: The received message has no context, most likely because a non-existent service was invoked.

User response: Contact IBM Software Support, and provide the log file. If the service exists and is successfully deployed, provide the service artifacts.

IOGS0108E The *web_service_attribute* value cannot be determined from the message context. The service might be incorrectly deployed. In the administrative console (http://ip_address:port/imssoap), click View Services, and then click the service link to see if the WSDL is available for browsing.

Explanation: Either the operation name, service name, or target namespace value is missing from the message context. The service cannot be identified.

User response: Check the web service is deployed and running by using the SOAP Gateway administrative console. If the web service is not running, check the correlator file and the WSDL file. Redeploy the web service.

IOGS0110E The service identifier (*server_identifier*) was not found in the runtime cache.

Explanation: The reported service identifier (consists of the target namespace, service name, and operation name) uniquely identifies the web service. The requested web service is either not yet deployed or invalid.

User response: Check to see that the web service is deployed by using the SOAP Gateway administrative console. Use the SOAP Gateway management utility iogmgmt -view correlatorfile ALL command. SOAP Gateway must be running in order to view all correlators in the runtime cache for deployed web services.

Related reference:

"-view -correlatorfile: View correlator information" on page 462
Use the -view -correlatorfile command to view correlator information in either the runtime or master configuration.

IOGS0111E The (*correlator_file*) correlator for service identifier (*service_identifier*) was not found in the runtime cache.

Explanation: The unique service identifier consists of the target namespace, service name, and operation name. The information stored in the cache for this identified web service might have been corrupted.

User response: Contact IBM Software Support, and provide the log file from debug mode.

Related tasks:

"Setting the trace level for SOAP Gateway" on page 304
You can turn on internal tracing for SOAP Gateway to help diagnose problems. The trace level can be changed to control the amount of logging.

IOGS0112E The (*WS-Security_server_binding_policy*) XML file that has been loaded into memory is ill-formed. Message = *error_message*.

Explanation: The `policy.xml` file is corrupted. The error message is the XML error thrown from the XML parser, where the ill-formed element is identified with a line number, element name, or reason.

User response: Correct the ill-formed element, and then restart the SOAP Gateway server.

IOGS0113E SOAP Gateway is unable to configure the (*WS-Security_server_binding_policy*) file that has been loaded into memory. Message = *error_message*.

Explanation: The user that starts the SOAP Gateway server might not have the permission to access the WS-Security policy and binding files in the *install_dir*/imssoap/WS-SECURITY/*token_type*/server/ directory. Another possibility is that the policy and binding files were modified and ill-formed.

User response: Check the directory and file permission. Check the `policy.xml` or `bindings.xml` files. Use the provided policy and binding files as samples, and correct any problems. If the XML files are modified, restart the server for the changes to take effect.

IOGS0114E The (*WS-Security_server_binding_policy_in_cache*) file cannot be loaded from memory. SOAP Gateway must default to the (*WS-Security_server_binding_policy__in_filesystem*) file in the file system.

Explanation: The file might have been manually modified and the XML is ill-formed.

User response: Correct the XML file by following the sample policy and binding XML files in the in the *install_dir*/imssoap/WS-SECURITY/*token_type*/server/ directory. Restart the SOAP Gateway server for the changes to take effect.

IOGS0115E SOAP Gateway is unable to configure the AXIS parameter (*parameter*) for the service identifier (*service_identifier*). Message = (*error_message*).

Explanation: SOAP Gateway cannot correctly apply the WS-Security configuration information to the web service.

User response: Check the SOAP Gateway log file at debug mode. Correct the XML file by following the sample policy and binding XML files in the in the *install_dir*/imssoap/WS-SECURITY/*token_type*/server/ directory. Restart the SOAP Gateway server for the changes to take effect.

IOGS0117E SOAP Gateway cannot initialize the properties related to the idle connection cleanup function and must use the default values. The idle connection cleanup function is disabled by default.

Explanation: The server properties associated with the idle connection cleanup function are either missing or in an incorrect format that SOAP Gateway cannot process. The default value (disable idle connection cleanup) will be used.

User response: Issue the following command to reset the property values.

```
iogmgmt -prop -r frequency_in_minutes -v maximum_connection idle_time_in_minutes
-m minimum_number_of_connections_to_keep
```

Related reference:

"-prop: Set SOAP Gateway properties" on page 450
Use the -prop command to modify the SOAP Gateway server properties.

IOGS0118E SOAP Gateway encountered an invalid data type with the idle connection cleanup frequency property. The value must be equal to or greater than 0. The default value of 0 is used, and the idle connection cleanup function is disabled.

Explanation: This error occurs when SOAP Gateway cannot successfully convert the cleanup frequency value specified in the SOAP Gateway server property from a string to an integer.

User response: Issue the following command to reset the value:

```
iogmgmt -prop -r frequency_in_minutes
```

Related reference:

"-prop: Set SOAP Gateway properties" on page 450
Use the -prop command to modify the SOAP Gateway server properties.

IOGS0119E The cleanup frequency in minutes property of the idle connection cleanup function could not be converted to an integer. The default value of 0 is used, and the idle connection cleanup function is disabled.

Explanation: This error occurs when SOAP Gateway cannot successfully convert the cleanup frequency value specified in the SOAP Gateway server property to an integer.

User response: Issue the following command to reset the value:

`iogmgmt -prop -r frequency_in_minutes`

IOGS0120E SOAP Gateway encountered an invalid data type with the idle connection cleanup threshold time property. The value must be greater than 1. The default value of 20 is used.

Explanation: This error occurs when SOAP Gateway cannot successfully convert the idle threshold time specified in the SOAP Gateway server property to an integer.

User response: Issue the following command to reset the value:

`iogmgmt -prop -v connection_idle_time_in_minutes`

IOGS0121E The idle threshold time in minutes property of the idle connection cleanup function could not be converted to an integer. The default value of 20 is used.

Explanation: This error occurs when SOAP Gateway cannot convert the idle threshold time specified in the SOAP Gateway server property to an integer.

User response: Issue the command `iogmgmt -prop -v connection_idle_time_in_minutes` to reset the value.

IOGS0122E SOAP Gateway encountered a data type error with the minimum number of connections to keep for the idle connection cleanup function. The value must be equal to or greater than 0. The default value of 0 is used.

Explanation: This error occurs when SOAP Gateway cannot successfully convert the minimum number of connections specified in the SOAP Gateway server property to an integer.

User response: Issue the command `iogmgmt -prop -m minimum_connections_to_keep` to reset the value.

IOGS0123E The minimum number of connections to keep property of the idle connection cleanup function could not be converted to an integer. The default value of 0 is used.

Explanation: This error occurs when SOAP Gateway cannot successfully convert the minimum number of connections specified in the SOAP Gateway server property to an integer.

User response: Issue the command `iogmgmt -prop -m minimum_connections_to_keep` to reset the value.

IOGS0124E SOAP Gateway could not interrupt the IMS Connect socket that is waiting for callout requests from IMS. Error message: *message_from_IMS***.**

Explanation: SOAP Gateway cannot interrupt the timer on IMS Connect before gracefully shutting down the server. IMS Connect might not be available due to network issues, or IMS Connect is down.

User response: Check the error returned from IMS Connect. Check the state of the Client ID that is associated with the interruption by using the VIEWHWS command on IMS.

IOGS0125E The provider request cannot be serviced because the server is shutting down. Details of the rejected service: *details***.**

Explanation: The server received a graceful shutdown request. All subsequent incoming requests are now blocked. An AxisFault is sent back to the client application with the details about the target namespace and service name of the service that was rejected.

User response: The client application can optionally choose to retry the request after the server is started up.

IOGS0126E The ITCAM TTAPI or SOAP Gateway transaction logger failed to start. Problem initializing = (*component***). Message = (***error_details***).**

Explanation: This error occurs when SOAP Gateway cannot start either the IBM Tivoli Composite Application Manager for Transactions (ITCAM) Transaction Tracking API (TTAPI) or the local transaction logger.

In the message text:

component
The component that failed to start.

error_details
The error details from the specified component.

User response: Resolve the error identified in the message and re-start either the ITCAM TTAPI or the transaction logger.

IOGS0127E Unable to capture a transaction tracking event for vertical ID = (*vertical_ID*). The active tracking component is: (*component*). Exception details: = (*exception*).

Explanation: The IBM Tivoli Composite Application Manager for Transactions (ITCAM) Transaction Tracking API (TTAPI) or SOAP Gateway transaction logger was unable to capture an event for a message with the indicated vertical ID because of an exception.

System action: Processing continues.

User response: Determine whether the problem requires any corrective action by consulting the exception details. If this error occurs more than once, restart the ITCAM TTAPI or transaction logger.

IOGS0128E A transaction tracking system encountered an internal error while processing an event for vertical ID = (*vertical_ID*). Restart (*component_name*)

Explanation: Either the SOAP Gateway transaction logger or the IBM Tivoli Composite Application Manager for Transactions (ITCAM) Transaction Tracking API (TTAPI) unexpectedly shut down because of an internal error. The error occurred while processing a transaction tracking event for the message with the indicated vertical ID.

System action: New transaction tracking events are not written to the local transaction log or sent to the remote data collector. Message processing continues normally.

User response: Restart the transaction logger or the ITCAM TTAPI.

IOGS0130E Encountered an exception during transaction tracking event processing with vertical ID = (*vertical_ID*). Message = (*error_details*).

Explanation: SOAP Gateway encountered an exception while processing an event for the indicated message. In the message text, *error_details* is the message from the transaction tracking subsystem.

System action: The transaction logger or IBM Tivoli Composite Application Manager for Transactions (ITCAM) Transaction Tracking API (TTAPI) are shut down if they are running. Message processing continues normally.

User response: Restart the transaction logger or the ITCAM TTAPI.

IOGS0131E An exception occurred while shutting down a transaction tracking system: (*name*). The exception was: (*exception*).

Explanation: SOAP Gateway caught an exception while shutting down either the local transaction logger or the IBM Tivoli Composite Application Manager for Transactions (ITCAM) Transaction Tracking API (TTAPI).

System action: SOAP Gateway forces the affected component to shut down. If this message is generated during SOAP Gateway server shutdown, the server continues to shut down normally.

User response: Restart the local transaction logger or the ITCAM TTAPI. If the affected component does not start, contact IBM Software Support with the exception details.

IOGS0133E Encountered an exception when processing a request with vertical ID = (*vertical_ID*). Message = (*exception*).

Explanation: SOAP Gateway was unable to continue processing the request with the specified vertical ID. The exception information contains details about the error.

System action: Processing continues.

User response: Examine the exception text to determine the cause of the underlying error.

IOGS0136E The file path string *user_specified_path* contains one or more invalid characters. Special characters, such as *, ?, |, %, and &, are not allowed.

Explanation: The *user_specified_path* cannot contain special characters.

User response: Correct the file path and reissue the command.

IOGS4001I SOAP Gateway is initializing the runtime cache.

Explanation: This is an informational message to log the start of the runtime cache initialization process.

User response: No user action is needed.

IOGS4002I SOAP Gateway has detected that the operating system type is *operating_system*.

Explanation: This is an informational message to log the identified operating system.

User response: No user action is needed.

IOGS4003I SOAP Gateway has configured *SG_directory* to *directory_path*.

Explanation: This is an informational message to log the sequence of setting the various directories for SOAP Gateway installation, Java runtime environment (JRE), web service WSDL files, and SOAP Gateway server properties. *SG_directory* might be:

- SG_HOME_DIR
- SG_JAVA_BIN
- SG_JAVA_JRE
- SG_REGION_NAME
- SG_SOAP_LOG4J_PATH
- SG_SOAP_PROPERTIES_PATH
- SG_XML_DIR_PATH

User response: No user action is needed.

IOGS4004I SOAP Gateway has discovered this list of deployed *web_service_artifacts* files to be loaded into the runtime cache: *list of files*.

Explanation: This is an informational message to log the WSDL or correlator files that are loaded into the runtime cache. *web_service_artifacts* might be “WSDL” or “correlator”.

User response: No user action is needed.

IOGS4005I SOAP Gateway was unable to detect any deployed *web_service_artifacts*. Please review the deployment process. SOAP Gateway is operational.

Explanation: This is an informational message to indicate that no web service artifact files were detected. *web_service_artifacts* might be “artifacts”, “Callout Connection Bundles”, or “Connection Bundles”.

User response: No user action is needed.

IOGS4006I SOAP Gateway has loaded SOAP Gateway server properties into the memory.

Explanation: This is an informational message to log that the server properties have been loaded.

User response: No user action is needed.

IOGS4007I The idle connection cleanup job started at *timestamp*.

Explanation: The idle connection cleanup job started at the reported time.

User response: No user action is needed.

IOGS4008I The idle connection cleanup function is enabled at *timestamp***. The job is scheduled to run with cleanup frequency in minutes =** *cleanup_frequency***, idle threshold time in minutes =** *connection_idle_time***, and minimum number of idle connections to keep =** *minimum_connections***.**

Explanation: The idle connection cleanup function is enabled, and the job is scheduled to run based on the reported parameter values.

User response: No user action is needed.

IOGS4009I The idle connection cleanup function is disabled at *timestamp***.**

Explanation: The idle connection cleanup function is disabled at the reported time. The cleanup schedule is cancelled.

User response: No user action is needed.

IOGS4010I The idle connection cleanup schedule is cancelled at *timestamp***.**

Explanation: The idle connection cleanup schedule is cancelled at the reported time because the server is shutting down.

User response: No user action is needed.

IOGS4011I The idle connection cleanup schedule is already cancelled because the function is disabled at *timestamp***.**

Explanation: This message is logged when the server is shutting down.

User response: No user action is needed.

IOGS4012I The idle connection cleanup statistics for pool *pool_id* **at** *timestamp***: total number of idle connections at the start of cleanup =** *number_of_connection***, number of purged idle connections after cleanup =** *connections_purged***.**

Explanation: This message is for informational purposes only.

User response: No user action is needed.

IOGS4013I SOAP Gateway successfully interrupted the IMS Connect socket that is waiting for callout requests from IMS. The socket is for client ID: *client_ID***.**

Explanation: SOAP Gateway has successfully interrupted the IMS Connect socket that is waiting for callout requests from IMS. When the thread policy is set to "One Thread Per Tpipe," each Client ID that is used in a resume tpipe request and is waiting for a callout message on that particular tpipe would get an informational message for the successful cancellation.

User response: No user action is needed.

IOGS4014I The connection issue with IMS Connect is resolved. Resume tpipe calls will resume for connection bundle *connection_bundle_name* **and tpipe** *tpipe_name***.**

Explanation: SOAP Gateway has successfully resumed connection with IMS Connect and will make a resume tpipe call to pull callout messages.

User response: No user action is needed.

IOGS7001W SOAP Gateway has configured the SG_INSTALL_DIR to *actual_SG_install_directory***. The original path set for SG_INSTALL_DIR either is not a directory or was not configured. For z/OS, SG_INSTALL_DIR is configured in the configuration member. For distributed platforms, it is configured in iogstart.sh or iogstart.bat located in the bin\ directory.**

Explanation: This is a warning message to log that the server path is set to a different directory than what is specified in the configuration member or in the server startup script.

User response: Verify that this difference in the directory settings is not an issue.

IOGS7002W The provider inbound tracking ID (*SG_tracking_id*) in SOAP Gateway is different from the outbound tracking ID from IMS Connect (*IMSCONNECT_tracking_id*).

Explanation: This is a warning message to log that the tracking ID in the inbound service request is different from the outbound tracking ID that is returned from IMS Connect. SOAP Gateway sends the tracking ID to IMS Connect and checks against the tracking ID it receives in the response message from IMS Connect. If the tracking ID that SOAP Gateway issued is different from what it received, there might be race-conditions or concurrency issues in IMS Connect or OTMA.

User response: In the unlikely event when this tracking ID mismatch issue occurs, check the OTMA trace records. Collect IMS Connect recorder traces and SOAP Gateway debug logs before you contact IBM Software Support.

IOGS7003W An incorrectly formed callout request message was encountered for element *element_name*.

Explanation: This is a warning message to log that the callout request message is incorrectly formed.

User response: Check the callout request that IMS Connect sent to IMS. Mostly likely the callout request is incorrectly formed. This error scenario might happen if you are not using the IMS Connect XML adapter function and are sending SOAP Gateway the XML request directly.

Related concepts:

"Preparing callout messages" on page 236
If you are not using the IMS Connect XML adapter function to convert the data between bytes and XML, you must ensure that the callout message from your IMS application is in a valid XML format.

IOGS7004W A value is missing for callout request element *element_name*.

Explanation: This is a warning message to log that an element in the callout request message is missing a value.

User response: Check the callout request that IMS Connect sent to IMS. Mostly likely the callout request is incorrectly formed. This error scenario might happen if you are not using the IMS Connect XML adapter function and are sending SOAP Gateway the XML request directly.

Related concepts:

"Preparing callout messages" on page 236
If you are not using the IMS Connect XML adapter function to convert the data between bytes and XML, you must ensure that the callout message from your IMS application is in a valid XML format.

IOGU messages

IOGU messages are issued by the migration tool that you run to migrate web service files for previous versions of IMS Enterprise Suite SOAP Gateway. The migration tool is provided to ease the upgrade process by migrating web service correlator files and sever configuration XML file.

IOGU0000E SOAP Gateway has detected a Java stack trace useful for technical support. *Error_details.*

Explanation:

Possible error details are:

- The stack trace is written to the log.
- The stack trace cannot be written to the log because no logger has been detected.

User response: This is an unexpected error. Contact IBM Software Support.

IOGU0001E The migration failed because the backup folder cannot be created or accessed at: *file_location*. **Ensure that the user that runs this migration command has the correct permission, and the system is not out of disk space.**

Explanation: The SOAP Gateway management utility iogmgmt -migrate command cannot create or access the backup directory possibly due to permission issues or lack of disk space.

User response: Ensure that the user that issues the command has the proper permission to create or access folders in the *SOAP_Gateway_install_directory*\tools directory. On non-Windows systems, the permission must be set to 755. Check the available disk space.

IOGU0002E The migration process stopped because the correlator files cannot be moved from *original_folder* **to the backup folder at** *backup_folder***. Check if the system is out of disk space.**

Explanation: The migration tool cannot make a copy of all existing correlator files to the backup folder in order to start the migration process.

User response: Check the available disk space. Ensure that the user that runs this migration tool has the appropriate permission to write to the backup directory.

IOGU0003E The migration process stopped because the summary report file (*log_file***) does not exist or is not accessible. Verify that the file name and file path exist and permissions are set properly.**

Explanation: The migration tool creates a summary report file and logs to the file when each correlator file is processed and migrated to the new correlator schema. Possible reasons for this error include lack of disk space, or insufficient permission to write to the file.

User response: Check the available disk space. Ensure that the user that runs this migration tool has the appropriate permission to write to the file and the directory.

IOGU0004E The (*correlator_file***) file cannot be migrated, because the** *file_type file_name* **is not valid. The detail information is** *error_details***.**

Explanation: The reported invalid file can be either a WSDL or XSD file, or a correlator file. The reason is captured in *error_details*.

User response: Correct the problem based on the provided details and rerun the migration tool.

IOGU0005E The migration process stopped because of an invalid field or attribute name *attribute_name* **in the** *file_name* **correlator file.**

Explanation: The reported invalid field or attribute name prevents the migration process from proceeding.

User response: Correct the invalid field or attribute name and rerun the migration tool.

IOGU0006E The migration process stopped because the WSDL file *file_name* **is missing the required attribute,** *attribute_name***.**

Explanation: The migration cannot proceed until the missing required attribute is provided.

User response: Supply the missing attribute and rerun the migration tool.

IOGU0007I The correlator files are backed up from *original_location* **to** *backup_location***.**

Explanation: This message is logged before the migration tool starts migrating the correlator files to the new schema.

User response: No action is required.

IOGU0008I The following *number_of_files* **correlator file(s) did not contain corresponding WSDL or XSD file name(s). These correlator files were not updated:** *list_of_files***.**

Explanation: The migration tool runs successfully, but this is an informational message to report the files that were not migrated because of a missing corresponding WSDL or XSD file name.

User response: Examine the reported list of files. Because related web services would not work without the WSDL or XSD information in the correlator file, these correlator files are most likely incorrectly left in the original correlator directory.

IOGU0009I The following correlator file(s) were updated to the new schema: *list_of_files*.

Explanation: The migration tool runs successfully, and this message reports the list of correlator files that are migrated to the new schema.

User response: No user action is required.

IOGU0010I The following *number_of_files* **correlator file(s) are already in the new schema and remain unchanged:** *list_of_files*.

Explanation: The migration tool runs successfully. The files in this list were already in the new schema, so they are unchanged. This situation occurs, for example, if you rerun the migration tool after you fix a reported issue.

User response: No user action is required.

IOGU0011E *Migration_task* **migration encountered the following error:** *error_details*.

Explanation: The migration task could be web service correlator files migration or the server configuration (`server.xml`) migration.

User response: Contact IBM Software Support with the error details.

IOGU0012I *Migration_task* **migration check completed. [***status_details***]**

Explanation: The migration task could be web service correlator files migration or the server configuration (`server.xml`) migration.

User response: No user action is required.

IOGU0013E The migration process stopped because *file_name* **does not exist on the file system.**

Explanation: The file required for the migration does not exist. The file or the directory it is in might be accidentally deleted.

User response: Reapply the APARs or levelset and then rerun the iogmgnt -migrate command.

IOGU0014E The migration stopped because a backup copy of *file_name* **cannot be created with this name and path:** *full_path_to_file*. **Make sure you have the correct permissions and the system is not out of disk space.**

Explanation: The file required for the migration does not exist. The file or the directory it is in might be accidentally deleted.

User response: Ensure that you have the permission to write to the system, and the system is not out of disk space.

IOGU0015E *Number_of_files* **correlator file(s) were not migrated. Make sure you have the correct permission and the system is not out of disk space. Correct the error and try again.**

Explanation: The reported number of correlator files could not be copied over to this SOAP Gateway installation.

User response: See the migration report for the list of correlator files that were not migrated. Ensure that you have the permission to write to the system, and the system is not out of disk space.

IOGU0016E *Number_of_files* **wsdl/xsd file(s) were not migrated. Make sure you have the correct permission and the system is not out of disk space. Correct the error and try again.**

Explanation: The reported number of WSDL or XSD files could not be copied over to this SOAP Gateway installation.

User response: See the migration report for the list of WSDL or XSD files that were not migrated. Ensure that you have the permission to write to the system, and the system is not out of disk space.

IOGU0017E *Number_of_files* **aar file(s) were not migrated. Make sure you have the correct permission and the system is not out of disk space. Correct the error and try again.**

Explanation: The reported number of AAR files could not be copied over to this SOAP Gateway installation.

User response: See the migration report for the list of AAR files that were not migrated. Ensure that you have the permission to write to the system, and the system is not out of disk space.

IOGU0018I **Migration summary report. Migrate files from:** *path_to_previous_installation* **to:** *path_to_current_installation***.**

Explanation: This is an informational message to report the source and the target of the migration.

User response: No action is required.

IOGU0021I **The following** *number_of_files* **correlator file(s) were updated to the new schema:** *list_of_files***.**

Explanation: This is an informational message to report the correlator files that were updated to the new schema. This message is generated when the correlators were using an older version of schema in IMS Enterprise Suite Version 1.1 SOAP Gateway Fix Pack 1 or the base release.

User response: No action is required.

IOGU0022I **The following** *number_of_files* **correlator file(s) cannot be copied. Make sure that you have the correct permissions and the system is not out of disk space.**

Explanation: When the summary report contains this IOGU0022I message, an IOGU0015E message was sent to the console during the migration operation that indicated the source correlator files that could not be migrated.

User response: Check the list of files that cannot be copies and ensure that file permission is properly set and adequate disk space is available on the target system.

IOGU0023I **The following** *number_of_files* **WSDL and XSD file(s) cannot be copied. Make sure that you have the correct permissions and the system is not out of disk space.**

Explanation: When the summary report contains this IOGU0023I message, an IOGU0016E message was sent to the console during the migration operation that indicated the source WSDL or XSD files that could not be migrated.

User response: Check the list of files that cannot be copies and ensure that file permission is properly set and adequate disk space is available on the target system.

IOGU0024I **The following** *number_of_files* **AAR file(s) cannot be copied. Make sure that you have the correct permissions and the system is not out of disk space.**

Explanation: When the summary report contains this IOGU0024I message, an IOGU0017E message was sent to the console during the migration operation that indicated the source AAR files that could not be migrated.

User response: Check the list of files that cannot be copies and ensure that file permission is properly set and adequate disk space is available on the target system.

IOGU0025E **The migration process stopped because the directory path specified for SOAP Gateway to migrate from** *path_to_previous_installation* **does not exist or cannot be accessed.**

Explanation: The migration script stopped because the source directory is not accessible.

User response: Check that the source directory is specified correctly. The source directory must be an absolute path to where a previous release of SOAP Gateway is installed. Ensure that access permission is set correctly.

IOGU0027I **Migrating web services and connection bundle files from:** *path_to_previous_installation*

Explanation: This is an informational message sent to the console to indicate the status of the migration.

User response: No action is needed.

IOGU0028I The following *number_of_files* **correlator file(s) either did not contain a valid WSDL or XSD file name(s), or the corresponding WSDL or XSD file can not be found. The files were not updated:** *list_of_files***.**

Explanation: This is an informational message to report the correlator files that were not updated. Because the corresponding WSDL or XSD file is either not valid or not found for the reported correlator files, they are most likely incorrectly left in the source directory and are not used for any web service.

User response: If the reported correlator files are not used for any existing web service, no action is needed.

IOGU0030I The following *number_of_files* **correlator file(s) were copied to:** *path_to_new_location***.**

Explanation: This is an informational message to report the correlator files that were copied and the location they were copied to.

User response: No action is needed.

IOGU0031I The following *number_of_files* **WSDL and XSD file(s) were copied to:** *path_to_new_location***.**

Explanation: This is an informational message to report the WSDL and XSD files that were copied and the location they were copied to.

User response: No action is needed.

IOGU0032I The following *number_of_files* **AAR file(s) were copied to:** *path_to_new_location***.**

Explanation: This is an informational message to report the AAR files that were copied and the location they were copied to.

User response: No action is needed.

IOGU0033I The connection bundle file was copied to *path_to_new_location***.**

Explanation: This is an informational message to report that the connection bundle file has been copied, and the new location they were copied to.

User response: No action is needed.

IOGU0034I The connection bundle file was copied to *path_to_new_location***. Previous connection bundle file was renamed to** *new_name***.**

Explanation: This is an informational message to report that the connection bundle file has been copied, and the new location it was copied to. Because an existing connection bundle file of the same name exists, before the copy, the existing file was renamed, with the timestamp as the suffix. You can run the migration tool as many times as necessary to migrate all web services successfully without losing the previous copies.

User response: No action is needed.

IOGU0035I Your web services and connection bundle files are migrated. For any security files, you must manually copy them over. You must re-configure your server properties. You must restart the SOAP Gateway server for the migration changes to take effect. Refer to the SOAP Gateway online readme file for more information.

Explanation: This is an informational message to report that the migration is successful, and subsequent manual migration tasks are required for any security files and server properties.

User response: No action is needed.

IOGU0036W Migration function for *type_of_files* **is not supported for source release** *old_release* **to target release** *new_release***. Additional message:** *additional_message***.**

Explanation: This warning message is logged to inform users that the migration function for the identified type of files is not supported for migrating from the *old_release* to the *new_release*. The migration utility continues to run after issuing the warning.

User response: Full migration is only supported from V2.1 to V2.2. If the specified source release directory is for a previous version, server configuration information is not preserved. Point to a V2.1 installation directory for a full migration to V2.2, or manually configure the server after migration.

IOGU0037E The migration command failed to deploy the *web_service_type* **web service to** *target_location***: Web service definition and associated schema XML files:** *WSDL_or_XSD_file_name***. Correlator XML file:** *correlator_file_name***.**

Explanation: The *web_service_type* variable would indicate whether it is a provider, callout, or business event web service that failed to be migrated. The *target_location* is either the target service path for provider web services, or the master or runtime configuration for callout or business event web services.

User response: Check that the reported web service works properly in the previous version.

For callout or business event web services, also check whether the tpipe information exists in the connection bundle.

IOGU0038I No migration action was taken.

Explanation: This is an informational message to indicate that no migration action was taken. This message is issued when you rerun the migration tool, but no changes are found since the last time the tool is run.

User response: No action is needed.

IOGU0039I Service file (*service_AAR_file***) already exists in the target (***target_server_directory***). No migration action was taken.**

Explanation: This is an informational message to indicate that no migration action was taken because the service to be migrated already exists in the target location.

User response: No action is needed.

IOGU0040I The migration command successfully deployed the *web_service_type* **web service to** *target_location***: Web service definition and associated schema XML files:***WSDL_or_XSD_xml_file***. Correlator XML file:** *correlator_file***.**

Explanation: This is an informational message to indicate that the provider, callout, or business event (*web_service_type*) web service and its associated XML files are successfully migrated and the web service is deployed in the target location. The *target_location* is either the target service path for provider web services, or the master or runtime configuration for callout or business event web services.

User response: No action is needed.

IOGU4000I The *type_of_files* **variable is set to** *directory***.**

Explanation: This is an informational message sent to the console to log the installation path variable assignments.

User response: No action is needed.

IOGU4002I No changes were made for *directory***.**

Explanation: This message is generated during migration when the migration utility encounters a file that is eligible for migration but finds that no changes are needed.

User response: No action is needed.

IOGU7000E The *file_type_installation_path* **property cannot be configured. The path set for** *file_type_installation_path* **might not be a directory, or was not configured. Additional message:** *additional_message***.**

Explanation: The migration utility can not determine the source or the target location. Most likely the source directory specified is not a directory, does not exist, or was not a SOAP Gateway installation directory.

User response: Run the migration command again and specify a valid SOAP Gateway installation directory.

IOGU7002E **The XML cannot be processed because the (*file_name*) file is ill-formed at line *line_number*. Additional message: *additional_message*.**

Explanation: An XML parsing error occurred, such as the `server.xml` file.

User response: Contact IBM Software Support.

IOGU7003E **The** iogmgmt -migrate **command failed because a valid command parameter was not specified. The command requires a valid command parameter. See the SOAP Gateway management utility command reference information for a list of valid command parameters.**

Explanation: An invalid argument was specified with the management utility iogmgmt -migrate command.

User response: Run the iogmgmt -migrate command again and specify a valid argument.

IOGX messages

IOGX messages are returned from the XML parser.

IOGX002E **Error encountered with the connection bundle properties [***connbundle_file***]: [***error_message***]**

Explanation: The connection bundle file is not valid and it does not conform to its schema definition. Possible reason is the connection bundle file has been modified manually instead of using the SOAP Gateway management utility.

User response: Recreate the connection bundle file by using the SOAP Gateway management utility. You have to first remove the invalid connection bundle file located in the *`connbundle_file`* file. You might need to write down the property values in the invalid connection bundle file before you create the new one.

Alternatively, you can modify the connection bundle file to be compliant with the schema that is defined in the *`SOAP_Gateway_install_directory`*`\server\webapps\imssoap\xml\connbundle.xsd` file.

IOGX003E **Error encountered with correlator properties [***correlator_file***]:[***error message***]**

Explanation: The correlator file is invalid and it does not conform to its schema definition. Possible reason is the correlator file has been modified manually instead of using the SOAP Gateway management utility.

User response: Recreate the correlator file by using the SOAP Gateway management utility. You have to first remove the invalid correlator file located in [*correlator_file*]. You might need to write down the property values in invalid correlator file before you create the new one.

Alternatively, you can modify the correlator file to be compliant with the schema that is defined in the *`SOAP_Gateway_install_directory`*`\server\webapps\imssoap\xml\correlator.xsd` file.

IOGX027E **Parse error encountered with the file [***XML_file_being_parsed***]:[***error message***]**

Explanation: The XML file being parsed encountered a parse exception.

User response: Ensure that the XML file that is being parsed follows the XSD schema definition rules that it is based on.

IOGX0000E **SOAP Gateway has detected a Java stack trace useful for technical support.** *Error_details.*

Explanation: Possible error details are:

- The stack trace is written to the log.
- The stack trace cannot be written to the log because the logfile appender is set to OFF. Turn on the logfile appender and try to recreate the problem so the stack trace could be written to the log.
- The stack trace is: *stack_trace_details*

User response: If the stack trace is written to the log, examine the `imssoap.log` file in the *`install_dir`*`/imsbase/logs` directory.

If the stack trace is not written to the log, use the iogmgmt -prop command to set the trace level to 2 (ERROR) or above:

```
iogmgmt -prop -u -l 5
```

IOGX0001W SOAP Gateway cannot find any web service information in the cache. Valid correlator, WSDL, and connection bundles files for each service must exist.

Explanation: Possible reasons include:

- SOAP Gateway cannot find any correlators.
- SOAP Gateway cannot find any WSDL files.
- Web service artifacts are not migrated to the requirements for this version of SOAP Gateway.

If you encounter this message during server startup, the server startup process is unaffected by this warning, except that the related web service is not loaded into the runtime cache.

User response: If a web service is expected to be in service, use the SOAP Gateway management utility to create the connection bundle. Use Rational Developer for System z Version 8.0.3.2 or later to generate the correlator file and the WSDL file. Deploy the web service by using the SOAP Gateway management utility.

IOGX0002W The correlator entries cannot be processed because the (*correlator_filename*) file is ill-formed at line *line_number*. *error_details*.

Explanation: The XML at the specified line is ill-formed. The correlator file cannot be processed, and the correlator entries are not loaded. If you encounter this message during server startup, the server startup process is unaffected by this warning, except that the related web service is not loaded into the runtime cache.

User response: If this web service is expected to be in service, correct the ill-formed XML at the specified line. Use Rational Developer for System z Version 8.0.3.2 or later to generate the correlator file and the WSDL file. Redeploy the web service by using the SOAP Gateway management utility -deploy command.

IOGX0003W The correlator entries cannot be processed because the (*web_service_artifact*) file cannot be read or found. *Exception_message*.

Explanation: Possible reasons include:

- The file could not be read.
- The file could not be found.
- The file encoding is not UTF-8.

If you encounter this message during server startup, the server startup process is unaffected by this warning, except that the related web service is not loaded into the runtime cache.

User response: If the web service is expected to be in service, check that the identified file actually exists, is properly encoded, and is not corrupted. Correct the problem, and redeploy the web service by using the SOAP Gateway management utility -deploy command.

IOGX0004W The correlator entries cannot be processed because the (*web_service_artifact*) file encountered an exception: *Exception_message*.

Explanation: If you encounter this message during server startup, the server startup process is unaffected by this warning, except that the related web service is not loaded into the runtime cache.

User response: If the web service is expected to be in service, examine the exception and correct the problem. Redeploy the web service by using the SOAP Gateway management utility -deploy command.

IOGX0005W The correlator entries cannot be processed because the correlator version *version_number* of the (*correlator_file*) file is not supported with this release of SOAP Gateway.

Explanation: The correlator entries are not compatible with this release of SOAP Gateway.

If you encounter this message during server startup, the server startup process is unaffected by this warning, except that the related web service is not loaded into the runtime cache.

User response: If the web service is expected to be in service, create a valid correlator file by using Rational Developer for System z Version 8.0.3.2 or later.

IOGX0006W The correlator entries cannot be processed because the value of (*attribute_value*) for the (*attribute_name*) attribute in the (*correlator_file*) file is not valid.

Explanation: The value might be null or empty.

If you encounter this message during server startup, the server startup process is unaffected by this warning, except that the related web service is not loaded into the runtime cache.

User response: If the web service is expected to be in service, correct the value or regenerate the correlator file by using Rational Developer for System z Version 8.0.3.2 or later. Redeploy the web service by using the SOAP Gateway management utility -deploy command.

IOGX0007W The correlator entries cannot be processed. The value of (*element_value*) for the (*element_name*) element in the (*correlator_file*) file is either null or not valid. serviceName = (*service_name*), operationName = (*operation_name*), portName = (*port_number*), targetNamespace = (*target_namespace*).

Explanation: The value of the reported element is either null or empty. The provided web service information helps you identify the correlator entry where the invalid value is found.

If you encounter this message during server startup, the server startup process is unaffected by this warning, except that the related web service is not loaded into the runtime cache.

User response: If the web service is expected to be in service, correct the element value or regenerate the correlator file by using Rational Developer for System z Version 8.0.3.2 or later. Redeploy the web service by using the SOAP Gateway management utility -deploy command.

IOGX0008W The correlator entries cannot be processed. The value of (*element_or_attribute_value*) for (*element_or_attribute*) in the *web_service_artifact_filename* file is null, empty, or not found.

Explanation: The value for the reported element or attribute in the WSDL or connection bundle file is not valid, and therefore the corresponding correlator file cannot be processed or loaded in to the cache. If you encounter this message during server startup, the server startup process is unaffected by this warning, except that the related web service is not loaded into the runtime cache.

User response: If the web service is expected to be in service, correct the value that caused the problem. Use the SOAP Gateway management utility to modify the connection bundle. Redeploy the web service by using the SOAP Gateway management utility -deploy command.

IOGX0009W The web service cannot be updated for operation (*operation_name*) because (*wsdl_filename*) is null, or the cache is empty. The correlator entries cannot be processed.

Explanation: The WSDL file for the specified operation does not exist, or the runtime cache is empty, so the web service the that operation name refers to cannot be updated. If you encounter this message during server startup, the server startup process is unaffected by this warning, except that the related web service is not loaded into the runtime cache.

User response: If the web service is expected to be in service, ensure that a valid WSDL file exists. Redeploy the web service by using the SOAP Gateway management utility -deploy command.

IOGX0010W *Exception_message*. The (*web_service_artifact_filename*) file is empty, invalid, in wrong encoding, or not found. The correlator entries cannot be processed. *Exception_details*.

Explanation: Possible exception messages are:

- Illegal Argument Exception
- Unsupported Encoding Exception
- WSDL Exception

If you encounter this message during server startup, the server startup process is unaffected by this warning, except that the related web service is not loaded into the runtime cache.

User response: If the web service is expected to be in service, ensure that the named web service artifact file exists, is valid, and is in UTF-8 encoding.

IOGX0011W The correlator (*correlator_entry*) in the (*web_service_artifact_file*) file does not provide sufficient information to qualify the web service. serviceName = (*service_name*), operationName = (*operation_name*), portName = (*port_name*), targetNamespace = (*target_namespace*).

Explanation: The correlator entry, defined by the operation name, port name, and service name, does not provide enough information to qualify the web service.

If you encounter this message during server startup, the server startup process is unaffected by this warning, except that the related web service is not loaded into the runtime cache.

User response: If the web service is expected to be in service, regenerate the correlator file by using Rational Developer for System z Version 8.0.3.2 or later. Redeploy the web service by using the SOAP Gateway management utility -deploy command.

IOGX0012W The correlator (*correlator_entry*) cannot be stored in the cache because the operation is already associated with the (*web_service_artifact*) file. serviceName = (*service_name*), operationName = (*operation_name*), portName = (*port_name*), targetNamespace = (*target_namespace*).

Explanation: The correlator entry, defined by the operation name, port name, and service name, because the operation is already associated with a different web service.

If you encounter this message during server startup, the server startup process is unaffected by this warning, except that the related web service is not loaded into the runtime cache.

User response: If the web service is expected to be in service, check that the operation name is correct. Regenerate the correlator file by using Rational Developer for System z Version 8.0.3.2 or later. Redeploy the web service by using the SOAP Gateway management utility -deploy command.

IOGX0013W The service cannot be determined as either a provider, consumer, or business event service for correlator (*correlator_entry*) based on the following information: Service = (*service_name*), Operation = (*operation_name*), PortName = (*port_name*), TargetNamespace = (*target_namespace*). The connection bundle information must be defined in the correlator and present in the connection bundle, or the wsdlFile name or xsdFile name must be properly set.

Explanation: The correlator entry, defined by the operation name, port name, and service name, does not provide enough information to determine the scenario. Either the related connection bundle information does not exist, or the WSDL file or XSD file name is not set.

If you encounter this message during server startup, the server startup process is unaffected by this warning, except that the related web service is not loaded into the runtime cache.

User response: If the web service is expected to be in service, use the SOAP Gateway management utility to correct the connection bundle information. Redeploy the web service and specify the correct connection bundle, WSDL, or XSD information.

IOGX0014W The (*element_name*) element in the (*web_service_artifact*) file contains the following attribute values: *attr1* = (*attr1_value*), {*attr2*} = (*attr2_value*), {*attr3*} = (*attr3_value*). These values are not supported by the business event service(s). The correlator entries are not loaded into the cache.

Explanation: The correlator entry or entries contain attributes that are not supported by the business event scenario.

If you encounter this message during server startup, the server startup process is unaffected by this warning, except that the related web service is not loaded into the runtime cache.

User response: If the web service is expected to be in service, check the correlator entries based on the reported attributes and attribute values. Create a valid correlator file by using Rational Developer for System z Version 8.0.3.2 or later. Use the SOAP Gateway management utility to . Redeploy the web service by using the SOAP Gateway management utility -deploy command.

IOGX0015W Entries in the {*correlator_file*} correlator cannot be loaded into the cache. The (*service_name*) web service specified in the (*web_service_artifact*) file is a duplicate: operationName = (*operation_name*), portName = (*port_name*), targetNamespace = (*target_namespace*).

Explanation: The named web service already exists.

If you encounter this message during server startup, the server startup process is unaffected by this warning, except that the related web service is not loaded into the runtime cache.

User response: If the web service is expected to be in service, check the correlator entry defined by the operation name and port name.

IOGX0016W The value of (*element_value*) for the (*element_name*) element in the correlator (*correlator_entry*) cannot be populated because (*entry_name*) is not a valid entry. The service entry is not loaded into the cache.

Explanation: The value of the named element can be populated and loaded into the runtime cache only when the named entry is valid for the web service..

If you encounter this message during server startup, the server startup process is unaffected by this warning, except that the related web service is not loaded into the runtime cache.

User response: If the web service is expected to be in service, correct the invalid entry and redeploy the web service by using the SOAP Gateway management utility -deploy command.

IOGX0017W The schema (*schema_name*) cannot be loaded from the memory. SOAP Gateway must default to validating the schema (*master_configuration_schema_name*) from the file system.

Explanation: The schema defined in the correlator or connection bundle cannot be loaded from the cache. SOAP Gateway must validate the schema from the master configuration in the file system.

If you encounter this message during server startup, the server startup process is unaffected by this warning, except that the related web service is not loaded into the runtime cache.

User response: If the web service is expected to be in service, check that the named schema in the correlator or connection bundle is not manually modified.

IOGX0018W The correlator entries cannot be processed. One or more attributes for the (*element_name*) element in the (*web_service_artifact*) file is either null or empty: Attributes = (*list_of_attributes*).

Explanation: If you encounter this message during server startup, the server startup process is unaffected by this warning, except that the related web service is not loaded into the runtime cache.

User response: If the web service is expected to be in service, check that all the attributes in the identified element have valid value. Correct the problem and then redeploy the web service by using the SOAP Gateway management utility.

IOGX7004E SOAP Gateway cannot configure the XML parser because a {*exception_reason*} exception occurred. *Exception_details*.

Explanation: Possible reasons for the exception include:

- Can not configure a parser
- Can not provide a parser
- Unable to set parser property
- Unrecognized parser property

User response: Contact IBM Software Support and provide the exception details and stack trace information.

IOGX7005E SOAP Gateway cannot configure the XML parser because a {*exception_reason*} exception occurred. The SOAP Gateway server must exit. *Exception_details*.

Explanation: The SOAP Gateway server shuts down because a severe error occurs. Possible reasons for the exception include:

- Can not configure a parser

- Can not provide a parser
- Unable to set parser property
- Unrecognized parser property

User response: Contact IBM Software Support and provide the exception details and stack trace information.

IOGX7006E SOAP Gateway cannot load the schema (schema_name) because a *exception_reason* **exception occurred. The schema is either invalid, in an incorrect encoding, empty, or not found.** *Exception_details***.**

Explanation: Possible reasons for the exception are:

- UnsupportedEncodingException
- FileNotFoundException

User response: Ensure that the encoding to UTF-8. Check that the `correlator.xsd` correlator schema file exists in *`install_dir`*`/imssoap/xml/` directory. If the file is accidentally deleted, contact IBM software support for the correlator schema file, or reinstall SOAP Gateway.

IOGX7007E SOAP Gateway cannot load the schema at because a *exception_reason* **exception occurred. The schema is either invalid, in an incorrect encoding, empty, or not found. The** *exception_reason* **server must exit.** *Exception_details***.**

Explanation: Possible reasons for the exception are:

- UnsupportedEncodingException
- FileNotFoundException

User response: Ensure that the file encoding is UTF-8. Check that the `correlator.xsd` correlator schema file exists in *`install_dir`*`/imssoap/xml/` directory. If the file is accidentally deleted, contact IBM software support for the correlator schema file, or reinstall SOAP Gateway.

IOGX7008E The filename entry value (*wsdlFile_name***) in the (***correlator_file***) correlator file does not match the (***wsdl_file***) file that was specified.**

Explanation: The WSDL file name entry in the correlator file does not match the WSDL file that was specified during deployment. Verify that the WSDL file name in the correlator is the same as the WSDL file specified.

User response: Correct the correlator, or specify the correct WSDL file name when using the SOAP Gateway management utility.

Chapter 11. SOAP Gateway management utility reference

The SOAP Gateway management utility provides a command line interface to manage the SOAP Gateway server runtime, configure server properties, and work with web service artifacts.

The following table shows the support and restrictions of the SOAP Gateway management utility for each scenario.

Table 39. Features supported by the management utility

Scenario	Administrative tasks supported by the SOAP Gateway management utility(iogmgmt)
Web service provider	• Full support for administrative tasks. • Changes to web service artifacts are immediately reflected in the runtime configuration.
Callout application (web service consumer)	• Full support for administrative tasks. • Changes to callout application artifacts are immediately reflected in the runtime configuration. • Changes to callout properties are made to the runtime configuration, but the callout threads and callout worker thread pool must be restarted.
Business event	• Full support for administrative tasks. • Changes to business event artifacts are immediately reflected in the runtime configuration. • Changes to callout properties for business event processing are made to the runtime configuration, but the callout threads and callout worker thread pool must be restarted. • Creation of correlator XML files for WebSphere Business Monitor is supported only with Rational Developer for System z.

Restriction: Changes to active connection bundle entries are not reflected until the next time the SOAP Gateway server starts. All valid connection bundle entries are made active when the SOAP Gateway server starts.

iogmgmt commands

To use the SOAP Gateway management utility, from the installation directory where the SOAP Gateway is installed (*SOAP_Gateway_install_directory*\imsserver\deploy), type `iogmgmt` (for Windows) or `./iogmgmt` (for z/OS and Linux on System z).

On Windows 7 systems, depending on the SOAP Gateway installation directory, you might need to open a command prompt as an administrator and change directories to the SOAP Gateway management utility directory (*install_dir*/imsserver/deploy) before you can issue any command.

An iogmgmt command consists of the `iogmgmt` statement followed by arguments to specify the command and associated options.

Issuing a command with more than one instance of a parameter will override all but the last instance of the parameter. For example, issuing the command `iogmgmt -corr -u -r MyCorr.xml -p MyService -i MyOperation -n NewBundleName1 -n NewBundleName2` will set the connection bundle name for the specified correlator entry to `NewBundleName2`.

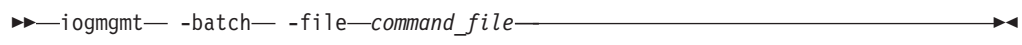
SOAP Gateway management utility commands are case sensitive and must be entered in all lower case or as shown in the command reference topic.

For SOAP Gateway servers running on z/OS, the SOAP Gateway management utility must run on the same LPAR as the target server.

-batch: Run management utility commands in batch mode

The -batch command runs multiple SOAP Gateway management utility commands as a batch in one JVM instance.

Syntax



Usage

Use this command and specify the file that contains one or more SOAP Gateway management utility commands to run. All SOAP Gateway management utility commands, except the iogmgmt –batch command, can be specified in the file. Separate the commands with semicolons.

A `batch.`*`timestamp`*`.log` file is created each time that the command is run. The log contains the result of the run, including everything that is sent to the console. The log file also shows the elapsed time.

If any errors are encountered, a `batchFail.`*`timestamp`*`.txt` file is generated that contains a list of failed commands. Use this failure log to identify and correct the commands that are problematic. This file is provided to facilitate problem correction and rerun of corrected commands.

Parameters

`-batch`
Specifies to run SOAP Gateway management utility commands in batch mode. The shortcut for the **`-batch`** keyword is **`-b`**.

`-file`
Specifies either the absolute or relative path to the file that contains semicolon-separated SOAP Gateway management utility commands to run in the batch mode. The shortcut for the **`-file`** keyword is **`-f`**.

Example

```
iogmgmt -batch -file  c:\path_to\mySOAPcommands.txt
```

The `mySOAPcommands.txt` might consist of a list of SOAP Gateway management utility commands, as shown in the following example:

```
iogmgmt -deploy -w MyWSDL.wsdl -r MyCorrelator.xml;
iogmgmt -deploy -w HelloService.wsdl -r HelloService.xml;
iogmgmt -deploy -w MyWSDLWSS.wsdl -r MyCorrelatorWSS.xml -t SAML20Token;
iogmgmt -deploy -w myCalloutWSDL.wsdl -r myCalloutCorrelator.xml;
iogmgmt -deploy -w myCalloutWSDLWSS.wsdl -r myCalloutCorrelatorWSS.xml -t SAML20Token
```

If an error is encountered with one of the commands, the batch command continues to run the subsequent commands.

-callout -startall: Start all callout threads

The -callout -startall command starts all callout threads.

Syntax

Parameters

-callout -startall
Specify to start all callout threads.

Example

```
iogmgmt -callout -startall
```

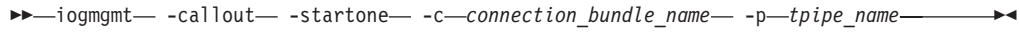
Related concepts:

"Thread management for callout messages retrieval" on page 174
SOAP Gateway supports two options to determine how to manage the callout threads to send the requests to poll the hold queue for callout request messages: one thread per tpipe, or one thread per connection bundle.

-callout -startone: Start a specific callout thread

The -callout -startone command starts a specific callout thread based on the provided connection bundle name and tpipe name.

Syntax



Parameters

-callout -startone
Specifies to start a specific callout thread.

-c *connection_bundle_name*
Specifies the callout connection bundle name. This parameter is required.

-p *tpipe_name*
Specifies the name of the tpipe for inbound messages from IMS. This parameter is required only when the thread policy is set to one thread per tpipe.

Example

```
iogmgmt -callout -startone -c IMSPHBK -p SGPSING1
```

Related concepts:

"Thread management for callout messages retrieval" on page 174
SOAP Gateway supports two options to determine how to manage the callout threads to send the requests to poll the hold queue for callout request messages: one thread per tpipe, or one thread per connection bundle.

-callout -startpool: Start the thread pool

The -callout -startpool command starts the thread pool.

Syntax

Parameters

-callout -startpool
Specifies to start the callout thread pool.

Example

```
iogmgmt -callout -startpool
```

Related concepts:

"Thread management for callout messages retrieval" on page 174
SOAP Gateway supports two options to determine how to manage the callout threads to send the requests to poll the hold queue for callout request messages: one thread per tpipe, or one thread per connection bundle.

-callout -stopall: Stop all callout threads

The -callout -stopall command stops all callout threads.

Syntax

Parameters

-callout -stopall
Specify to stop all callout threads.

Example

```
iogmgmt -callout -stopall
```

Related concepts:

"Thread management for callout messages retrieval" on page 174
SOAP Gateway supports two options to determine how to manage the callout threads to send the requests to poll the hold queue for callout request messages: one thread per tpipe, or one thread per connection bundle.

-callout -stopone: Stop a specific callout thread

The -callout –stopone command stops a specific callout thread based on the provided connection bundle name and tpipe name.

Syntax

Parameters

-callout -stopone
Specifies to stop a specific callout thread.

-c *connection_bundle_name*
Specifies the callout connection bundle name. This parameter is required.

-p *tpipe_name*
Specifies the name of the tpipe for inbound messages from IMS. This parameter is required only when the thread policy is set to one thread per tpipe.

Example

```
iogmgmt -callout -stopone -c IMSPHBK -p SGPSING1
```

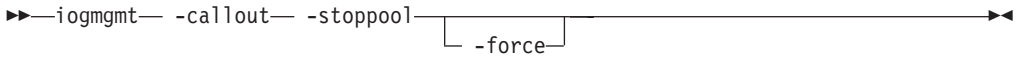
Related concepts:

"Thread management for callout messages retrieval" on page 174
SOAP Gateway supports two options to determine how to manage the callout threads to send the requests to poll the hold queue for callout request messages: one thread per tpipe, or one thread per connection bundle.

-callout -stoppool: Stop the thread pool

The -callout –stoppool command stops the thread pool.

Syntax



Parameters

-callout -stoppool
Specifies to stop the thread pool after all in-flight messages are processed.

-force
Specifies to stop the thread pool immediately and discard all in-flight messages.

Example

To stop the thread pool gracefully:

```
iogmgmt -callout -stoppool
```

To stop the thread pool immediately:

```
iogmgmt -callout -stoppool -force
```

Related concepts:

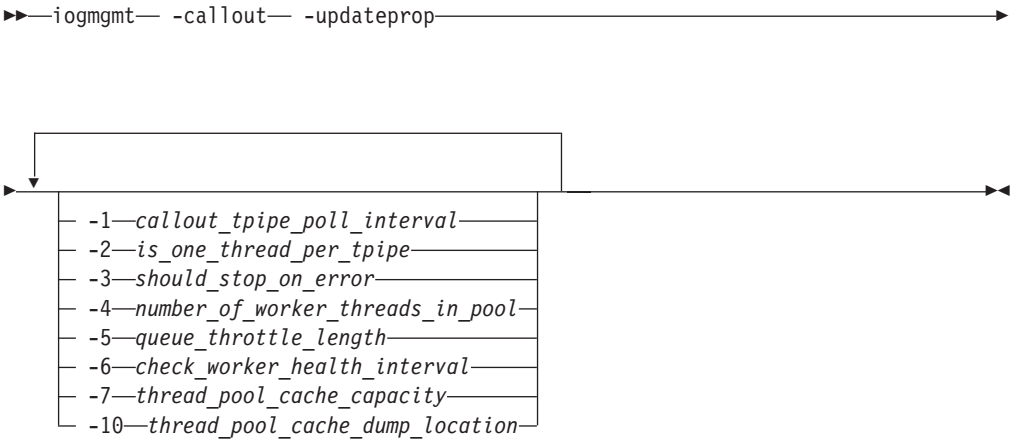
"Thread management for callout messages retrieval" on page 174
SOAP Gateway supports two options to determine how to manage the callout threads to send the requests to poll the hold queue for callout request messages: one thread per tpipe, or one thread per connection bundle.

-callout -updateprop: Update SOAP Gateway callout properties

The -callout –updateprop command updates the SOAP Gateway callout properties.

Syntax

Important: Before modifying the SOAP Gateway callout properties, see "Thread management and configuration considerations" on page 180 for information about how the SOAP Gateway server uses these properties.



Parameters

-callout -updateprop
Specifies to update the callout properties that are specified.

-1 *callout_tpipe_poll_interval*
Specifies the time interval in milliseconds that each SOAP Gateway callout thread idles between attempts to poll the assigned tpipe for messages.

-2 *is_one_thread_per_tpipe*
Specifies whether to create one thread per tpipe. Valid values are true and false. The default value is true. If you set this property to false, one thread is created for each connection bundle instead.

-3 *should_stop_on_error*
Specifies whether a callout thread stops when it encounters an error. Valid values are true and false.

-4 *number_of_worker_threads_in_pool*
Specifies the number of worker threads in the thread pool. Valid values are 1 - 32.

-5 *queue_throttle_length*
Specifies the maximum number of in-flight messages (jobs) that can be stored in the work queue by the callout threads. Valid values are 1 - 64.

-6 *check_worker_health_interval*
Specifies the time in milliseconds between each check of the thread pool by the daemon thread. The daemon thread sets the number of active worker threads to the value specified with the **-4** parameter of this command. Valid values are 1 - 86400000 ms.

-7 *thread_pool_cache_capacity*
Specifies the maximum number of messages that are stored in the cache for the thread pool. Valid values are 1 - 2000.

-10*thread_pool_cache_dump_location*
Specifies the fully qualified path to a directory where the thread pool cache memory dump is saved when you issue the iogmgmt -view -workerthreads command.

Examples

The following example updates a callout property to set the queue throttle length to a maximum of 50 in-flight messages that are allowed in the queue.

```
iogmgmt -callout -updateprop -5 50
```

The following example updates the callout properties to set the callout tpipe poll interval to 4 seconds, and to stop the callout thread when an error occurs:

```
iogmgmt -callout -updateprop -1 4000 -3 true
```

Related concepts:

"Thread management for callout messages retrieval" on page 174
SOAP Gateway supports two options to determine how to manage the callout threads to send the requests to poll the hold queue for callout request messages: one thread per tpipe, or one thread per connection bundle.

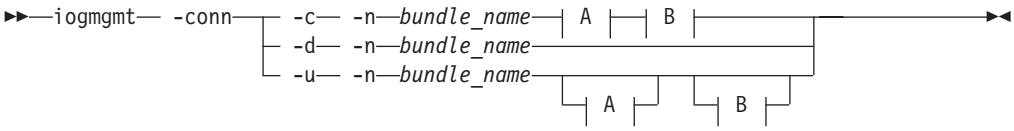
Related tasks:

"Updating SOAP Gateway callout properties" on page 336
You can update SOAP Gateway callout properties by using the SOAP Gateway management utility.

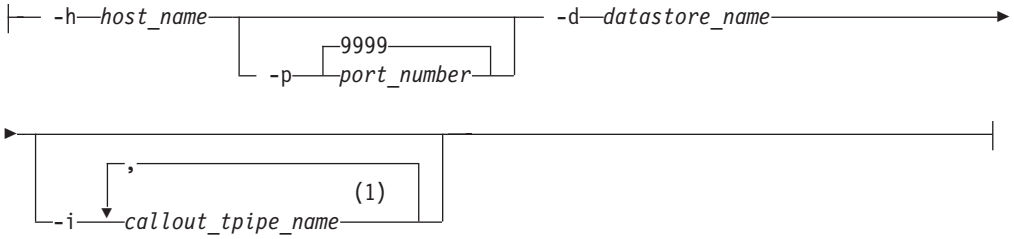
-conn: Create, update, or delete a connection bundle

Use the -conn command to create, update, or delete a connection bundle.

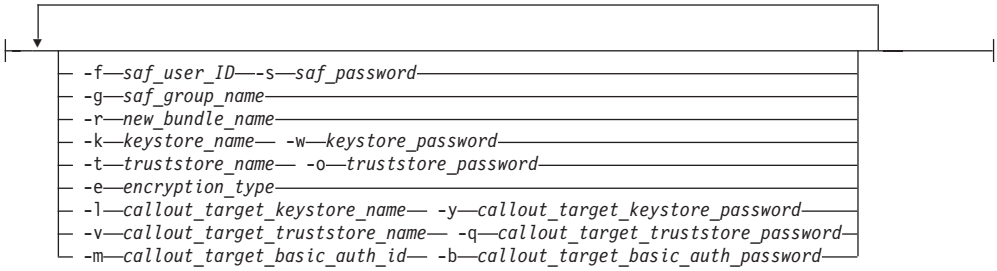
Syntax



A:



B:



Notes:

1 At least one tpipe name is required for a callout application connection bundle. Do not specify any tpipe names for a provider application connection bundle.

Parameters

Use the following parameters to create, update, or delete a connection bundle for a web service or callout application.

This command changes the server master configuration. Changes take effect in the runtime configuration the next time the server starts.

-conn
Specifies the connection bundle task.

-c Specifies that you want to create a connection bundle.

-u Specifies that you want to update an existing connection bundle.

-d Specifies that you want to delete an existing connection bundle. All parameters other than the connection bundle name are ignored when you specify option `-d`.

-n *bundle_name*
The name of the connection bundle to create or update. The correlator file uses this name to specify which connection properties to use for its associated web service. The name must be 20 characters or less and contain no spaces.

-h *host_name*
The name or IP address of the host system where IMS Connect is running.

-p *port_number*
The port number that SOAP Gateway uses when sending messages to IMS Connect. Valid port numbers are from 0 to 65535. The default is 9999.

-d *datastore_name*
The name of the target IMS Connect data store definition. The name must match the ID parameter of the data store statement that is specified in the IMS Connect configuration member of the IMS.PROCLIB data set, HWSCFGxx. This name also serves as the XCF member name for IMS during internal XCF communications between IMS Connect and IMS OTMA. The name must be in all uppercase characters, and must be eight characters or less.

-f *saf_user_ID*
The default security authorization facility (SAF) user name or Resource Access Control Facility (RACF) ID that is used for connections with IMS. The user ID must be eight characters or less.

-s *saf_password*
The SAF or RACF password that is used for connections with IMS. The password must be eight characters or less.

-g *saf_group_name*
The name of the SAF group that is used for connections with IMS. The name must be eight characters or less.

-r *new_bundle_name*
Specifies a new name for the connection bundle. The name must be 20 characters or less and contain no spaces.

-k *keystore_name*
The fully qualified path name of the keystore in which client certificates or private keys are stored. A value for this parameter is required if the IMS Connect is configured to use client authentication. Do not specify this parameter if you are not using SSL authentication. Specifying this parameter for a connection bundle that is not using a fully valid HTTPS connection will result in an IOGC0004E error message when a web service that uses this connection bundle is invoked.

-w *keystore_password*
The password of the keystore in which client certificates or private keys are stored. A value for this parameter is mandatory if a keystore is specified. Do not specify this parameter if you are not using SSL authentication. Specifying this parameter for a connection bundle that is not using a fully valid HTTPS connection will result in an IOGC0004E error message when a web service that uses this connection bundle is invoked.

-t *truststore_name*
The fully qualified path name of the truststore in which trusted certificates are stored. A value for this parameter is required if IMS Connect is configured to use client or server authentication. Do not specify this parameter if you are not using SSL authentication. Specifying this parameter for a connection bundle that is not using a fully valid HTTPS connection will result in an IOGC0004E error message when a web service that uses this connection bundle is invoked.

-o *truststore_password*
The password of the truststore in which trusted certificates are stored. A value for this parameter is required if a truststore name is specified. Do not specify this parameter if you are not using SSL authentication. Specifying this parameter for a connection bundle that is not using a fully valid HTTPS connection will result in an IOGC0004E error message when a web service that uses this connection bundle is invoked.

-e *encryption_type*

The encryption type used for SSL communication with IMS Connect. A value of **STRONG** indicates that a strong cipher suite must be used. A value of **WEAK** indicates that a weak cipher suite must be used, typically selected for export. A value of **NONE** indicates that authentication must be performed with no encryption of the messages exchanged. Strong and weak are related to the key length of the ciphers. All ciphers that can be used for export are classified as weak and all others as strong. The type of cipher suite varies depending on the JSSE provider and the version of the SSL package provided. For FIPS 140-2 and NIST SP800-131a, set the encryption type to **STRONG**.

In the process of establishing the SSL session, during the handshake sequence, SOAP Gateway selects the first cipher suite from the appropriate (strong or weak) list of cipher suites supported by the JSSE provider. SOAP Gateway

provides this cipher suite to the server (IMS Connect) so the server can determine if it supports the cipher suite specified by the client. Subsequent cipher suites might be selected from the list as the client and server negotiate a cipher suite to be used for the session.

Do not specify this parameter if you are not using SSL authentication. Specifying this parameter for a connection bundle that is not using a fully valid HTTPS connection will result in an IOGC0004E error message.

-i *callout_tpipe_name*
The callout tpipe name. Separate multiple tpipe names with commas.

-l *callout_target_keystore_name*
The name and location of the keystore used by a callout application to authenticate with a target web service. This property is only used when calling out to a web service that uses client authentication.

-y *callout_target_keystore_password*
The keystore password used by a callout application to authenticate with a target web service. This property is required when a callout target keystore name is specified. The password length must be 6 - 20 alphanumeric characters.

-v *callout_target_truststore_name*
The name and location of the truststore used by a callout application to authenticate with a target web service. This property is only used when calling out to a web service that uses client or server authentication.

-q *callout_target_truststore_password*
The truststore password used by a callout application to authenticate with a target web service. This property is required when a callout target truststore name is specified. The password length must be 6 - 20 alphanumeric characters.

-m *callout_target_basic_auth_id*
The callout basic authentication user ID, up to 255 characters. This property is only used when calling out to a web service that uses basic authentication.

-b *callout_target_basic_auth_password*
The callout basic authentication password, up to 255 characters. This property is required when a callout basic authentication user ID is specified.

Create examples

This example command creates a web service provider connection bundle that is named `MyBundle`. It specifies to connect on port 9990 to the data store `MYSTOR` on the IMS host named `MYHOST`.

```
iogmgmt -conn -c -n MyBundle -h MYHOST -p 9990 -d MYSTOR
```

The following command creates a callout connection bundle with basic authentication that is named MyCalloutConnBundle. It specifies to connect on port 9998 to the data store IMSSTOR1 on the IMS host named ICONHOST. The tpipe to pull the callout messages is tpipe1.

```
iogmgmt -conn -c -n MyCalloutConnBundle -h ICONHOST -p 9998
-d IMSSTOR1 -i tpipe1
-m MyCalloutBasicAuthID -b MyCalloutBasicAuthPwd
```

The following example creates a connection bundle with client authentication by specifying the keystore and truststore names and passwords the callout client uses to be authenticated with the target web service:

```
iogmgmt -conn -c -n MyCalloutConnBundle -h ICONHOST -p 9998
-d IMSSTOR1 -i tpipe1
-l /path/to/MyCalloutClientKeystore.ks -y MyCalloutclientKeystorePwd
-v /path/to/MyCalloutClientTruststore.ks -q MyCalloutClientTruststorePwd
```

Update example

This example changes the name of the connection bundle from `MyBundle` to `YourBundle`. It also specifies a default SAF user name of USER1 and SAF password `defaultpass`.

```
iogmgmt -conn -u -n MyBundle -r YourBundle -f USER1 -s defaultpass
```

Delete example

This example deletes the connection bundle that is named `YourBundle`.

```
iogmgmt -conn -d -n YourBundle
```

Related concepts:

"Connection bundle properties" on page 20
The connection bundle specifies the connection and security properties for SOAP Gateway when it communicates with IMS Connect.

"Connection bundle management" on page 296
Manage connection bundle entry names and usage by web services to ensure stable and predictable SOAP Gateway server behavior.

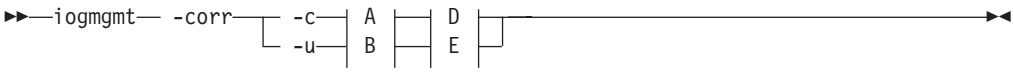
Related tasks:

"Creating a connection bundle entry for callout applications" on page 266
Create a connection bundle entry that describes the connection properties for accessing IMS by using the SOAP Gateway management utility. The connection bundle entries are stored in the `connbundle.xml` file.

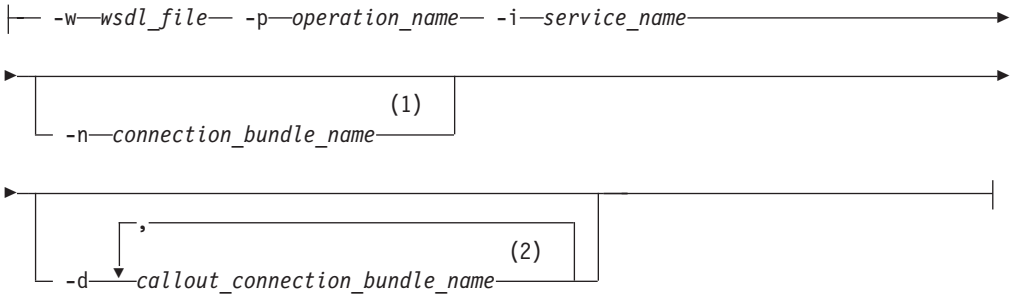
-corr: Create or update a correlator entry

Use the -corr command to create or update the transaction and runtime properties of a correlator entry.

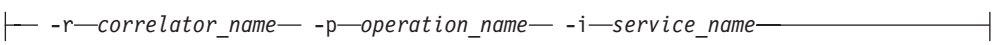
Syntax



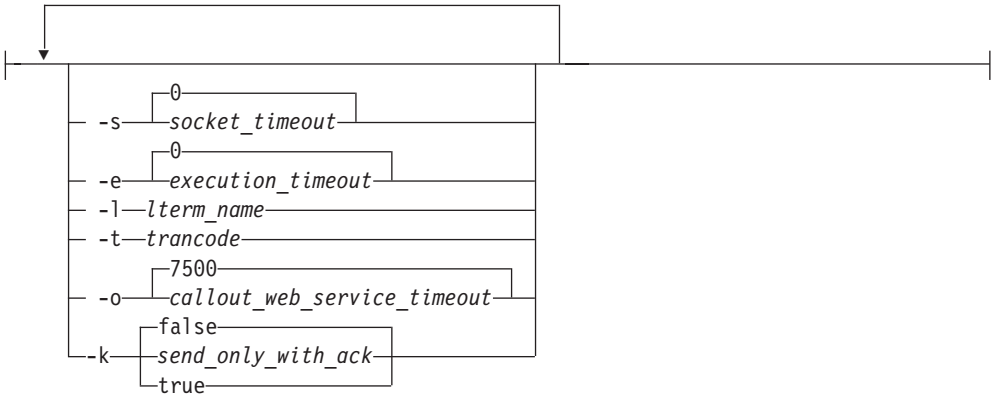
A:



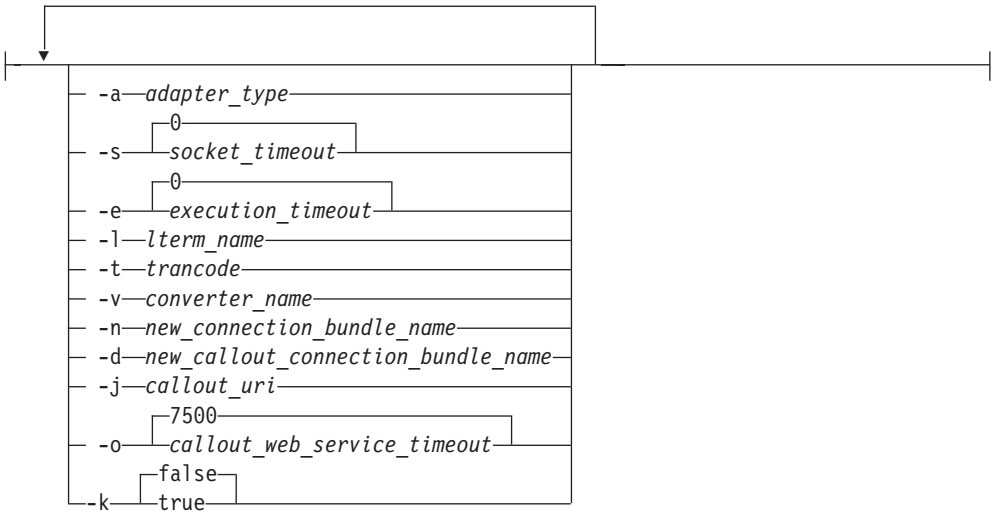
B:



D:



E:



Notes:

1 A connection bundle name is required for a web service provider correlator and is optional for a request-response callout application correlator.

2 A callout connection bundle name is required for a callout application correlator. A callout connection bundle is a connection bundle that contains callout tpipe names. Use commas to separate multiple connection bundle names for a synchronous callout application.

Parameters

Use the following parameters to create a SOAP Gateway correlator file that contains properties that define how the external web service or callout application operate. To create or update a correlator file for the callout scenario, specify a callout connection bundle name.

The service and operation names in the correlator entry form the unique identifier for a web service or callout application.

-corr
Specifies the correlator task.

-c Create a correlator file entry. The correlator file name is the URN of the web service. The URN is specified by the soapAction attribute of the soap:operation element in the WSDL file. This parameter is required when you are creating a correlator file.

Restriction:

- This utility cannot create correlator files for services that use an XML adapter in the target IMS Connect. To create a correlator that works with an IMS Connect XML adapter, use IBM Rational Developer for System z.
- This utility cannot create correlator files for the WebSphere Business Events or WebSphere Business Monitor scenarios. Use IBM Rational Developer for System z.

-u Update a correlator file entry.

-a The XML adapter type. Valid values are `IBM_XML_Adapter` (default) and `No_Adapter`. If you do not use the XML adapter function, specify `No_Adapter` and do not specify the converter name (the **-v** parameter). As a result, the adapterType entry in the correlator file is set to blank. This parameter is only valid when updating an existing correlator with option **-u**.

-d *callout_connection_bundle_name*
When creating a correlator, this parameter specifies the name of the callout connection bundle name. The name must be less than or equal to 20 alphanumeric characters with no spaces and no null (""). This parameter is required when creating a callout correlator. A single connection bundle can contain both callout and non-callout connection properties.

When updating an existing correlator, this parameter specifies the name of a new callout connection bundle to use with the correlator.

A callout correlator for a synchronous callout application can specify multiple connection bundle names. Separate multiple names with commas.

-e *execution_timeout*

Specifies the timeout value for IMS Connect to send a message to IMS and receive the response. The value must be between -1 and 3,600,000 (one hour). When the value is set to -1, the execution timeout is infinite. When the value is set to 0, IMS Connect uses its internal timeout value.

If you do not specify an execution timeout value or if the value that you specify is invalid, the timeout value in the IMS Connect configuration member is used and the interaction continues to run. If a valid execution timeout value is set, this value is converted into a value that IMS Connect can use. The following table describes how the values you specify are converted to the values that IMS Connect uses:

Table 40. Execution timeout value conversion rules

Range of execution timeout values in the correlator file	Conversion rule
1 – 250	If the value is not divisible by 10, it is converted to the next greater increment of 10.
251 – 1 000	If the value is not divisible by 50, it is converted to the next greater increment of 50.
1 001 – 60 000	If the value is not divisible by 1 000, it is converted to the next greater increment of 1 000. Values that are exactly between increments of 1 000 are converted to the next greater increment of 1 000.
60 001 – 3 600 000	The value is converted to the nearest increment of 60 000. Values that are exactly between increments of 60 000 are converted to the next greater increment of 60 000.

The default is zero. Optional.

-i *service_name*
Specifies the web service name your IMS application calls out to. The name must be a valid value from the WSDL file. The operation name and service name together form the unique identifier for the correlator. This parameter is required for all scenarios except when referencing an XSD schema file for WebSphere Business Monitor.

-k **false**|**true**
Specifies whether a synchronous callout application uses the send-only protocol with acknowledgement for response messages. This value is used only for synchronous callout application correlators, and the default is false if the **-k** parameter is not specified. If you specify the **-k** parameter, you must also specify the value of true or false.

-l *lterm_name*
The name used to override the value in the I/O PCB LTERM field of the IMS application program. The name must be eight characters or less. Optional.

-n *connection_bundle*
When creating a correlator, this parameter specifies a connection bundle name. The name must be 20 characters or less. Ensure that the connection bundle exists before accessing the web service. This parameter is required when creating a non-callout (provider) correlator entry, and can optionally be added to the correlator for a request-response callout application. A single connection bundle can contain both callout and non-callout connection properties.

When updating an existing correlator, this parameter specifies the name of a new connection bundle to use with the correlator.

Restriction: Changes the value of this property in an active correlator do not take effect until the next time the SOAP Gateway server starts.

-o *callout_web_service_timeout*
Specifies the number of milliseconds to wait for a response from the web service. The value must be greater than or equal to 0. The default is 7500. This parameter is only used for callout correlator entries.

-p *operation_name*
The operation name and service name together form the unique identifier for the correlator. The name must be a valid value from the WSDL file. The

operation name and service name together form the unique identifier for the correlator. This parameter is required for all scenarios except when referencing an XSD schema file for WebSphere Business Monitor.

-r *correlator_name*
The name of the correlator you want to update. The correlator name is the URN of the web service. The URN is specified by the soapAction attribute of the soap:operation element in the WSDL file. If the correlator file is in the SOAP Gateway XML directory (*install_dir*/imssoap/xml), you only need to specify the file name. If the correlator file is anywhere else, you must specify the full path to the file as well.

-s *socket_timeout*
Specifies the socket timeout in milliseconds. The timeout value for SOAP Gateway to receive a response from IMS Connect before disconnecting the socket and returning an error. The value must be between 0 and 3,660,000. The socket timeout value must be greater than or equal to the execution timeout value. The default value 0 prevents the socket from timing out. Optional.

Important: Response messages might be dropped if the default IMS Connect default value for execution timeout is used and is smaller than the manually configured socket timeout value.

-t *trancode*
The IMS transaction code of the IMS application that is invoked by the web service. If you use the IMS Connect XML Adapter function, leave it blank. Otherwise, you must specify a transaction code value. The transaction code value must be eight characters or less. If the transaction code is less than eight characters, SOAP Gateway adds blanks to the end of the transaction code value. If you want to add trailing spaces to the transaction code value, enclosed the value with double quotes (for example, "MYTRCD"). SOAP Gateway removes the quotes at run time. If you do not specify a transaction code, no transaction code is added by SOAP Gateway to the input message before it sends the message to IMS.

Note: In a callout application, a transaction code is used only for the response message in an asynchronous request-response interaction. In a synchronous interaction, the response is always returned on the same connection as the request message.

-j *callout_uri*
The URI (Uniform Resource Identifier) used to call out to a Business Event server. This parameter is only used to modify a Business Event correlator file created in Rational Developer for System z.

-v *converter_name*
The converter name if you are using the IMS Connect XML adapter function. The name must be eight characters or less. This parameter is only valid when updating an existing correlator with option **-u**.

Restriction: Existing correlator files that work with an XML adapter in the target IMS Connect can be updated with this utility, but not created. To create a correlator that works with an IMS Connect XML adapter, use Rational Developer for System z.

-w *wsdl_file*
Specifies which WSDL file to use. If the WSDL file is in the SOAP Gateway WSDL directory (*install_dir*/imssoap/wsdl), you only need to specify the file

name. If the WSDL is anywhere else, you must specify the full path to the file as well. This parameter is required when you are creating a correlator file.

Note: For the WebSphere Business Monitor scenario, this parameter specifies the XSD file instead of a WSDL file.

Create example

This example creates a correlator file for the operation `MyOperation` of service `MyService` that is defined in `MyWSDL.wsdl`. The WSDL file is already in the WSDL directory. This example specifies that the web service is to use the connection bundle named bundle1, and that the IMS transaction code is TRAN1.

```
iogmgmt -corr -c -w MyWSDL.wsdl -p MyOperation -i MyService -n bundle1 -t TRAN1
```

Update example: Setting the socket timeout value

This example updates the entry for the MyOperation, MyService web service already defined in `MyCorrelator.xml` to set the socket timeout value to 1000 milliseconds.

```
iogmgmt -corr -u -r MyCorrelator.xml -p MyOperation -i MyService -s 1000
```

Update example: Specifying not to use the XML adapter function

This example updates the entry for the MyOperation, MyService web service already defined in `MyCorrelator.xml` to specify that no adapter is used.

```
iogmgmt -corr -u -r MyCorrelator.xml -p MyOperation -i MyService -a No_Adapter
```

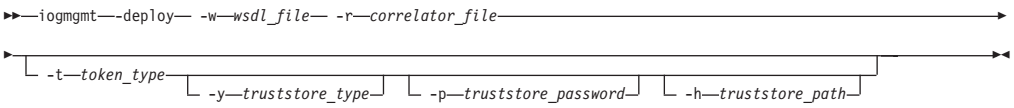
Related tasks:

"Creating a correlator file for a callout application" on page 239
You can manually create a correlator file with the SOAP Gateway management utility if you do not have IBM Rational Developer for System z.

-deploy: Deploy a web service or callout application

The -deploy command deploys a web service, callout application, or business event application to the active configuration of the SOAP Gateway server.

Syntax



Parameters

Use the following parameters to deploy a web service to the server.

-deploy
Specifies the deployment task.

-w *definition_file*
Specifies which WSDL file to use for the web service or application. If the file is in the SOAP Gateway WSDL directory (*install_dir*/imssoap/wsdl), only the file name is required. If the file is anywhere else, you must specify the full path to the file. If the fully qualified path to the file includes spaces, enclose

the entire path in quotation marks: " ". For a callout application sending business event data to WebSphere Business Monitor, specify the XSD file instead of a WSDL file.

Restriction: For the provider scenario, if the specified WSDL file imports additional XSD schema files, those files must be present in the same directory as the WSDL when it is deployed. If nested XSD references are used, all of the referenced XSD files must be in the same directory as the WSDL. Referencing XSD files using absolute paths, paths starting with a period "(.)"', or the `<include>` tag are not supported. Nested WSDL files are not supported.

For the provider scenario, web services can share the same XSD schema files, or XSD files of the same name but different content.

For the consumer or business event scenarios:

- Sharing XSD schema files among web services are not supported.
- Nested XSD import statements are not supported unless you are using Rational Developer for System z Version 8.5.1 or later to generate the COBOL copybook from a web service WSDL file for the synchronous callout scenario.

-r *correlator_file*
Specifies the correlator file to use. If the correlator file is in the SOAP Gateway XML directory (*install_dir*`/imssoap/xml`), only the file name is required. If the correlator file is anywhere else, you must specify the full path to the file. If the fully qualified path to the file includes spaces, enclose it in quotation marks (" "). The path cannot include parentheses. Soft links are not supported.

-t *token_type*
Specifies the type of security token to use.

The valid values for the provider scenario are:

- `SAML11Token`
- `SAML11SignedTokenTrustAny`
- `SAML11SignedTokenTrustOne`
- `SAML20Token`
- `SAML20SignedTokenTrustAny`
- `SAML20SignedTokenTrustOne`
- `UserNameToken`

`SAML11SignedTokenTrustAny` or `SAML20SignedTokenTrustAny` indicates the signed SAML tokens are trusted regardless of the signer. `SAML11SignedTokenTrustOne` or `SAML20SignedTokenTrustOne` indicates that signature must be verified with the certificates in a specified truststore. A truststore, its password, and the absolute path to the truststore must be provided.

The valid values for the synchronous callout scenario are:

- `SAML11Token`
- `SAML20Token`

Specifying a value for this parameter enables WS-Security for the provider web service or synchronous callout web service. If this parameter is not used, WS-Security is disabled for the service or application being deployed.

-y *truststore_type*
Specifies the type of the truststore that contains the certificates to use to verify the signature in SAML tokens. The valid values are `JCEKS`, `JKS`, and `PKCS12`.

When the token type is set to SAML11SignedTokenTrustOne or SAML20SignedTokenTrustOne, a truststore, its password, and the absolute path to the truststore must be provided.

-p *truststore_password*
Specifies the password for the truststore that contains the certificates to use to verify the signature in SAML tokens. When the token type is set to SAML11SignedTokenTrustOne or SAML20SignedTokenTrustOne, a truststore, its password, and the absolute path to the truststore must be provided.

-h *truststore_path*
Specifies the absolute path to the truststore that contains the certificates to use to verify the signature in SAML tokens. When the token type is set to SAML11SignedTokenTrustOne or SAML20SignedTokenTrustOne, a truststore, its password, and the absolute path to the truststore must be provided.

Example: Deploying a web service without WS-Security

The following example deploys the file MyWSDL.wsdl, which is in *install_dir*/imssoap/wsdl (otherwise, the fully qualified path must be specified). The correlator file for this web service is MyCorrelator.xml in *install_dir*/imssoap/xml directory (otherwise, the fully qualified path must be specified). WS-Security is not enabled.

```
iogmgmt -deploy -w MyWSDL.wsdl -r MyCorrelator.xml
```

Example: Deploying a web service that requires a SAML 2.0 unsigned token

The following example deploys a web service with WS-Security enabled, and the token type is SAML 2.0 unsigned token.

```
iogmgmt -deploy -w MyWSDL.wsdl -r MyCorrelator.xml -t SAML20Token
```

Example: Deploying a web service that requires a SAML 1.1 signed token

The following example deploys a web service with WS-Security enabled, and the token type is SAML 1.1 signed token. The signature in the token must be verified with the certificates in the specified truststore before the token can be trusted.

```
iogmgmt -deploy -w MyWSDL.wsdl -r MyCorrelator.xml -t SAML11SignedTokenTrustOne
-y JCEKS –p MyPassword –h /u/ssl/trust.jceks
```

Example: Deploying a callout application

The following example deploys a callout application.

```
iogmgmt -deploy -w myCalloutWSDL.wsdl -r myCalloutCorrelator.xml
```

Example: Deploying a callout application with a SAML 2.0 unsigned token

The following example deploys a callout web service that would pass the user ID information for the user that initiates the IMS callout request in a SAML 2.0 unsigned token to the external web service.

```
iogmgmt -deploy -w myCalloutWSDL.wsdl -r myCalloutCorrelator.xml -t SAML20Token
```

Related tasks:

"Deploying a web service to SOAP Gateway" on page 327
A web service must be deployed to the SOAP Gateway server before it is available

to client applications.

"Deploying a callout application to SOAP Gateway" on page 267
Deploy a callout application or business event emitter to SOAP Gateway with the SOAP Gateway management utility.

-diagnose: Diagnose SOAP Gateway problems

The -diagnose command provides logs that are used by IBM Software Support to troubleshoot problems with SOAP Gateway.

Syntax

Note: The information produced by this command is intended for use by IBM Software Support. You might be instructed to use this command by IBM Software Support or by an error message that requires diagnostic information to resolve.

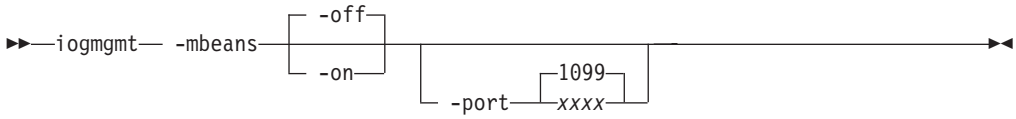
Parameters

-diagnose
Generate diagnostic information.

-mbeans: Configure SOAP Gateway JMX monitoring

Use the -mbeans command to switch the SOAP Gateway server JMX monitoring MBeans on or off and set the port number for JVM monitoring.

Syntax



Usage

This command enables and disables the following monitoring functions for the SOAP Gateway server:

JVM monitoring
Standard JVM health-check information. This information includes heap memory and CPU utilization.

Apache Tomcat MBeans for Catalina monitoring
The Apache Tomcat servlet container, Catalina, is instrumented with JMX MBeans.

SOAP Gateway MBean for web service provider monitoring
The SOAP Gateway MBean, SOAPGatewayProviderMonitorMBean, provides an interface to get statistics about SOAP Gateway web service invocations, current workload, connections and connection bundles, and deployed services.

Changes made with this command do not take effect until the next time that the SOAP Gateway starts. You can set the port number for monitoring, enable or

disable monitoring, or both with one command.

Parameters

-mbeans
Change JMX monitoring configuration. The shortcut for this keyword is -mb.

-off
Disable the JMX monitoring functions. By default, JMX monitoring is disabled.

-on
Enable the JMX monitoring functions.

-port
Specify the JMX connection port. This port number cannot be shared with the server listening port, secure listening port, or shutdown port. If the port is not available when SOAP Gateway starts, the JMX interface is disabled. The shortcut for this keyword is -p.

Example

The following example enables JMX monitoring for the server on port 9090:

```
iogmgmt -mbeans -on -port 9090
```

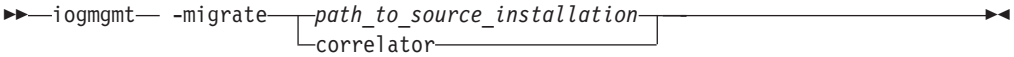
-migrate: Migrate and upgrade SOAP Gateway

The -migrate command upgrades SOAP Gateway artifacts and settings to the latest version and generates a migration log.

Syntax

Important: View the APAR information, online readme, and relevant documentation for the latest release before using this command. This command might perform different migration tasks depending on the changes required for the latest release.

The command generates a migration log file `migration.log` in the same location as the SOAP Gateway server log. Check the log for any migration issue.



Parameters

-migrate
Perform migration tasks for the latest release level. Either the *path_to_source_installation* parameter or the correlator keyword must be specified.

path_to_source_installation
Specify the absolute path to the installation of IMS Enterprise Suite Version 2.1 or Version 2.2 SOAP Gateway to migrate web services and server properties to the new installation. The previous installation must be on the same system or the same direct access storage device (DASD) as the new installation.

Windows On Windows systems, enclose the path in quotes.

You can also clone a Version 3.1 SOAP Gateway server by specify the absolute path to where the server component of the IMS Enterprise Suite Version 3.1 SOAP Gateway server master copy is installed. The two server instances must be on the same system or the same DASD.

The following files are copies from the source installation over to the current installation:

- Connection bundle files are copied into the *install_dir*/imssoap/xml directory.
- Correlator files are checked, updated to the current schema, and stored in the *install_dir*/imssoap/xml directory.
- WSDL and XSD files, if they exist in the source directory, are not copied over, but are embedded in the web service AAR files. If the source WSDL or XSD file for a web service is not found, the web service AAR file is directly copied into the *install_dir*/imssoap/WEB-INF/services directory.

The migrated web services are automatically deployed.

The following server properties or files from Version 2.1 or Version 2.2 are migrated to the new server version:

- All properties configured through the SOAP Gateway management utility are migrated over:
- Web Services Security server policy and bindings are copied over.
- Connection bundles are copied over.
- The wsjaas.conf file is copied over.
- The following properties in the log4j.properties file are migrated:
  - log4j.appender.LOGFILE.File
  - log4j.logger.com.ibm.ims
  - log4j.appender.CONSOLE.Threshold
  - log4j.appender.LOGFILE.Threshold
  - log4j.appender.LOGFILE.encoding

Important: The -migrate command does not validate the existing values. If an invalid value was previously manually added, the problem would not surface until during run time.

A migration report is saved in the same directory as the SOAP Gateway server log. Check the migration report for the migration results. For failed cases, manually deploy the services by using the iogmgmt -deploy command. For more information, see "Migrating from IMS Enterprise Suite Version 2.1 SOAP Gateway" on page 104 or "Migrating from IMS Enterprise Suite Version 2.2 SOAP Gateway" on page 106.

correlator
This keyword indicates that only the correlator files are to be migrated to the new schema (correlator version 3.0). The correlator files to be migrated must reside in the SOAP Gateway *install_dir*/imssoap/xml directory. .

Example: Migrating from Version 2.2 to Version 3.1

In the following example for Linux for System z, the -migrate command is issued from the *install_dir*/imsserver/deploy directory of a new Version 3.1 installation to migrate Version 2.2 web services and server properties.

```
./iogmgmt -migrate /opt/IBM/IMS_Enterprise_Suite_V2.2/SOAP_Gateway/imsserver
```

Example: Cloning a Version 3.1 SOAP Gateway server

In the following example, the -migrate command is issued from the *install_dir*/imsserver/deploy directory of a newly installed Version 3.1 server to copy the web services and server properties from a master Version 3.1 server. For the source server, specify the absolute path to the directory where the **imsserver** component is installed.

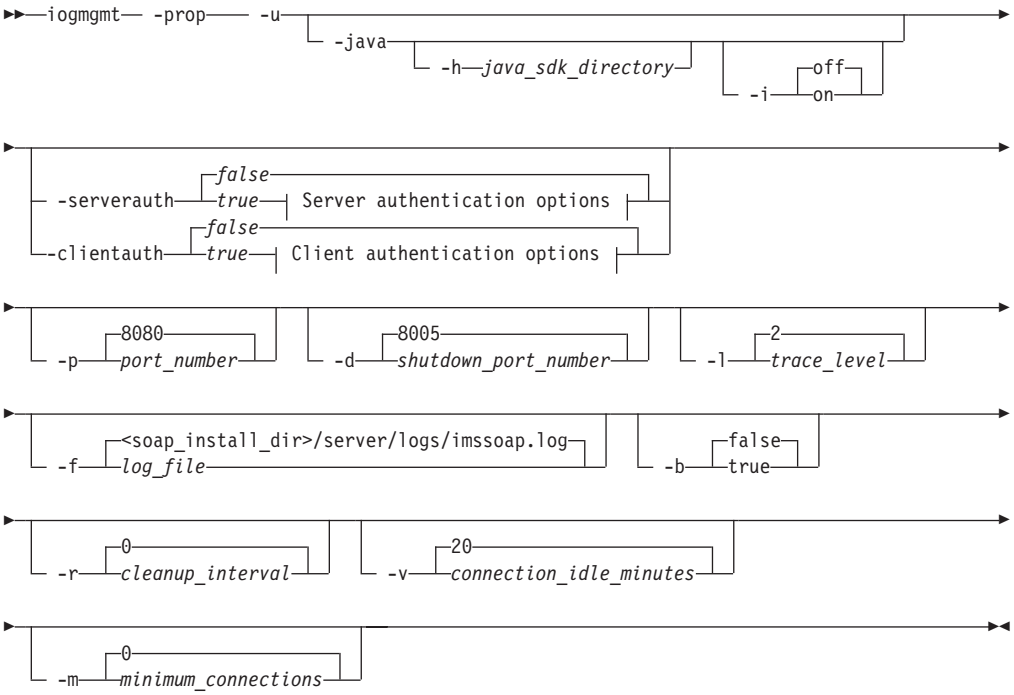
```
./iogmgmt -migrate /opt/IBM/IMS_Enterprise_Suite_V3.1/imsserver/
```

-prop: Set SOAP Gateway properties

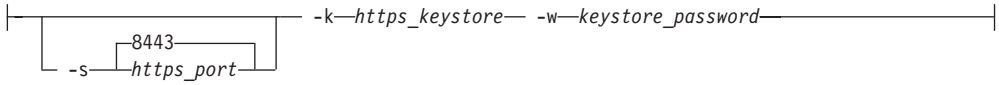
Use the -prop command to modify the SOAP Gateway server properties.

Syntax

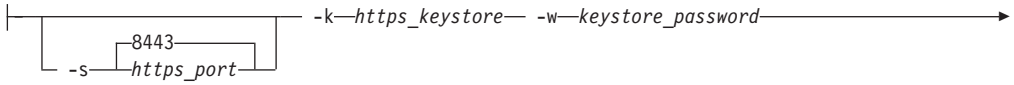
Changes made with this command do not take effect until the next time the SOAP Gateway server starts.



Server authentication options:



Client authentication options:



Parameters

Use the following parameters to modify the SOAP Gateway server properties.

-prop
This parameter specifies to change SOAP Gateway server properties.

-u
Update server properties. The **-u** option must be specified with this command.

-java
This parameter specifies the change the Java Runtime Environment settings for the SOAP Gateway server. You must specify one or all of the following configuration parameters:

-h *java_sdk_directory*
Specifies a new path to an instance of the IBM SDK for Java Technology.

-i off | on
Changes the Java IFA setting for the SOAP Gateway server. Set this parameter to `on` to allow the server to use a System z Application Assist Processor (zAAP) if one is available. The default is `off`.

-clientauth false | true
This parameter activates or deactivates client authentication on the SOAP Gateway server. The valid values are `true` and `false`. Server authentication and client authentication are mutually exclusive. If one authentication type is enabled, you must disable it before you enable the other authentication type.

-serverauth false | true
This parameter activates or deactivates server authentication on the SOAP Gateway server. The valid values are `true` and `false`. Server authentication and client authentication are mutually exclusive. If one authentication type is enabled, you must disable it before you enable the other authentication type.

-b false | true
Specifies whether to pass only the SOAP body (true), or the entire SOAP message (false). The default is false.

-f *log_file*
Specifies the fully qualified path for the SOAP Gateway log files location. The default location for log files on z/OS is *-PathPrefix-*`/usr/lpp/ims/imses/V`*x*`R`*x*`/soap_gw/imsbase/logs`. The default location for the log files on all other platforms is *install_dir*`/imsbase/logs`.

-k *https_keystore*
Specifies the fully qualified address and file name for the SSL keystore. The file extension of the keystore must be `.ks`.

-l *trace_level*
Specifies the trace level. The trace level defines how much information is written to the internal SOAP Gateway server log file. The trace level can be one of the following values:

Table 41. Trace level settings for the SOAP Gateway server log file

Trace level value	Trace level	Description
0	z/OS Off Linux Windows This trace level is not valid	• z/OS The SOAP Gateway server log file and console appenders are disabled. Messages are not written to the SOAP Gateway server log file or job log. Error and fatal messages are still sent to WTO. • Linux Windows A command to set this trace level results in an IOGD0650W warning message. The command is ignored.
1	Fatal	Only errors that result in an immediate server shutdown are logged.
2	Error	• In addition to logging errors that result in an immediate server shutdown, other errors and exceptions are also logged. • NACK responses for synchronous callout applications that use the send-only with acknowledgement protocol are logged. • This is the default trace level.
3	Warn	In addition to what is logged at the error level, warning messages are also logged.
4	Information	• In addition to what is logged at the warning level, the entry and exit of important events and functions are also logged. • All ACK and NACK messages for synchronous callout applications that use the send-only with acknowledgement protocol are logged.
5	Debug	In addition to what is logged at the information level, the contents of buffers sent to and received from IMS Connect and SOAP Gateway are also logged.

-p *port_number*
Specifies the port number for the SOAP Gateway server. SOAP Gateway is configured to run and listen on the SOAP request over the HTTP communication protocol using port 8080 by default. The port number must be a positive number.

-d *shutdown_port_number*
Specifies the dedicated shutdown port for the SOAP Gateway server. The default is port 8005.

-s *https_port*
Specifies the HTTPS port number. The default HTTPS port number is 8443. Valid port numbers are 0 - 65535.

-t *https_truststore*
Specifies the fully qualified address and file name for the SSL truststore. The file extension of the truststore must be `.ks`.

-a *truststore_password*
Specifies the truststore password. The password must be 6 - 20 characters long.

-w *keystore_password*
Specifies the keystore password. The password must be 6 - 20 characters long.

-r *cleanup_interval*
Specifies the idle connection cleanup interval, in minutes. The default value (0) disables the idle connection cleanup feature. A positive value enables idle connection cleanup. Negative values are invalid.

-v *connection_idle_minutes*
Specifies the maximum connection idle time, in minutes. Connections that have been idle for longer than this value will be removed at the next idle connection cleanup interval. Specify this value only if idle connection cleanup is enabled. This value must be greater than or equal to 1 minute. The default value is 20 minutes.

-m *minimum_connections*
Specifies the minimum number of connections to keep in each connection pool when idle connection cleanup is enabled. The default value is 0.

Example: Update server port and logging

The following example updates the HTTP port number for the SOAP Gateway server to 8081 and disables logging.

```
iogmgmt –prop -u –p 8081 –l 0
```

Example: Enable client authentication

The following example enables SSL on the default port (8443) with client authentication.

```
iogmgmt -prop -u -clientauth true -s 8443 -k keystore_loc
-w keystore_pwd -t truststore_loc -a truststore_pwd
```

Example: Change HTTPS port

The following example changes the HTTPS port number on a SOAP Gateway server to 8993. The required keystore name, keystore password, truststore name and truststore password must be specified.

```
iogmgmt -prop -u -clientauth true -s 8993 -k keystore_loc
 -w keystore_pwd -t truststore_loc -a truststore_pwd
```

Example: Change log file location

The following example changes the server log file location to `C:\logs`.

```
iogmgmt -prop -u -f C:\logs
```

Related tasks:

"Changing the port number of the SOAP Gateway server" on page 295
To change the port number of SOAP Gateway, use the SOAP Gateway management utility.

Related information:

"IOGD0084E" on page 386
The *command_name* command to enable client authentication failed because an SSL port number, keystore name, keystore password, truststore name, and truststore password must be specified.

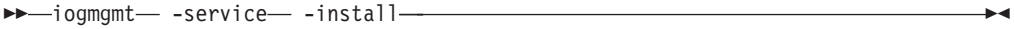
-service -install: Install the SOAP Gateway server as a Windows service

Windows

Use the -service -install command to install and register the SOAP Gateway server as a Windows service.

On Windows 7 systems, depending on the SOAP Gateway installation directory, you might need to open a command prompt as an administrator and change directories to go to the SOAP Gateway management utility directory (*install_dir*/imsserver/deploy) before you can issue any command.

Syntax



Parameters

-service -install
Specifies to install the SOAP Gateway server as a Windows service.

This command registers the IMSSOAPGateway service as a Windows service. SOAP Gateway must have been installed by using the IBM Installation Manager.

-service -start: Start the SOAP Gateway server as a Windows service

Windows

Use the -service -start command to start the SOAP Gateway server as a Windows service.

Syntax

Parameters

-service -start
Specifies to start the SOAP Gateway server as a Windows service.

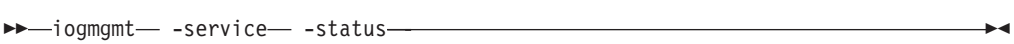
SOAP Gateway must have been registered as a Windows service by using the iogmgmt -service -install command.

-service -status: View the SOAP Gateway server status as a Windows service

Windows

Use the -service -status command to view the SOAP Gateway server status as a Windows service.

Syntax



Parameters

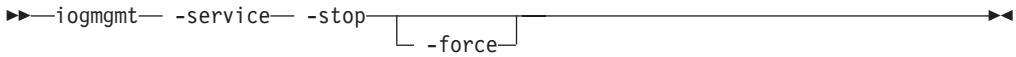
-service -status
Specifies to view the SOAP Gateway server status as a Windows service.

-service -stop: Stop the SOAP Gateway server as a Windows service

Windows

Use the -service -stop command to stop the SOAP Gateway server as a Windows service.

Syntax



Parameters

-service -stop
Specifies to gracefully stop the SOAP Gateway server as a Windows service.

-force
Specifies to stop the SOAP Gateway server as a Windows service immediately without waiting for in-flight messages to process.

Example

To stop the Windows service gracefully:

```
iogmgmt -service -stop
```

To stop the Windows service immediately:

```
iogmgmt -service -stop -force
```

Related concepts:

"SOAP Gateway server shutdown options" on page 293
There are different methods to stop the SOAP Gateway server, depending on the host operating system.

Related reference:

"-service -start: Start the SOAP Gateway server as a Windows service" on page 454
Use the -service -start command to start the SOAP Gateway server as a Windows service.

-service -uninstall: Unistall the SOAP Gateway server as a Windows service

Windows

Use the -service -uninstall command to remove the SOAP Gateway server as a Windows service.

Syntax

Parameters

-service -uninstall
Specifies to uninstall the SOAP Gateway server as a Windows service.

-start: Start the SOAP Gateway server

The -start command starts the SOAP Gateway server.

Syntax

Restriction:

This command is not valid for SOAP Gateway servers running on z/OS. On z/OS systems, use the START AEWIOGPR z/OS console command.

For SOAP Gateway servers running as Windows services, use the iogmgmt -service -start command.

```
►►—iogmgmt— -start————————————————————————————————————►◄
```

Parameters

-start
Specifies the task to start the server

Example

```
iogmgmt -start
```

Related concepts:

"SOAP Gateway server startup options" on page 292
There are different methods to start the SOAP Gateway server depending on the host operating system.

Related reference:

"-service -start: Start the SOAP Gateway server as a Windows service" on page 454
Use the -service -start command to start the SOAP Gateway server as a Windows service.

-stop: Stop the SOAP Gateway server

The -stop command stops the SOAP Gateway server.

When you issue this command, SOAP Gateway would try to shut down the server gracefully. *Graceful shutdown* means that SOAP Gateway would block all incoming web service requests and communicate to IMS Connect to stop further IMS callout requests or business event emission destined for SOAP Gateway. SOAP Gateway would wait for a maximum of 5 minutes for all in-flight messages to be processed before it shuts down. If necessary, you can use the *forced shutdown* option (-stop -force) to shut down the server immediately.

Tips:

z/OS The -stop command is not valid for SOAP Gateway servers that run on z/OS. On z/OS systems:

- Use the STOP AEWIOGPR z/OS console command to stop the server gracefully. Replace AEWIOGPR with your custom job name if it is named differently.
- Use the CANCEL AEWIOGPR z/OS console command to force an immediate shutdown.

Windows For SOAP Gateway servers that run as Windows services, use the iogmgmt -service -stop command.

Syntax

Parameters

-stop
Stop the server gracefully.

-force
Stop the server immediately and discard all work in progress.

Restriction:

- This keyword is only valid for SOAP Gateway servers that run on Linux, if the server is started by using the iogmgmt -start command or the desktop shortcut. If the server was started by running the `iogstart.sh` script in the *install_dir*`/imsserver/bin` directory, forced shutdown option is not supported.
- This keyword is not supported for SOAP Gateway servers that run in a foreground window mode on Windows platforms. If the SOAP Gateway server runs as a Windows service, use the iogmgmt -service -stop -force command instead.
- On z/OS, use the CANCEL AEWIOGPR command instead.

Example

To stop the SOAP Gateway server gracefully on distributed platforms:

```
iogmgmt -stop
```

To stop the SOAP Gateway server immediately on Linux platforms:

```
iogmgmt -stop -force
```

Related concepts:

"SOAP Gateway server shutdown options" on page 293
There are different methods to stop the SOAP Gateway server, depending on the host operating system.

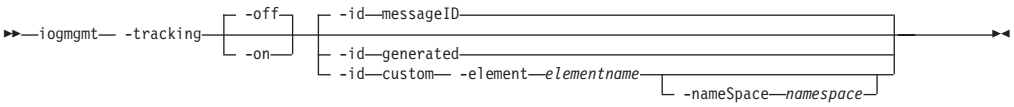
Related reference:

"-service -stop: Stop the SOAP Gateway server as a Windows service" on page 455
Use the -service -stop command to stop the SOAP Gateway server as a Windows service.

-tracking: Configure SOAP Gateway-to-IMS transaction tracking IDs

Use the -tracking command to enable, disable, and configure the horizontal tracking IDs for inbound SOAP messages (the provider scenario).

Syntax



Parameters

-tracking
Change the SOAP Gateway horizontal tracking ID configuration for the provider scenario. The shortcut for this keyword is **-tr**.

Setting with this parameter does not affect callout transaction logging because callout transaction logging uses vertical tracking IDs only.

-on
Enables SOAP Gateway horizontal tracking IDs. When IDs are enabled, SOAP Gateway adds a tracking ID to every inbound request message (the provider scenario). This tracking ID is used by the SOAP Gateway transaction logger to correlate messages at multiple stages of request and response message processing. The ID is also associated with processing activity for the message in IMS Connect and IMS.

-off
Disables tracking IDs. By default, SOAP Gateway does not add an ID to messages.

-id
Sets the type of identifier that SOAP Gateway adds to incoming messages. This parameter applies to inbound message tracking for the provider scenario only. For callout messages, the tracking ID is always generated.

The valid values are:

messageID
SOAP Gateway uses the value from the WS-addressing `messageID` element of the inbound SOAP request message header as the tracking ID. This value is generated by the client application. If the `messageID` element does not exist, is empty, or is longer than 40 bytes, SOAP Gateway generates an ID for the message instead. When message IDs are activated, **messageID** is the default tracking ID type.

generated
SOAP Gateway generates a unique tracking ID for every inbound SOAP request message.

custom
SOAP Gateway retrieves the value of a user-specified SOAP element to use as the tracking ID. If the value of the user-specified element does not exist, is empty, or is longer than 40 bytes, SOAP Gateway generates a tracking ID for the message. This tracking ID type has one required parameter and one optional parameter:

-element
Specifies the name of the target SOAP message element. Only the value of the element is used for the tracking ID. If any attributes are set for the target element in the message, they are ignored. When the tracking ID type is set to **custom**, **-element** is a required parameter. It is not valid with any other tracking ID type. The shortcut for this keyword is -e.

-nameSpace
Specifies a namespace for the target element. SOAP Gateway attempts to find an exact match for the namespace of the target element (specified with the **-element** parameter) in the SOAP message. If an exact match is not found, SOAP Gateway generates an ID instead. When the tracking ID type is set to **custom**, **-nameSpace** is an optional parameter. It is not valid with any other tracking ID type. The shortcut for this keyword is -ns.

Example

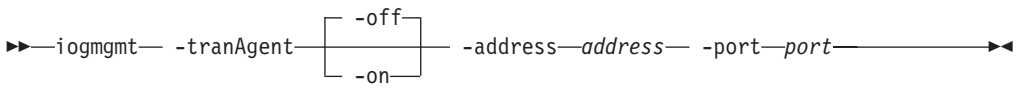
The following example enables uniquely generated tracking IDs for incoming request messages:

```
iogmgmt -tracking -on -id generated
```

-tranAgent: Configure the IBM Tivoli Composite Application Manager for Transactions (ITCAM) Transaction Tracking API (TTAPI)

Use the -tranAgent command to enable or disable the IBM Tivoli Composite Application Manager for Transactions (ITCAM) Transaction Tracking API (TTAPI) and set the address for the target data collection server.

Syntax



Usage

For the provider scenario, to enable SOAP Gateway-to-IMS transaction tracking, issue the iogmgmt -tracking -on command first for SOAP Gateway to send the horizontal IDs.

This feature requires a remote IBM Tivoli Composite Application Manager for Transactions (ITCAM) data collector to receive the information that is sent by the SOAP Gateway implementation of the ITCAM TTAPI.

Parameters

-tranAgent
Configures the ITCAM TTAPI for the SOAP Gateway server. The shortcut for this keyword is -ta.

-on
Enables the ITCAM TTAPI.

-off
Disables the ITCAM TTAPI. By default, the ITCAM TTAPI is disabled.

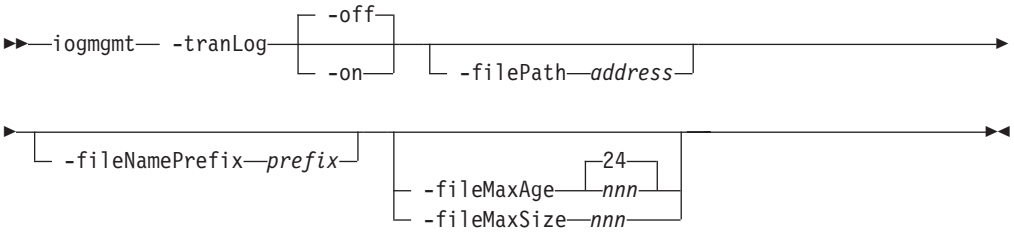
-address
Specifies the host name for the target ITCAM data collector. You can specify an IP address or a fully qualified domain name. The shortcut for this keyword is -a.

-port
Specifies the port number for the target ITCAM data collector. The shortcut for this keyword is -p.

-tranLog: Configure the SOAP Gateway transaction logger

Use the -tranLog command to enable or disable the SOAP Gateway transaction logger for both web service provider and callout transactions.

Syntax



Parameters

-tranLog
Configures the SOAP Gateway transaction logger. The shortcut for this keyword is **-tl**.

-on
Activates the SOAP Gateway transaction logger.

-off
Deactivates the SOAP Gateway transaction logger. Any existing transaction log files are not altered or deleted, but new transactions are not logged. By default, the transaction logger is disabled.

-filePath *path*
Specifies a fully qualified directory path where SOAP Gateway saves transaction log files. By default, this location is the server log directory. The shortcut for this keyword is **-fp**.

-fileNamePrefix *prefix*
Specifies a prefix for the transaction log file names to differentiate them from other files in the directory. By default, the log file prefix is `tranLog`. The shortcut for this keyword is **-f**.

-fileMaxSize *nnn*
Specifies a maximum size for the transaction log files. This value is specified in megabytes, and the minimum is 1 megabyte. After the active log file reaches the specified size, a new log file is created.

This parameter is mutually exclusive with **-fileMaxAge**. If values are specified for both parameters, or if there is already a value specified for one parameter when this command is issued, the last value overrides any previous settings. For example, if **-fileMaxSize** is set to 10 megabytes and **-fileMaxAge** is then set to 24 hours, the setting for **-fileMaxSize** is discarded.

The shortcut for this keyword is **-fs**.

-fileMaxAge *nnnUnit*
Specifies a maximum age for the transaction log files. Specify the units in

hours (H), minutes (M), or seconds (S). The unit must follow the numeric value immediately without a space. After the active log file reaches the specified age, a new log file is created. By default, **-fileMaxAge** is 24 hours (24H).

This parameter is mutually exclusive with **-fileMaxSize**.

The shortcut for this keyword is **-fa**.

Example

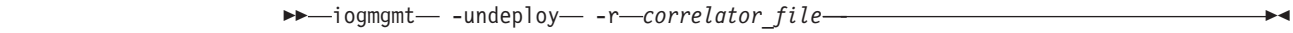
This example activates the transaction logger, and sets the maximum log file age to one hour. A new transaction log file is created every hour after the command is issued.

```
iogmgmt -tranLog -on -fileMaxAge 1H
```

-undeploy: Undeploy a web service or callout application

Use the -undeploy command to undeploy a web service or callout application. Undeploying a service removes the XML file for the service from the runtime cache and master configuration.

Syntax



Parameters

Use the following parameters to undeploy a web service.

-undeploy
Undeploy a web service. The correlator XML file for the service is removed from the runtime cache and master configuration.

For web service provider scenario, the WSDL file and any imported XSD files in the master configuration are left untouched when the web service is undeployed.

-r *correlator_file*
Specifies the correlator file name of the web service to undeploy. Because the correlator file for a deployed web service must be in the SOAP Gateway XML directory (*install_dir*/imssoap/xml), you do not need to specify the path. If you must use fully qualified path to the file:

- The path must point to the correlator file in the SOAP Gateway XML directory.
- If the path contains spaces, enclose the whole path in quotation marks (" ").
- Soft links are not supported.

Example

This example undeploys a web service defined by the service name and operation name in MyCorr.xml by removing the associated web service archive file and correlator XML file from the server master configuration and runtime configuration.

```
iogmgmt -undeploy -r MyCorr.xml
```

-view -connectionbundle: View all connection bundle entries

Use the -view -connectionbundle command to view all connection bundle entries in the server master configuration.

Syntax

Parameters

-ConnectionBundle
View all connection bundles in the SOAP Gateway master configuration. The shortcut for this keyword is -cb.

Related concepts:

Chapter 8, "Administering the SOAP Gateway server," on page 291
Administer the SOAP Gateway server with the SOAP Gateway management utility.

-view -connectionbundleentry: View a connection bundle entry

Use the -view -connectionbundleentry command to view a specific connection bundle entry.

Syntax

Parameters

-ConnectionbundleEntry *connection_bundle_name*
View a specific connection bundle entry by name. The shortcuts for this keyword are-ce and-n.

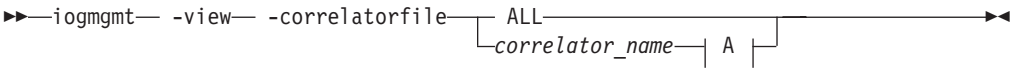
Related concepts:

Chapter 8, "Administering the SOAP Gateway server," on page 291
Administer the SOAP Gateway server with the SOAP Gateway management utility.

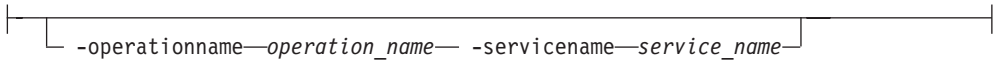
-view -correlatorfile: View correlator information

Use the -view -correlatorfile command to view correlator information in either the runtime or master configuration.

Syntax



A:



Parameters

-correlatorfile ALL
View a list of all correlator files in the runtime configuration if the server is started, or the master configuration if it is stopped. The shortcut for this keyword is `-cf` or `-r`.

-correlatorfile *correlator_name*
View detailed correlator information in a single correlator XML file in the runtime configuration if the server is started, or the master configuration if it is stopped. The shortcut for this keyword is `-cf` or `-r`.

-operationname *operation_name*
View information about a specific operation name in the correlator file. The shortcuts for this keyword are `-p` and `-opr`.

-servicename *service_name*
View information about a specific service name in the correlator file. The shortcuts for this keyword are `-i` and `-svcn`.

Related concepts:

Chapter 8, "Administering the SOAP Gateway server," on page 291
Administer the SOAP Gateway server with the SOAP Gateway management utility.

-view -calloutproperties: View callout properties

Use the -view -calloutproperties command to view the current configuration of SOAP Gateway callout properties.

Syntax

Parameters

-CalloutProperties
View the current configuration of the SOAP Gateway callout properties. The shortcut for this keyword is `-cp`.

Related concepts:

"Thread management for callout messages retrieval" on page 174
SOAP Gateway supports two options to determine how to manage the callout threads to send the requests to poll the hold queue for callout request messages: one thread per tpipe, or one thread per connection bundle.

Chapter 8, "Administering the SOAP Gateway server," on page 291
Administer the SOAP Gateway server with the SOAP Gateway management utility.

-view -calloutthreads: View the status of callout threads

Use the -view -calloutthreads command to view the status of all active callout threads.

Syntax

Parameters

-CalloutThreads
View the status of the callout threads. The shortcut for this keyword is `-ct`.

Related concepts:

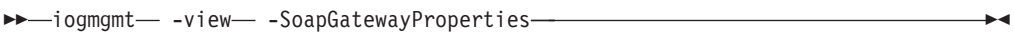
"Thread management for callout messages retrieval" on page 174
SOAP Gateway supports two options to determine how to manage the callout threads to send the requests to poll the hold queue for callout request messages: one thread per tpipe, or one thread per connection bundle.

Chapter 8, "Administering the SOAP Gateway server," on page 291
Administer the SOAP Gateway server with the SOAP Gateway management utility.

-view -soapgatewayproperties: View SOAP Gateway server properties

Use the -view -soapgatewayproperties command to view the configuration of the SOAP Gateway server.

Syntax



Parameters

-SoapGatewayProperties
View the current configuration of the SOAP Gateway server properties. The shortcut for this keyword is `-sgp`.

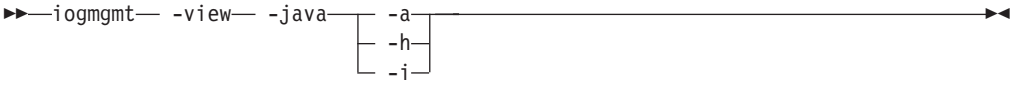
Related concepts:

Chapter 8, "Administering the SOAP Gateway server," on page 291
Administer the SOAP Gateway server with the SOAP Gateway management utility.

-view -java: View SOAP Gateway Java properties

Use the -view -java command to view the current configuration of the Java Runtime Environment for SOAP Gateway.

Syntax



Parameters

Submit the command `iogmgmt -view -java` with one of the following options:

-a View all configurable Java properties for the server.

-h View the Java home directory.

-i View the Java IFA setting.

-view -workerthreads: View status of the callout worker thread pool

Use the -view -workerthreads command to get the current status of the callout worker thread pool.

Syntax

Parameters

-WorkerThreads
View the status of the callout worker threads in the thread pool. The shortcut for this keyword is `-wt`.

Tip: This information is saved as a dump file in the directory specified with the `-10` parameter of the `iogmgmt -callout -updateprop` command. You can verify the current directory by issuing the `iogmgmt -view -calloutproperties` command.

Related concepts:

Chapter 8, "Administering the SOAP Gateway server," on page 291
Administer the SOAP Gateway server with the SOAP Gateway management utility.

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 may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
J46A/G4
555 Bailey Avenue
San Jose, CA 95141-1003
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 information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurement may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrates programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating

platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS," without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

© (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. _enter the year or years_. All rights reserved.

Privacy policy considerations

IBM Software products, including software as a service solutions, (“Software Offerings”) may use cookies or other technologies to collect product usage information, to help improve the end user experience, to tailor interactions with the end user or for other purposes. In many cases no personally identifiable information is collected by the Software Offerings. Some of our Software Offerings can help enable you to collect personally identifiable information. If this Software Offering uses cookies to collect personally identifiable information, specific information about this offering’s use of cookies is set forth below.

This Software Offering does not use cookies or other technologies to collect personally identifiable information.

If the configurations deployed for this Software Offering provide you as customer the ability to collect personally identifiable information from end users via cookies and other technologies, you should seek your own legal advice about any laws applicable to such data collection, including any requirements for notice and consent.

For more information about the use of various technologies, including cookies, for these purposes, See IBM’s Privacy Policy at http://www.ibm.com/privacy and IBM’s Online Privacy Statement at http://www.ibm.com/privacy/details the section entitled “Cookies, Web Beacons and Other Technologies” and the “IBM Software Products and Software-as-a-Service Privacy Statement” at http://www.ibm.com/software/info/product-privacy.

Trademarks

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

The following terms are trademarks or registered trademarks of other companies, and have been used at least once in this information:

- Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.
- Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

- Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc., in the United States, other countries, or both.
- Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.
- UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, or service names may be trademarks or service marks of others.

Index

D

E

F

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

W

X

Z

Product Number: 5655-TDA

Printed in USA

SC19-4112-05

Spine information:

IMS Enterprise Suite **Version 3 Release 1**

SOAP Gateway Administrator's Guide and Reference